

George W. Anderson

"George's practical real world experience shines in this book and should prove invaluable to any enterprise deployment of SAP. I am glad George finally documented these best practices so that the rest of the world can benefit from his insight and know-how."

Jim Dierkes
Principal Consultant
Enterprise Applications Practice
Microsoft Consulting Services

SAP Planning

Best Practices in Implementation

SAMS

George W. Anderson

SAP Planning: Best Practices in Implementation

SAMS

201 West 103rd Street, Indianapolis, Indiana 46290

SAP Planning: Best Practices in Implementation

Copyright © 2003 by Sams Publishing

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

International Standard Book Number: 0-7897-2875-3

Library of Congress Catalog Card Number: 2002117343

Printed in the United States of America

First Printing: June 2003

06 05 04 03 4 3 2 1

Sams offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact

U.S. Corporate and Government Sales

1-800-382-3419

corpsales@pearsontechgroup.com

For sales outside of the U.S., please contact

International Sales

1-317-581-3793

international@pearsontechgroup.com

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Sams Publishing cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The authors and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the CD or programs accompanying it.

Associate Publisher

Michael Stephens

Acquisitions Editor

Loretta Yates

Development Editor

Sean Dixon

Managing Editor

Charlotte Clapp

Project Editor

Matthew Purcell

Copy Editor

Margaret Bersen

Indexer

Erika Millen

Proofreader

Leslie Joseph

Technical Editor

David Braithwaite

Review Editor

William F. Anderson II

Team Coordinator

Cindy Teeters

Interior Designer

Anne Jones

Cover Designer

Anne Jones

Graphics

Tammy Graham

Laura Robbins

Contents at a Glance

Part I	Preparing for Your mySAP Business Suite Implementation	
1	Introduction to SAP Implementation Planning	3
2	What an SAP Project Looks Like	23
3	Crafting Your mySAP Solution Vision	61
4	Designing and Initially Staffing the SAP Technical Support Organization (TSO)	93
Part II	Getting Started—Sizing and Blueprinting	
5	Total Cost of Ownership Analysis in Preparation for Sizing	123
6	Identifying High Availability and Disaster Recovery Requirements	161
7	Sizing: Engaging the SAP Solution Stack Vendors	221
8	Staffing the Technical Support Organization	275
9	Critical SAP Training Requirements	309
10	Developing the SAP Data Center	339
11	Performing mySAP.com Component Installations	381
12	Rounding Out Support for SAP	425
Part III	Moving Into SAP Functional Development	
13	Gaining Control of Change Control	465
14	SAP Systems and Operations Management	503
15	Functional, Integration, and Regression Testing	543
Part IV	Preparing the Production Environment	
16	Systems and Stress Testing	569
17	Preparing for SAP Go-Live	621
	Index	651

Table of Contents

Part I Preparing for Your mySAP Business Suite Implementation	
1 Introduction to SAP Implementation Planning	3
Welcome to SAP Planning!	3
My Audience and Approach.....	5
What Exactly Are “Best Practices and Approaches”?	7
Addressing the Real Challenges of SAP Implementations.....	8
What Is Covered	8
What Is Not Covered	9
What Exactly Is SAP?	9
How Do mySAP.com and mySAP Fit into the Big Picture?	10
SAP Infrastructure Planning—What It’s All About	12
How to Speak SAP—Terms and Terminology	13
Organizing the Book—Parts and Chapters	14
Special Materials Provided by the Book	19
More on Leveraging “Tools and Techniques”	20
The MCL and SIPP	21
Summary	22
2 What an SAP Project Looks Like	23
In the Beginning	23
The Business Needs Help!	24
Tactical Reasons for Implementing SAP	26
Strategic “Big Picture” Goals of Implementing mySAP.com	27
Application Integration	28
Improved Operational Reporting	28
Improved Strategic Reporting	29
Flexible Business Process Support	29
How SAP Has Benefited Customers in the Real World	30
Beware—When SAP Projects Are Less Than Successful	32
“Not Best” Implementation Practices	32
Ignoring Change Control	35
Hiring an Inexperienced Implementation Partner	36
Setting the Stage for a Successful Implementation	38
Promoting Buy-in Throughout the Company	38
Nailing Down the Real Business Requirements	39
Mapping Business Requirements to Solution Characteristics	40
Determining Realistic Service Level Agreements	41
How to Estimate ROI Early in the Game	42
Translating Solution Characteristics into Technology Drivers	44

The Importance of a Methodology	44
Pinning Down the Initial Implementation Budget	46
The Structure and Role of the Steering Committee	47
Business Unit Buy-in	49
The “Napkin” Plan—Identifying Major Milestones	50
Working Backwards to Develop Timelines	51
A Closer Look at the SAP Solution Stack	51
Giving Attention to Change Control	56
The Role of the SAP Technical Support Organization	56
Moving Beyond the “Napkin” Plan	57
How to Begin Measuring Progress and Success	57
Maintaining the High-Level Project Plan	58
Tools and Techniques	58
Summary	59
3 Crafting Your mySAP Solution Vision	61
What Is a Solution Vision?	61
Impact on the Business	63
Technology Perspective	63
Considering mySAP Components to Be Implemented	65
Considering SAP System Landscape Requirements	65
SAP System Landscape Design and Planning Approaches	67
Simplifying Your SAP System Landscape	68
High Availability and the SAP System Landscape	70
Disaster Recovery Considerations	70
Addressing Training Requirements	71
The Performance-Driven System Landscape	73
Driving Scalability into Your System Landscape	74
The TCO-Driven System Landscape	76
Security Considerations	77
Manageability Considerations	78
The System Landscape and Accessibility	79
Capturing Your mySAP Solution Vision	80
Leveraging SAP Sizing Questionnaires	81
Developing a Request for Information	82
Revisiting the SAP Infrastructure Implementation Budget	84
Outsourcing—Another Method of Achieving Your Solution Vision	84
Prerequisites of ITO—Information Technology Outsourcing	85
Potential Benefits of Outsourcing SAP Infrastructure	86
The Shortcomings of Outsourcing in the Real World	87
Analyzing Outsourcing Versus Doing It Yourself	88
ASP Hosting for SAP	89
HP’s Utility Data Center	89
Tools and Techniques	90
Summary	90

4 Designing and Initially Staffing the SAP Technical Support Organization (TSO)	93
Overview—What Is an SAP Infrastructure TSO?.....	93
The SAP TSO—What It Is Not	95
Leveraging the SAP IT Job Market	97
TSO Overview—Roles and Responsibilities	101
Building the High-Level SAP Project Team	102
How Many PMs Does It Take to Implement SAP?.....	102
Characteristics of a Good SAP Project Manager.....	103
Organizational Chart Approaches in the Real World	103
Customer Example #1, Hardware Versus SAP Groups	104
Customer Example #2, The DBA Cubbyhole	105
Customer Example #3, All Under One Roof	106
Customer Example #4, Leveraging Computer Operations	107
Customer Example #5, Multiple Production Instances	108
SAP TSO Organization Best Practices and Approaches.....	109
Recruiting the Core TSO Leads	112
What to Look for in the Solutions Architect.....	113
Qualities of an SAP Infrastructure/Basis Lead.....	114
Consultants Versus Training Your Own Internal Resources	116
Training Your Own Staff—Values and Headaches	116
Hitting the Ground Running—Consultants Versus Your Budget	117
Compromising and Finding Organizational Balance	118
Consultants Versus Internal Resources in the Real World	118
Revising the SAP Infrastructure Implementation Budget	119
Tools and Techniques	120
Summary	120
Part II Getting Started—Sizing and Blueprinting	
5 Total Cost of Ownership Analysis in Preparation for Sizing	123
Purpose of This Chapter	123
What Is Total Cost of Ownership?.....	124
How the SAP Solution Vision Drives TCO	127
The Impact of High Availability Requirements on TCO.....	128
Disaster Recovery Requirements That Drive TCO	130
Performance Requirements and TCO.....	133
How Scalability Impacts TCO	134
Other SAP Solution Vision TCO Drivers	136
TCO and the SAP Solution Stack	136
Standardization and Total Cost of Ownership.....	137
Hardware TCO—Server Considerations	138
Hardware TCO—Disk Subsystem Considerations	140
Operating System TCO	141

Relational Database TCO	142
Application Layer TCO	145
Solution Stack Upgrade Considerations	148
Other Solution Stack TCO Considerations	149
Lowering TCO Through People and Processes	150
Attracting and Retaining Talented SAP TSO Members	151
Maintenance Costs	152
Financial Terms	153
Operations and Systems Management Costs	153
TCO Risk and Risk Factors	154
Updating the RFI, RFP, or Internal Knowledge Repository	158
Documenting TCO Observations and Key Drivers	158
Tools and Techniques	159
Summary	159
6 Identifying High Availability and Disaster Recovery Requirements	161
Introduction to Availability	161
Causes of Downtime	163
Similarities Between HA and DR	163
How HA and DR Differ	164
Addressing Availability Early in the SAP Project	165
Determining HA Requirements—The “Nines”	166
Availability Planning—Documenting Requirements and Key Drivers	167
Determining Your Required Level of Disaster Tolerance	167
DR and Return on Investment Calculations	168
Calculating the Cost of Downtime	169
Educating the Solution Architect on HA/DR Options	170
Remedying Single Points of Failure in the Stack	171
SAP Data Center Infrastructure SPOFs	172
Power Considerations	173
Network Infrastructure	174
Rack Infrastructure in the Real World	175
The Ultimate SPOF—The SAP Data Center	177
SAP Server SPOFs	178
Clustering	179
Disk Subsystem Single Points of Failure	180
Disk Drives and RAID	181
Cloning and Triple-Mirroring	182
Disk Controllers	184
Disk Subsystem Infrastructure	184
PMD—Physically Moving Drives	185
Campus and Metropolitan Clusters	186
HP Continental Cluster	188
HP/Compaq Data Replication Manager (DRM)	189

Single Points of Failure Inherent in the OS.....	190
Tape Shipping and Disk-to-Disk Copies.....	190
Microsoft Cluster Service—MSCS.....	191
MSCS Issues in the Real World.....	191
HP MC/ServiceGuard and Similar UNIX Cluster Approaches.....	192
HP/Compaq Tru64 Cluster File System	193
Linux Clusters	194
SPOFs and the Database Layer.....	194
Standby Database and Log Replication Solutions.....	195
Oracle Standby Database Offerings	195
HP Oracle Disaster Recovery (ODR).....	196
Oracle Advanced Replication	197
Clustering Microsoft SQL Server.....	197
Oracle 8i Clustering and Failsafe	197
9iRAC—Real Application Clusters.....	198
mySAP.com Single Points of Failure	200
Clustering mySAP.com Components in General.....	201
HP Somersault	202
SAP Replicated Enqueue	203
Specific mySAP.com Components SPOFs	204
Functional and Application-Layer SPOFs	206
SAP General Availability and DR Best Practices	207
SPOFs Beyond the SAP Solution Stack	208
The Disaster Recovery Organization	209
The Disaster Recovery Crash Kit	210
Testing the Disaster Recovery Process	213
Availability Pitfalls and Mishaps in the Real World	214
A Final Look at the SAP System Landscape	217
Preparing for the Next Big Step—Sizing	217
Completing the Knowledge Repository or RFI	218
Revising Your SAP Implementation Budget Again	219
Tools and Techniques	219
Summary	220
7 Sizing: Engaging the SAP Solution Stack Vendors	221
Overview—The Sizing and Blueprinting Process	221
Sharing Your Vision—The RFI and Sizing Questionnaires	222
Obtaining Vendor's SAP Sizing Questionnaires	223
Fostering Apples-to-Apples Sizings	225
Sizing Terminology	227
General Sizing Best Practices and Approaches	230
Understanding Different Sizing Methodologies	232
Sizing Tools, Practices, and Assumptions	234
Best Practices Regarding System Landscape Design	236

Preparing for the SAP Sizing Process	237
The Pre-Sizing Conference Call in the Real World	237
Building Your Sizing Evaluation Team	240
The Sizing Review Process	241
SAP Hardware, OS, and Database Sizing	242
Common Server Considerations for mySAP Sizing	242
Disk RAID Configurations and Features	243
Commonly Deployed Disk Subsystems for SAP	244
Storage Virtualization—The Latest Paradigm in Enterprise	
Computing for SAP	246
Operating System Factors	248
Database Selection and Sizing Factors for SAP	249
Sizing Considerations for mySAP Components	250
Understanding the “New” Basis Layer—Web AS	251
The SAP Exchange Infrastructure	252
Enterprise Portal Sizing Rules of Thumb	253
Sizing R/3 Enterprise	254
Sizing mySAP Business Intelligence	255
Rules of Thumb for Sizing mySAP CRM	257
Sizing Considerations for mySAP PLM	259
Supply Chain Management Sizing Considerations	260
Sizing mySAP SRM	261
mySAP Strategic Enterprise Management	263
Selecting Your SAP Solution Stack Partners	264
Pulling Out “New and Different” Sizing Data	265
Verify Support for Architected Solutions	265
Verifying Your Risk Level	266
Working with SAP Production References	266
Revising Your TCO Analyses	266
Selecting Your Core SAP Solution Stack Partners	267
Evaluating Specialized Solution Stack Vendors	268
Executing the SAP Infrastructure Planning Sessions	268
Day One	269
The Second Day	270
On the Third Day	272
Tools and Techniques	274
Summary	274
8 Staffing the Technical Support Organization	275
Best Practices Approach to Staffing the SAP TSO	275
Three Ways to Approach Staffing	277
Staffing the SAP TSO: Rapid Deployment	278
Best Practices and the Rapid Deployment Approach to	
Staffing	279
Step 1—Developing and Posting Job Descriptions	281

Step 2—How to Evaluate SAP Technical Resumes	281
Step 3—How to Perform an SAP Technical Phone Screen	284
Step 4—Setting up and Holding Interviews	285
Step 5—Key Interview Techniques and Approaches	286
Basic Sample Interview Questions	287
Advanced Skills Interview Questions	288
Step 6—How to Rank Prospective SAP Candidates	289
Interviews in the Real World	291
Bringing in the Best of the Best	293
Internal Transfers	293
External Consultants and Contractors	294
External Employee New-Hires	296
The First Week	297
How to Retain Your SAP Staff!	298
Understanding the Two Key SAP Support Staff Personalities	298
Communication—Keeping It Regular and Meaningful	300
Salary Requirements—Just the Beginning	302
The “Most Important Thing”	302
Performance and Other Incentive Bonuses	303
Other Compensation Alternatives in the Real World	304
Tools and Techniques	307
Summary	307
9 Critical SAP Training Requirements	309
Introduction to SAP Training	309
Who Needs Training?	310
Timing Is Everything	313
Training and the Role of the SAP System Landscape	314
Leveraging the Technical Sandbox	316
Development and the “Business Sandbox”	317
Wringing All the Value out of a Training System	317
Training to Support Your Unique SAP Solution Stack	318
Approaches to SAP Training	319
Formal Classes and Courseware	320
Onsite Training Workshops	323
Creating and Delivering Custom Training Curriculum	323
SAP Knowledge Warehouse	325
Using the Enterprise Portal for Knowledge Management	326
Other Computer-Based and Online Training	328
SAP TechEd and Similar Venues	328
Creating “Cookbooks” from Product Documentation and User Manuals	330
Feedback Loops—Improving the Value of Training	333
Functional and Development Consultants	333
SAP Technical Consultants	333

Other SAP Infrastructure Roles.....	334
SAP Operations/Help Desk	335
Additional SAP Support Specialists.....	335
Certification Programs.....	336
Tools and Techniques	337
Summary	337
10 Developing the SAP Data Center	339
Introducing the SAP Data Center	339
The SAP Data Center “Big Picture”	342
First Things First—Standardization	342
Data Center Physical Requirements	343
Power Requirements	344
Power Oversight in the Real World	347
Cooling and Other Environmental Controls	349
Rack Planning for Data Center Resources	350
Rack Layout and Design Considerations	350
Optimizing Rack Real Estate.....	352
Rack Mounting and Related Best Practices	353
Cabling and Cable Management	354
Network Infrastructure for SAP	355
Network Fault Tolerance	356
Central Systems and Optimal Network Configuration	360
Network Server Preparation	362
Optimum Server Configuration Best Practices for SAP	362
Operating System Best Practices for SAP	364
SAP Server Configurations in the Real World	369
General Storage Considerations	369
Special Considerations for Storage Area Networks	370
Deploying SAP on a SAN—General Best Practices and Observations	371
The Latest in SAN Technology—Leveraging Storage	
Virtualization	373
Storage Virtualization Best Practices	375
On the Road to Implementation	376
Planning for SAP Data Center Operations	376
Special Considerations When Deploying the Technical Sandbox	378
Special Considerations When Deploying the Development System	379
Tools and Techniques	379
Summary	380

11	Performing mySAP.com Component Installations	381
	Implementation Methodologies for SAP Installations	381
	ValueSAP and the Global ASAP Roadmap	382
	SAP Solution Manager for Implementation	383
	Rounding Out Your SAP System Landscape	384
	Speeding Up Your OS Installation Process	385
	Preparing for an SAP/Database Installation	385
	System Landscape Implementation Manager—SAPinst	386
	Leveraging Installation Documentation, Tools, and Approaches	387
	Do Your Homework	387
	SAP Master Guides	388
	SAP InstGuides—Traditional Installation Guides	389
	Addressing Pre-Installation Tasks	390
	Post-Installation Tasks	390
	Custom Checklists and Recipes	391
	Smart Implementations and the SAP Configuration Assistant	393
	Using SAPinst for Unattended Installations	395
	Installing Your mySAP-Enabling Foundation—NetWeaver	397
	Web Application Server 6.x	397
	The SAP Exchange Infrastructure 2.1	398
	Enterprise Information Portal 5.0	400
	Installation Details for ITS 6.20	403
	Installation Details for mySAP Components	404
	Installing R/3 Enterprise	405
	Primary mySAP BI Installation Considerations	406
	Primary mySAP CRM Installation Considerations	407
	Installing the SAP Knowledge Warehouse 6.0	408
	Installing mySAP PLM	408
	Primary mySAP SCM Installation Considerations	410
	Installing mySAP SEM	413
	Primary mySAP SRM Installation Considerations	414
	Addressing General mySAP Post-Installation Tasks	415
	SAP Client and Transport Strategy	415
	Security, Authorizations, and Trust Relationship Management	416
	Printing and Faxing	417
	Backup/Restore Considerations	419
	Archive Considerations in the Real World	420
	Tools and Techniques	422
	Summary	423
12	Rounding Out Support for SAP	425
	Identifying and Staffing the Remaining TSO Roles	425
	Where Exactly Are the Holes in Your TSO?	426
	How Big Is “Big Enough?” in the Real World	427

Approaches to “Filling in the Final Holes”	429
Keys to Expanding the SAP TSO	430
Transferring in New Internal Folks	432
Leveraging Third-Party Consultants Again	435
Consulting in the Real World—Fewer Is Better	436
Consulting in the Real World—Bigger Than Me	437
Let’s Give a Big Warm Welcome to the “Specialists”!	438
SAP Component Basis Specialists	438
SAP Component Functional Specialists and Programmers	440
Enterprise Application Integration Specialists	440
Performance Specialists	441
SAP Security, Access, and Authorizations Specialists	443
High Availability and Disaster Recovery Specialists	445
Testing, Data Conversion, and Other Specialists	446
The Most Overlooked Critical SAP Support Function	447
SAP Operations Expectations	447
SAP Monitoring—Ensuring Highly Available Systems	448
Taking Advantage of SAP Operations	450
Change Control—Where the Rubber Meets the Road	451
The Role of the SAP Help Desk	451
Staffing the SAP Help Desk	453
Common SAP Help Desk Questions in the Real World	454
A Simple Roadmap to SAP Help Desk Preparation	455
Managing End-User Perceptions	456
Putting It All Together in the Real World	456
A Sample “Small Business” SAP Support Organization	456
A Sample “Medium Business” SAP Support Organization	458
A Typical Fortune 50 Global SAP TSO	459
Tools and Techniques	462
Summary	462

Part III Moving Into SAP Functional Development

13 Gaining Control of Change Control	465
An Overview of Change Management	465
Change Management Mentality	465
The Real Reason for Managing Change—Stakeholders	468
Change Management Best Practices and Approaches	469
The Core Philosophy Behind Change Control—Testing	471
How Documentation Impacts Change Management	472
Minimizing Change Management with Standards	473
The Release Strategy Approach to Making Changes	474
Clear Two-Way Communications	477
Communicating Changes in the Real World	478

Another Way to Implement Change—Workflow	481
Change Control Tool Sets and Approaches	482
Feedback—Improving Change Management Incrementally.....	484
Change Control Affects Everything	485
Change Control and the SAP System Landscape	485
Change Control and the Phases of SAP Implementation.....	485
Change Control and the SAP Solution Stack	488
The Hardware, OS, and Database Layers.....	488
SAP Application Layer—Transport Strategies and More	491
Upgrades—Realizing the Benefits of Change Control	492
How to Organize and Plan for Change in the Real World.....	493
Change Management Review Board	495
Manager of Change Management	495
Senior SAP Change Management Analyst	496
Change Management Lessons Learned in the Real World.....	497
Change Control “Worst Practices”	500
Tools and Techniques	502
Summary	502
14 SAP Systems and Operations Management	503
Back to SAP Operations	503
What Is the SAP Operations Manual?.....	504
Detailing Your “Current State”.....	505
Documenting Daily Operations and Installation Procedures	507
Documenting Other Regularly Scheduled Procedures	508
“How To” Documentation	509
Documentation Best Practices in the Real World	509
Systems Management Techniques for SAP	511
Leveraging CCMS for Manual Processes/Checklists	512
Automating CCMS Data Collection Processes	513
CCMS, Transactional Monitors, and CEN	515
The SAP Solution Manager	517
Other Tools and Utilities	517
Piloting a Systems Management Application for SAP	518
Defining Requirements	519
Systems Management Challenges in the Real World	520
Leveraging In-House Applications	521
Creating the Short List	522
Evaluating Enterprise Systems Management Applications	523
BMC Patrol	524
CA Unicenter Application Management for R/3	525
HP OpenView VantagePoint for Windows	526
NetIQ AppManager for R/3	527
Enterprise Management Applications—Lessons Learned	528

Additional SAP Management Tools and Approaches	529
Hardware Management Tools and Utilities.....	529
OS Management Utilities.....	531
Database Management Tools	532
More Value from the SAP Solution Manager.....	532
SAP's Solution Services.....	534
SAP Note Assistant.....	534
Specialty Utilities	536
Tools and Techniques	540
Summary	541
15 Functional, Integration, and Regression Testing	543
Testing SAP Business Processes.....	543
Introduction to CATT and eCATT.....	544
Three Types of Business Process Testing	546
When to Execute Business Process Testing	547
The Critical Nature of Functional Testing	549
The Real Value in Integration Testing	550
Regression Testing	550
Functional Versus Stress Testing	551
How to Approach Business Process Testing	551
Third-Party Tools and Other Resources	553
SAP eCATT Differentiators	554
mySAP.com Landscape Considerations	556
Additional People Considerations	557
Process Constraints and Issues	558
Other Areas of Impact	559
Executing Business-Process-Specific Testing	560
The Test Workbench	561
Data to Track During Test Execution	562
Post-Execution Tasks	562
Compressing the Testing Phase in the Real World	563
Using Testing to Support SLAs in the Real World	563
The Weakest Link	564
Tools and Techniques	564
Summary	565
Part IV Preparing the Production Environment	
16 Systems and Stress Testing	569
Introduction—Preparing for Production Stress Testing.....	569
The Goals of Stress Testing	571
Performance Baselines and Success Criteria	572

System-Level Stress Testing and Pre-Tuning	574
Server Hardware and OS Testing Tools and Best Practices	575
Disk Subsystem and Database Testing	576
Network Infrastructure Testing in the Real World	577
mySAP.com Component Layer Testing	579
Key SAP Stress-Testing Considerations	580
Creating an SAP Stress Test Project Plan	580
Analyzing Online Users and Batch Processes	581
It's All About the Data!	581
Updating Your Project Plan	582
Real Versus Virtual Users	583
SAP-Aware Versus Freeware and Inexpensive Testing Tools	583
Developing Business Process Scripts	585
How to Test mySAP Components	585
SAP Standard Application Benchmarks	586
Where eCATT Fits In	588
SE38—One Answer to Cross-Application Stress Testing	588
Business Information Warehouse and SEM	588
Enterprise Portal and SRM/Enterprise Buyer Pro	590
Other mySAP Components and Considerations	590
Script Development and Preparation in the Real World	591
Core Script Development	592
Stress Testing Client Infrastructure	596
Creating Administrative and Other Utility Scripts	599
Login and Establishing a Virtual User Session	600
Ramping Up Users and Processes	602
Collecting Statistics	603
Logging Out—Gracefully Ending Your Test Session	605
Additional Scripting Tips and Tricks	606
Final Preparations Before "Test Week" Commences	609
Stress Test Execution During "Test Week"	610
Leveraging Your Testing Tools	610
Monitoring the Stress Test via SAP Transaction Codes	611
Operating System Utilities	611
Using Test Output For Continuous Improvement	612
Additional Stress-Testing Goals	613
Playing "What If"	614
Verify System Redundancy and Failover Perform as Expected	614
Ramp Up to Excessive Loads	614
Extracting the Last Drop of Value out of Testing	615
Verify That Backup/Restore Meets Requirements	616
Ensure That Your Disaster Recovery Plan Is Effective	617

Lessons Learned in the Real World	617
Tools and Techniques	619
Summary	619
17 Preparing for SAP Go-Live	621
Overview—Preparing for Cutover	621
The Cutover Plan	621
Preparing for Technical Go-Live	624
SAP GoingLive Check and Other Review Processes	625
SAPGUI Rollout Mechanism	626
Setting Up Batch Housekeeping Jobs	626
Final System Updates and Review	627
Locking Down the System	628
Preparing for the First Change Management Package	628
Final Administrative Technical Details	629
Backup/Restore Processes	629
Output Management	630
Tweaking Your Systems Management Approaches	631
More Details—Managing the SAP Enterprise	631
Determining Realistic SLAs	632
Tracking System Performance	634
The Changing Role of the SAP TSO	635
Updating Responsibilities and Roles	636
New Roles, New Personality Types	636
Preparing SAP Operations and the Help Desk	637
Final Preparations	638
Updating “As-Is” Documentation	638
Updating Procedural Documentation	639
Addressing Future Service and Support	640
Support Agreements	641
Leveraging Joint Escalation Centers	641
The First Week of Go-Live	644
Monitoring During Go-Live Week in the Real World	644
Planning for Feedback and Continuous Improvement	646
Post-Implementation Evaluation	648
Proper Congratulations	648
Tools and Techniques	649
Summary	649
Index	651

Foreword: Planning an SAP Deployment

Implementing SAP is all about embracing change—letting go of an old stovepipe way of doing things in favor of an enterprise-wide and holistic approach.

What A. J. Cronin said many years ago still holds true today:

“...if we have faith, a door will open for us, not perhaps one that we ourselves would ever have thought of, but one that will ultimately prove good for us.”

So it is with an SAP implementation. At the end of the project, an integrated enterprise system will be in place that allows a company to realize improved communications goals, data sharing, and ultimately—increased return on information (an old SAP adage that can be validated by thousands of customers today).

But the really amazing thing that happens, the thing that is often overlooked in both computer rooms and board rooms, is that the entire company turns a corner. What has happened in leaving behind the old way of doing things—and all the baggage that entails—is that the organization has positioned itself for growth. Indeed, this growth is painful in the beginning, for it involves learning and applying new approaches to age-old business dilemmas. But eventually, the classic “one step back” turns into many steps forward, and the company grows and evolves. And yes, it even manages to stay a few steps ahead of its non-enterprise-integrated and Web-enabled competition.

In the next few hundred pages, we will look at precisely how these changes are brought about by implementing the mySAP Business Suite. I will provide an in-depth look at how these solutions are planned, sized, implemented, staffed, tested, and to some extent supported and managed. My real-world experiences reflect the challenges embraced and conquered by many SAP enterprise customers as they themselves managed to change and essentially reinvent their companies.

What I do differently from other “SAP planning guides” is to approach all this change from an SAP infrastructure or SAP Basis/Web Application Server perspective. And I do so in a manner that not only maps to SAP’s proven ASAP methodology and its successors, but also identifies (and actually assists my readers in taking advantage of) best practices and approaches gleaned from thousands of implementations and upgrades, ranging from the latest NetWeaver-enabled mySAP solutions to core R/3 implementations. So read on, and position yourself NOT to make the same mistakes so many of us have already made, mistakes from which we have all learned so much!

About the Author

George W. Anderson is a six-year veteran of numerous successful end-to-end SAP implementations and hundreds of SAP design, upgrade, and testing engagements. An SAP Basis consultant by trade, he is currently employed by the Hewlett-Packard Company, responsible for leading a team of SAP Basis and other systems infrastructure folks throughout North America. In this regard, he has personally developed new SAP opportunities and actively participated in the consulting on the resulting projects.

George has been engaged in supporting mission-critical enterprises for 16 years. He started his career as a mainframe operator and JCL/COBOL developer while serving in the United States Marine Corps. After earning his MBA and leaving the Corps, he moved into supporting Banyan and Novell networks. He eventually followed the rise of Microsoft back into the enterprise space, designing and configuring NT Server-based solutions in complex heterogeneous environments.

After holding several positions of increasing responsibility with a national systems integrator, and eventually leading a team of more than 50 enterprise and database consultants, George joined Compaq Computer Corporation's fledgling Enterprise Consulting Services group in 1997.

It was here that George got his first real taste of SAP consulting, successfully assisting a major petroleum company with a number of proof-of-concept SAP R/3 stress-testing engagements. The goal was ambitious indeed for the time—to prove that Microsoft SQL Server 6.5 could actually scale to handle a mix of 600 SD, PP, FI, and MM users. This experience, coupled with a number of engagements in what was becoming known as “total cost of ownership” analysis, cemented George’s desire to pursue a career in SAP consulting. And in doing so, his focus was sharpened on designing highly capable, yet relatively low-cost SAP solutions for his customers.

In 1998, George moved into a senior engineering and consulting role within Compaq’s SAP Competency Center, and was soon adept at analyzing and mapping large company business requirements into SAP R/3 and BW technical solutions. He developed what may be considered the first real sizing methodology for SAP BW at this time, and later assisted in doing the same with APO and B2B (which eventually morphed into what today is known as Enterprise Buyer Pro).

George foresaw the upcoming trend with regard to Internet-enabling and extending SAP into the larger eBusiness enterprises. As a result, George added “.com” and “Internet Transaction Server” credentials to his repertoire. Soon, he was designing and building ITS on Microsoft IIS solutions in support of Employee Self-Services, and other burgeoning SAP Internet-related endeavors.

In late 1999, George returned to the "field," armed with a recent Project Management Certification as a "PMI PMP," followed by one of the first Microsoft Windows 2000 MCSE certifications.

Soon, with experience and certifications in Microsoft, Compaq, and SAP technologies, George found himself immersed in cutting-edge assignments involving Business Information Warehouse design and testing, a Business-to-Business Procurement pilot, SAP Workplace implementations, and other "mySAP" solutions projects.

In recent years, changes in technology saw George designing and implementing Storage Area Networks in SAP environments, implementing W2K and Unix clusters, and assisting large customer SAP R/3 proof-of-concept and other benchmarking exercises. Today, his work continues to provide challenge, and George is looking forward to providing many more years of SAP consulting and back-room engineering to enterprise SAP customers of *the new HP*.

Acknowledgments

First and foremost, I would like to thank our Lord and Savior Jesus Christ, for always being there for me even when I sometimes forget, for blessing me with my wonderful family, my awesome job, the time to write this book, and the absolute peace and joy that only come through knowing Him. I also want to acknowledge the support and faith that my wife Michelle has shown in me over the years, and simply say “thank you” for putting up with me when I’m happy and when I’m grumpy, when I’m away on business or at home working. I love you today, and will love you more tomorrow, Michelle—you are my most precious gift from God. And to my kids, Phillip and Ashley, a special thanks for sharing your “Papa” with his computer over the last few months. You are truly just awesome kids, and I love you.

I would also like to acknowledge the help and assistance my Dad provided over the last five months, reviewing and tweaking the contents of this book to help ensure it was both readable and understandable by “the rest of us.” Thank you, Dad. I am so fortunate to have a dad with your talents and gifts, a father who is there whenever I need him. And to think you only get better at this Dad thing every year!

And to my Mom, I always knew I was loved completely and without condition by you growing up, and this fact shaped me in so many ways, distilling in me a confidence that I could do and be anything. Generations beyond you will reap this legacy. There is nothing I would rather pass on to my own children than the love you gave me—status or wealth or possessions could never compare to that. I love you.

And to my friends, colleagues, and customers across the states, thank you for teaching me and guiding me and simply cutting me slack through all of the learning and growing years. I am honored to be counted among your friends—in your own ways, you have helped shape me, provided me with insight, shared your lives and careers with me, prayed for me, and helped make me into who I am today. Thank you, to my friends and colleagues John Dobbins, Fazil Osman, Rick Nye, Brad Martin, Eric Tapp, Thomas Miles, Tim Rhodes, Robert Ferguson, Karen Myers, Kathy Flanagan, David Bennett, John Murdock, Matthew Selheimer, Don Dean, Paul Chisari, Len Bailey, David Braithwaite, Ren Ganner, Roy Jackson, Kathy Richardson, and Robert Liebert. Thank you, my favorite customers in Austin, Texas; Cleveland, Ohio; Houston, Texas; and Cranston, Rhode Island. Thank you, my professional circle of SAP influences, including Naheem Hashmi, Jon Reed, everyone in SAPVirtual and the GSSC, HP’s SAP Solutions Center, Compaq’s original ECS organization, the technology consultants and trainers at SAP, and the folks at *SAP Professional Journal*. And thank you, Houston’s First Church of the Nazarene and all of the staff and people there I count as friends, who have helped me grow in Him over the last three years—especially Boyd and Myra, David and Beverly, Jack and Mary, Jerry and Gloria, Elizabeth, Robyn, Jeffrey, Sean, Chad, Brian, Deborah Keith, and the entire HFC youth staff and kids!

Tell Us What You Think!

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

As an associate publisher for Sams, I welcome your comments. You can email or write me directly to let me know what you did or didn't like about this book—as well as what we can do to make our books better.

Please note that I cannot help you with technical problems related to the *topic* of this book. We do have a User Services group, however, where I will forward specific technical questions related to the book.

When you write, please be sure to include this book's title and author as well as your name, email address, and phone number. I will carefully review your comments and share them with the author and editors who worked on the book.

Email: feedback@sampsUBLISHING.com

Mail: Michael Stephens
Sams Publishing
800 West 96th Street
Indianapolis, IN 46240 USA

For more information about this book or another Sams title, visit our Web site at www.sampsUBLISHING.com. Type the ISBN (excluding hyphens) or the title of a book in the Search field to find the page you're looking for.

PART I

Preparing for Your mySAP Business Suite Implementation

IN THIS PART

- 1 Introduction to SAP Infrastructure Planning
- 2 What an SAP Project Looks Like
- 3 SAP Solution Vision
- 4 Designing the SAP Infrastructure Technical Support Organization

1

Introduction to SAP Implementation Planning

Welcome to SAP Planning!

The core of the material you are about to read stems from seven years of SAP implementation experience across more than a hundred mid-size and global SAP customers. My goal is to bridge the gap between making the decision to purchase a mySAP software solution, and actually pulling the lever for "Go-Live," which, in essence, makes the new system available to your end users on the Production system. It is my hope that you will use this text as both a reference tool and an informed guide, steering clear of the hazards common to so many SAP implementations.

The rapid changes in information technology hardware, software, and in particular SAP AG's umbrella of ever-growing solutions, have had a profound influence on the way companies today access and manage their data. The role SAP has played in this regard, especially in the last few years, has been pivotal to say the least. When faced with dot-com's to the left of them, and hot best-of-breed applications to the right, SAP sought to embrace the best of all worlds and evolved so as to continue to meet its customers' and stakeholders' needs.

In wishing to share my own experiences in this regard, I asked myself "What is the number one reason for putting together this book?" The simple answer was: *one-stop shopping for "SAP Planning and Best Practices."*

IN THIS CHAPTER

- Welcome to SAP Planning!
- My Audience and Approach
- Addressing the Real Challenges of SAP Implementations
- What Exactly Is SAP?
- SAP Infrastructure Planning—What It's All About
- Special Materials Provided by the Book

I have put most everything a company needs to know or address in terms of planning/organizing for an SAP implementation from a Basis and Operations perspective into one book. Without this book, you would have to hunt through a hodgepodge of white papers, Web content, miscellaneous documents and articles, and a chapter here and there in the few really good texts that exist today. Instead of starting from ground zero, as so many SAP customers do, you will be able to put together custom project plans, implementation schedules, management justification, and more in just a few days. This is the book my colleagues and I were looking for five years ago, but no one had created it.

In addition, my experience is real-world and my perspective is fairly unbiased. That is, I am relatively unencumbered by my own personal database preferences, operating-system likes and dislikes, competing applications, and so on. As one of SAP's premier partner organizations, my team at HP claims a huge number of successful SAP installations to our credit, including some of the largest Microsoft and Oracle-based SAP installations in existence. So I understand the real gut issue that my readers face: The decision has been made to go with a mySAP solution, knowing full well that the risk on the business side is so high that there is little room for risk in the technical implementation. In different chapters of the book, then, I am quick to address challenges relevant to the following:

- Organizational changes that accompany an SAP implementation will drive sweeping changes across much of the company.
- Meeting the project's ROI goals in a timely fashion will impact everything from planning to developing the solution, to testing it, to implementation, and more.
- The Information Technology group (IT) will tend to think of this as an IT project, initially unaware of the integrated nature of SAP and how it will necessitate a tight partnership between the business and IT.
- At the end of the day, the IT department will be faced with implementing a technology solution before the scope of the business solution has crystallized for everyone, and despite the fact that the mySAP solution itself is unfamiliar.

Thus, IT will benefit from all of the help they can get out of people like myself who have already made the journey, know the issues, and have dealt successfully with an SAP project's uncertainties. This book will go a long way toward providing the processes, insights, and wisdom that will enable a company to get the job done right, and on time, the first time.

As SAP Basis consultants and SAP Infrastructure Project Managers, my team established years ago that a solution-agnostic approach to SAP consulting kept all of us working. We let the marketing, technology, and engineering guys do their thing,

while we focused our own efforts on implementation and otherwise taking care of our customers. This meant configuring our customers' various new, redeployed, or best-of-breed hardware and software components into a *solution*, regardless of the different technology vendors and partners involved. Indeed, we considered ourselves actually quite fortunate when we got involved early enough in a project to allow us to have a hand in the project's architecture design and selection. In light of this, I have worked with all of the major hardware, operating system, and database vendors upon which an SAP solution relies, growing and changing with the times as hot technologies replaced what so quickly became legacy.

Finally, I understand that the only reason a company would implement or upgrade SAP in the first place is for business reasons—to increase your competitiveness, reduce costs, make information more widely available across your company, enable better service to customers, improve decision-making capabilities, enhance resource planning, and ultimately improve the execution of so many different business processes. In summation, the technology required to support SAP is simply a means to an end, and not the end itself.

My Audience and Approach

So, you're ready to plunge into the world of SAP! Or, maybe you're in too deep already, perhaps even past that critical point of Go-Live, and need to step back and review where you are and how you got there. Perhaps you're soon going to be involved in a mySAP implementation, or are considering a support or management role at an existing SAP site. On the other hand, you may just be curious about what SAP infrastructure and implementation best practices and approaches are all about. In any case, you have come to the right place.

I suspect that many readers will use this text as a baseline of sorts, comparing their own SAP plans and implementations to what I have provided, looking for new ideas, or alternatives for approaching problems that are common to all system implementations. In lieu of this, I believe my readers fall into a number of categories, as my real-world subject matter appeals to many, including:

- Executive C-level managers and directors tasked with implementing or maintaining SAP environments
- SAP project managers and various SAP team leaders tasked with discrete subprojects related to implementing, supporting, testing, tuning, or training
- SAP Basis and Web Application Server consultants, engineers, and technicians, asked to size, configure, and implement mySAP solutions
- Database administrators (DBAs) and Storage Area Network (SAN) consultants with a need to maintain their piece of the SAP enterprise pie, or simply expand their knowledge

- Traditional data center operations and infrastructure management folks asked to step up and assist in developing or maintaining an SAP IT shop
- Network administrators, systems administrators, data center power/utility technicians, and others with similar roles supporting the very groundwork upon which the SAP solution lies
- Others internal to (or seeking employment with) an organization, interested in learning the process a company should follow in selecting, designing, and deploying SAP
- Technical individuals who are new to (or want to be a part of) the world of SAP—individuals who may be supporting similar enterprise applications or mission-critical environments (mainframes/mid-frames, more) and who want to make a career move into learning and supporting SAP
- Finally, non-technical business managers/supervisors who are soon to be thrust into an SAP project or environment

The book has been crafted along the lines of a project plan. In fact, on the CD enclosed with this book, I have actually included a comprehensive sample project plan, along with PowerPoint presentations, Excel sizing worksheets, various Word documents, scripts from a number of tools, each figure in the book, and much, much more. All of this is designed to get you quickly up to speed, while guiding you through your own SAP infrastructure implementation.

The core value that I provide to you is the chance to benefit from the work of others—namely, my own work and that of my team. With thousands of SAP installations out in the world today, there is little value in reinventing the wheel. Frankly, most everything you need or want in regards to a mySAP implementation has already been done, and done well, by someone else inside another company. So why not read on, and take full advantage of that?

Regardless of whether you are implementing or updating your mySAP.com enterprise, and whether it consists of SAP R/3, Business Information Warehouse (BW), Advanced Planner and Optimizer (APO), Product Lifecycle Management (PLM), Customer Relationship Management (CRM), Enterprise Buyer Pro (EBP), Enterprise Portal, or any other number of SAP enterprise components, there are certain tasks that must be planned for and executed across the board. And if you're interested in minimizing costs and managing your critical path to a successful outcome, these tasks must occur in a certain logical order or sequence. With all of this in mind, it seemed rather obvious that a "project plan" approach made the most sense for the book.

For new implementations, I suggest that you read the book sequentially, from the first to the last chapter. If you find yourself in the middle of a project, though, feel

free to jump to the chapters that best fit your project or timeline status. Of course, in doing so you might well “skip” over knowledge that could very well prove useful, too. I suggest quickly reviewing the Table of Contents, therefore, to determine if it makes sense in your particular case to go back and review any passed-over content.

What Exactly Are “Best Practices and Approaches”?

For every thousand implementations, there are a thousand ways to implement SAP. In the course of consulting, however, I have determined that there tends to be one or two “best” or “preferred” methods of doing a particular task, or addressing a particular problem. It is these nuggets of insight and knowledge that I hope to pass on to you, my reader, within the larger scope of covering an SAP infrastructure implementation end-to-end. Most of the concepts, practices, and approaches outlined in this book are the result of years of experience designing, deploying, and supporting SAP implementations enabled by Compaq’s, Digital Equipment Corp’s, and Hewlett Packard’s technology platforms. Like SAP AG, we too have endured many changes over the last few years, and have grown both stronger and wiser in doing so. We boast some of the largest implementations in the world, and some of the most complex in terms of enterprise integration requirements. And we boast some of the most cost-effective SAP solutions as well, pushing the envelope when it comes to embracing new computing paradigms and solutions approaches.

Given the many possibilities I just described, in my experience best practices and approaches related to SAP projects can be grouped into four general areas:

- People
- Processes
- Technology
- Return on Investment (ROI)

I will do my best to ensure that all four of these areas are covered in each chapter, as appropriate, along with relevant best practices and approaches. It is my intent to build an understanding of the problems and pitfalls you will encounter, and how these might best be rectified or avoided altogether as we march down the roadmap of an SAP implementation. As such, I view this book as simply an extension of my own SAP consulting work, a conglomeration of insight and experience melded together in one place. You are now my customer, and I am your (very inexpensive!) consultant. Given that the efficient and proper use of external consultants is one of many keys to a successful SAP implementation, you’re well on your way to success by just leveraging this book, the foundation of my “customer deliverable” to you!

Addressing the Real Challenges of SAP Implementations

In a world filled with books on SAP (those of us who work with SAP for a living like to hear it pronounced “ess-aye-pea,” by the way), this book is unique. In my review of numerous “how to” and other SAP planning guides over the years, I continually noticed how little attention was given to addressing the *real* challenges related to deploying an SAP solution. For example, little attention was ever given to:

- How to structure an SAP technical support organization
- How to encourage apples-to-apples SAP sizing exercises, and then evaluate each vendor’s solution approach on a level playing field
- How to determine realistic high-availability and performance requirements
- How to plan for and develop an SAP Data Center
- What to include in an SAP Operations Manual
- How to plan for and execute both functional and load/stress tests
- How to really leverage your SAP system landscape for training and testing
- What training is really required across user and technical boundaries, and when it should be delivered
- What role the help desk and SAP Operations teams must be both staffed for and prepared to play
- The infrastructure or basis tasks that need to be addressed, and when, to actually make it to Go-Live
- How to build “buy in” with the business folks—the owners and end users of the system.

I address all of these, and much more, from an SAP infrastructure perspective. And by following the project-plan approach outlined earlier, I promote a timeline that coincides nicely with SAP’s ASAP and newer roadmaps, which enables the functional development and related infrastructure deployment requirements to be mapped out in lockstep. Finally, to assist my readers with jumping into action (or simply avoiding re-creating the wheel), I include various tools and techniques that may be leveraged throughout your own SAP implementation. These are identified and discussed at the end of each chapter, and included on the CD that ships with this book.

What Is Covered

Each chapter covers the tasks and subsequent problems and solution approaches typically encountered when planning for, testing, or implementing SAP. My focus is

centered on all of the SAP infrastructure requirements necessary to provide the groundwork for the actual software functional configuration and deployment. Some people label this groundwork *SAP Basis*, which is SAP's own word for the foundation upon which the SAP solution resides. I prefer the terms "SAP Infrastructure" or "SAP Solution Stack," though, as Basis limits us in so many ways to a smaller set of discrete tasks or scope of responsibility. Besides, with the term "Basis" making way for newer terms associated with SAP's Web Application Server (and the newest 6.30 technology foundation for SAP components), my preferred terms allow us to cover both the older and newest worlds of SAP.

- ▶ To learn more about what the SAP Solution Stack entails at a high level, see "In the Beginning," p. 23 in Chapter 2. For detailed SAP Solution Stack information, refer to "A Closer Look at the SAP Solutions Stack," p. 51 also in Chapter 2.

What Is Not Covered

Although the functional programming, configuration, and work required to make SAP actually *useful* after it is installed is paramount to the overall success of any SAP implementation, I do not go into this detail here. Rather, I leave the topic of SAP's programming language, ABAP (Advanced Business Application Programming), and its more recently supported development option, Java, to the many books, articles, and other documents out there aimed squarely at this kind of activity. When appropriate, I discuss functional development, testing, and other related tasks as they impact our discussions from an SAP infrastructure perspective, however.

In addition, I assume that you have already selected SAP (or it has been selected for you!) as your enterprise solution package of choice. Certainly, there are a number of choices in the enterprise solutions arena—including products from PeopleSoft, Oracle, JD Edwards, Microsoft, Baan, and more. However, SAP continues to command the lion's share of enterprise implementations, even recently surpassing a number of "best of breed" specialty applications in terms of popularity. Some of these will be discussed later, but if you are looking for a book that will help you determine *which* enterprise application is right for you, you need to keep looking. Of course, you could decide to go with the market leader, a company positioned to remain successful (even in these times of tightened IT budgets), and end your search right here.

What Exactly Is SAP?

SAP AG (AG is the German equivalent of the term "incorporated") refers to the name of one of the largest software companies in the world, often referred to simply as SAP. The company, consisting originally of ex-IBM folks with a vision of creating an integrated enterprise software solution, is based out of Germany and has been in business since 1972.

SAP is also the tag given generically to software created and marketed by SAP AG. Their most popular application package by far is called SAP R/3, which competes in the *Collaborative Business Solutions* category of software, designed to facilitate business operations such as: order entry, materials and warehouse management, logistics, sales and distribution, financial and asset accounting, human resource management, and more.

Other applications created and marketed by SAP have become quite popular as well. We will cover many of these in detail later, but suffice it to say that SAP has offerings in data warehousing (mySAP Business Intelligence, which includes Business Information Warehouse, or SAP BW), supply chain management (Advanced Planner and Optimizer, or SAP APO), customer relationship management (mySAP CRM), product lifecycle management (mySAP PLM), business-to-business procurement (mySAP Supplier Relationship Management, which consists of Enterprise Buyer Pro, or EBP, and predecessors BBP and B2B), and much more. Today, it can be safely said that if there is any system or software need in the enterprise, SAP probably offers a product to fill that need. This is a much different scenario than three or four years ago, when SAP was a synonym for simply R/3.

How Do mySAP.com and mySAP Fit into the Big Picture?

Way back in the heady days of 1999 or so, when everything was “dot-com this” and “dot-com that,” SAP was already years ahead of the game. R/3 had been Internet-enabled since the introduction of version 3.1G, and the timing was right for SAP AG to introduce a new e-enabled vision of their growing product line. Out of this vision came “mySAP.com,” an umbrella term used to refer to the entire breadth and depth of SAP’s e-business solutions and products. Recently renamed to mySAP Business Suite, SAP defines mySAP.com as:

“A family of software and services that empowers customers, partners, and employees to collaborate successfully—anywhere, anytime.”

Thus, we can combine everything that SAP sells today under the term mySAP.com, and feel pretty comfortable about it, as illustrated in Figure 1.1.

However, the situation becomes a bit more complicated as we introduce the term *mySAP*. Without the .com suffix, mySAP refers to SAP *Solutions*. Thus, we wind up with various “mini-umbrellas” underneath mySAP.com, such as solutions like mySAP Business Intelligence (mySAP BI) or mySAP Supply Chain Management (mySAP SCM). Underneath these mini-umbrellas lie the actual software products, however, which SAP labels generically as *components*. For example, underneath the umbrella of mySAP BI reside SAP Business Information Warehouse, SAP Knowledge Management, and SAP Strategic Enterprise Management.

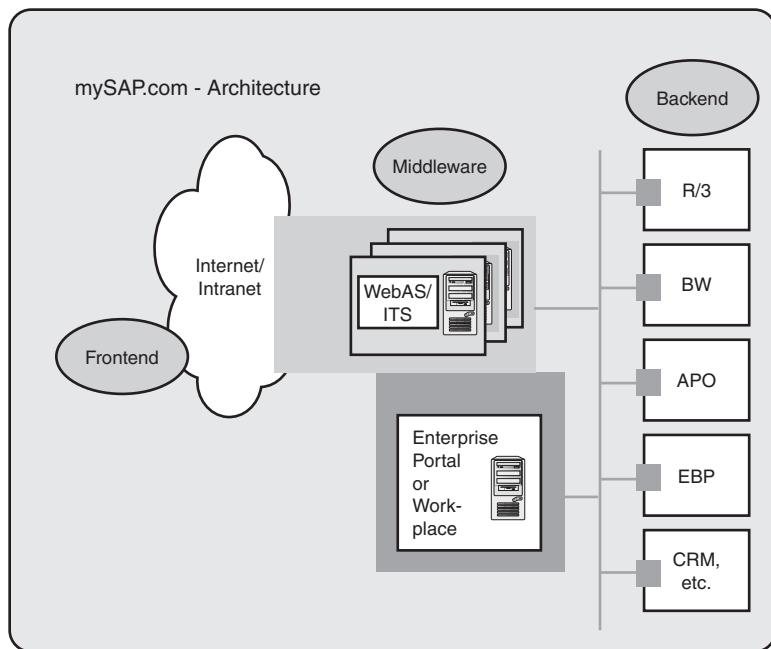


FIGURE 1.1 The SAP family of software and services.

Many other mySAP solutions exist as well, as illustrated in Figure 1.2. For example, cross-industry solutions include mySAP Marketplace, mySAP HR, mySAP Mobile Business, mySAP Financials, and quite a few others. And there are SAP's Industry Solutions to consider, too. These industry verticals are targeted at specific lines of business, and include solutions such as: mySAP Automotive, mySAP Healthcare, mySAP Oil & Gas, mySAP Retail, and over a dozen more.

The underlying software components of any given solution are neatly prefaced with the simple term "SAP." This is appealing to those of us who have been working with these products since day one, when they were introduced under the old umbrella of New Dimensions Products, because that's how they were labeled back then as well. So, R/3 has always been and continues to be called SAP R/3. Similarly, Advanced Planner and Optimizer (under the mini-umbrella of mySAP SCM) is still simply called SAP APO.

The term SAP, then, typically refers to the technical components used to implement a mySAP solution, under the overall umbrella of mySAP.com. For the remainder of this book, however, I will continue to use the term "SAP" to refer generically to any SAP product or component, or to the company itself.

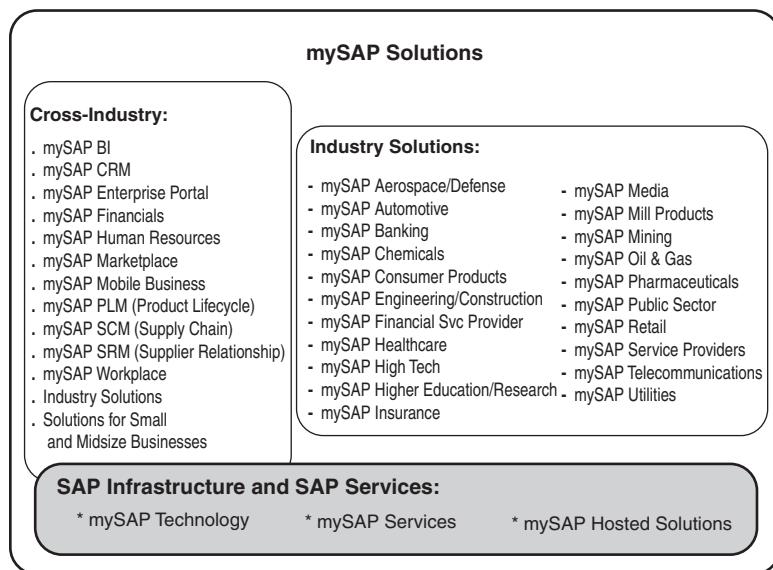


FIGURE 1.2 SAP’s Solutions represent both cross-industry capabilities and targeted point solutions for industry verticals.

SAP Infrastructure Planning—What It’s All About

As indicated earlier, SAP Infrastructure or SAP Basis refers to both the technical foundation and the actual SAP installation upon which all development activity and productive operations rely. Note that I will also use the term *SAP Solution Stack* interchangeably with SAP Infrastructure and SAP Basis.

The solution stack references the “layers” of infrastructure and technology that sit one on top of the other in support of an SAP solution, like the layers in a three-layer cake. Generally speaking, this might include the following:

- Physical space, like a computer room or other data center facility
- Power, cooling, and other utility-based infrastructure layers
- Physical mounting and racking layer
- Server and disk subsystem hardware layer
- Firmware layers associated with specific hardware
- Operating System (OS) layer
- OS drivers, service packs, updates, patches/fixes

- Database layer
- Database drivers, service packs, updates, patches/fixes
- SAP application layer, which in and of itself consists of multiple layers
- Internet-enabling layer
- SAP accessibility layer, including desktops, laptops, and other devices used to access a mySAP solution

Of course, each of these layers can be further broken down into more detailed layers. For example, server hardware covers the individual servers supporting an SAP solution. Drilling down deeper, we find the specific memory, CPU, I/O, and other server hardware subsystems or layers, too.

Further, multiple solution stacks typically exist in any given solution. For example, the general SAP desktop GUI solution stack might consist of an HP desktop running Microsoft Windows XP, HP's OS client drivers version 4.4, Internet Explorer 6.x, and so on. The general SAP BW server solution stack, on the other hand, might consist of an HP Proliant DL760 server with firmware and OS drivers from SmartStart 5.3, running Windows 2000 Advanced Server with Service Pack 2, and loaded with SAP BW 3.0B.

One of the keys to a sound technology solution is assembling a solution stack that is supported by all of the various technology vendors involved in the solution.

Assembling such a supported configuration is by no means trivial! This is one of the reasons why so much time is put into vendor and overall solution selection—minimizing the number of technology players while bringing together a supportable end-to-end solution is the ultimate goal.

Obviously we are interested here in SAP's technology solution stack, but you can apply this same approach to any technology or solution. That is, Exchange 2000 has its own unique solution stack, as does an Oracle iProcurement solution or a PeopleSoft HR/Microsoft SQL Server 2000 solution. The enterprise product/system might differ, and the solution stack will certainly differ, but the approach to building a supported and well-performing solution remains constant.

How to Speak SAP—Terms and Terminology

We have already covered quite a few terms and acronyms. However, especially if you are new to or a bit rusty in using SAP's general terminology, you should understand the following list:

Don't worry about memorizing all of this right away—to keep the book useful to all levels of readers, I will continue to spell out acronyms and explain key terms throughout the book.

- SAP Component—Any number of SAP’s products, like R/3, Business Information Warehouse (BW), Advanced Planner and Optimizer (APO), Customer Relationship Management (CRM), Enterprise Buyer Pro (EBP), Product Lifecycle Management (PLM), Strategic Enterprise Management (SEM), and so on.
- Instance—An “installation” of an SAP product, which ultimately equates to an SAP component with its own set of work processes.
- SAP R/3—The most popular and prevalent SAP component, R/3 is simply a client-server-based online transaction processing system. It includes functionality like Asset Management, Financial Accounting, Human Resource Management, Plant Maintenance, Production Planning, Quality Management, Sales and Distribution, Materials Management, Business Work Flow, and more.
- Landscape—The collection of systems supporting a single solution (SAP component) like R/3, BW, CRM, and so on. Note that each solution requires its own SAP system landscape.
- 3-System Landscape—Typically, each SAP Solution requires a Development environment, a Quality Assurance/Test environment, and a Productive environment.
- Central Instance (CI)—The main “SAP” installation in a system (as opposed to the “Database Server” installation or dedicated application server instances, and so on). The CI is responsible for managing locks, inter-server messaging, and queuing.
- System—Typically consists of multiple instances. For example, your SAP R/3 system may consist of a database instance, a CI instance, two batch server instances, and five application server instances.
- Client—A legal entity or “business” within an instance—this is what end-users actually log in to with their unique user IDs and passwords.
- SAPGUI (pronounced sap goo-ee)—SAP’s “classic” Graphical User Interface, which provides a Windows-like look and feel. Other accessibility options exist as well, including a number of Web-based user interfaces.

Organizing the Book—Parts and Chapters

This book is organized into four high-level sections, or *parts*. Chapter 1 precedes these parts, laying the groundwork for the rest of the book. Part One, “Preparing for Your mySAP Business Suite Implementation,” consists of the next three chapters. These chapters commence in Chapter 2 with addressing planning, business requirements, uptime, and service level requirements and considerations. Here, I also take a

close look at the SAP Solution Stack, tactical and strategic reasons for implementing mySAP solutions, and the critical roles that both executive buy-in and business unit buy-in play in ensuring a successful implementation. Discussions on measuring success, measuring actual return-on-investment, assembling a real-world SAP project steering committee, drafting a customized high-level project plan, and the critical role of change control are also incorporated in Chapter 2.

Then in Chapter 3, “Crafting Your mySAP Solution Vision,” we move into refining and communicating a vision of the “to-be” or “future-state” of the SAP system, including the impact that the new enterprise solution will have on the business from the perspective of change. The importance of solidifying and capturing this solution vision is presented, along with a discussion of the different technology perspectives that a company might have, and how each perspective impacts the project. Other things come into play too, all of which drive building a “Request For Information,” or RFI, that will assist you in sharing your detailed SAP component requirements and infrastructure biases with various SAP technology partners. This includes how the SAP system landscape for each component should take into consideration or address the following factors:

- Simplification
- High Availability
- Disaster Recovery
- Training
- Performance
- Scalability
- Total Cost of Ownership
- Security Requirements
- Manageability
- Accessibility

Next, I consider outsourcing as an alternative to hosting SAP infrastructure in-house, and then I begin Chapter 4, “Designing and Initially Staffing the SAP Technical Support Organization (TSO),” with discussions of designing a support structure and organizing the folks required to implement and manage SAP. The critical roles of Project Manager and SAP Solutions Architect are covered here, and I take a closer look at recruiting other key positions as well. Real-world organization approaches and best practices are then followed by a comprehensive study in the value of training internal folks versus bringing in consultants. I include case studies where different and sometimes extreme approaches were met with mixed success. I end the

chapter, and Part One, with the task of revising budgets in preparation for the next part of the book.

Part Two, “Getting Started—Sizing and Blueprinting,” revolves around sizing and architecture activities, including preparations required beforehand. These activities are approached landscape-wide, and start off with a discussion in Chapter 5, “Total Cost of Ownership Analysis in Preparation for Sizing,” on identifying drivers and characteristics of an SAP solution that impact total cost of ownership. The SAP Solution Stack is examined with a critical eye toward how each layer contributes to costing, as well as how each layer contributes to solution requirements like high availability, performance, and scalability. Discussion on people and process costs are covered here as well. Chapter 6, “Identifying High Availability and Disaster Recovery Requirements,” introduces us to the general term “availability” and how high availability and disaster recovery must be tackled early in the sizing process. Each layer in the SAP Solution Stack is analyzed in detail, and real-world approaches to increasing system availability are identified and discussed. I conclude this chapter with availability best practices, including the role of the Disaster Recovery Organization.

With our budget, business, and availability requirements in hand, we then move into Chapter 7, “Sizing: Engaging the SAP Solution Stack Vendors,” and begin working with the various hardware and software partners capable of assisting us with sizing an SAP enterprise solution. Configuration rules and approaches to sizing SAP’s discrete components are presented, and I share methods and approaches for effectively comparing each vendor’s solutions, beginning with building a team responsible for evaluating sizings. I next cover how to plan for and conduct the “Planning and Scheduling” meeting with the selected short list of vendors, ensuring that all involved are on the same page before moving forward. Chapter 7 concludes with information on how to structure the first few days of “pre-planning” meetings with your selected solution stack vendors, setting the stage for implementation and critical path scheduling.

Chapters 8, 9, and 10 cover staffing, training, and developing the SAP Data Center, respectively. Chapter 8, “Staffing the Technical Support Organization,” opens with key interview techniques and actual questions, which are designed to help you staff your project with the best people. And then I cover the often overlooked topic of how to actually retain these key project assets throughout your SAP undertaking, including addressing compensation, incentive/training programs, compensation alternatives, and something I like to call “the most important thing.”

Chapter 9, “Critical SAP Training Requirements,” drills down into training—who needs it, when they need it, why the topic rates its own chapter, and how such training may be delivered and administered. The role of your SAP system landscape and how it may be leveraged to facilitate training is sorted out here, too. Finally, in Chapter 10, “Developing the SAP Data Center,” we dive into the details surrounding

planning for and developing the data center ultimately responsible for ongoing SAP operations and day-to-day support—a very critical task indeed! In reality, this chapter can only be described as both comprehensive and unique in the world of SAP books and articles. I actually walk you through the same steps that I take with my own SAP customers, including: power and cooling requirements, network infrastructure planning, optimizing data center real estate through rack planning and management, designing and implementing basic Storage Area Networks (SANs) and other common disk subsystems, addressing various operating system installation approaches, and much, much more.

Chapter 11, “Performing mySAP.com Component Installations,” is geared toward actually installing specific SAP components, and filling in any voids in your SAP system landscape. A quick discussion of realistic implementation techniques is then followed up with actual working checklists and approaches for the basic SAP Solution Stack layers, covering SAP R/3, APO, EBP, Web Application Server 6.20, and other popular SAP solution components. SAP client strategies, post-installation checklists, and more real-world observations and challenges wrap up Chapter 11, and Part Two as well.

Chapter 12, “Rounding Out Support for SAP,” starts Part Three of this book, “Moving into SAP Functional Development.” I detail how to fill in the holes in your SAP Technical Support Organization, which at this point in the project requires specific product specialists and technical skills specialists required to meet near-term goals or support SAP on a long-term basis. Broader support functions, like the role of the SAP Help Desk and SAP Operations teams, are also addressed here. I finish this chapter by taking a look at how other customer organizations, both large and small, attend to their own unique SAP TSO, help desk, and operations functions. Next, in Chapter 13, “Gaining Control of Change Control,” I focus on the importance of managing change, including planning for change, organizing the change control function, and the ramifications of ignoring or side-stepping change control. Change control best practices, worst practices, and *lessons learned* close this chapter.

Chapter 14, “SAP Systems and Operations Management,” brings us to SAP systems management and other critical operations-derived functions. I include a comprehensive recap of the most popular SAP management toolsets and approaches available today—including what many SAP customers *actually* do in terms of day-to-day SAP systems management. I also drill down into best practices when it comes to operations, documentation standards and approaches, and other management tasks and tactics, much of which may be immediately leveraged by my readers.

Part Three, “Moving into SAP Functional Development” concludes with Chapter 15, titled appropriately, “Functional, Integration, and Regression Testing.” In this chapter I contrast business process testing with other forms of testing, such as “load/stress” testing, and offer a number of tried-and-true tools and approaches.

Although my focus is often on SAP's eCATT (Extended Computer Aided Test Tool), I also address people and process considerations before wrapping up with a discussion on the weakest link in an SAP implementation—coding.

The final section in the book, Part Four, "Planning for Production Go-Live," walks you through setting up your production system and then stress-testing it to ensure that your mySAP solution performs as expected. To that end, I leverage my deep experience in customer-specific SAP stress testing in Chapter 16, "Systems and Stress Testing," contending with tasks like:

- Testing discrete solution stack layers and components before performing solution-wide testing
- Identifying the mix of SAP business transactions to test
- Determining how to best generate a representative load on the system, including the ratio of online users to batch processes
- Identifying stress testing success criteria
- Developing and testing scripts
- Identifying methods of obtaining enough data to truly "pull off" a stress test
- Determining the need for "real users" versus the value in virtual client driver methods
- Monitoring the SAP stress test up and down the SAP Solution Stack

Much more is covered as well, including the value derived in testing failover scenarios, verifying that systems management processes work as expected, ensuring that tape backup/restore procedures are successful, and so on. And I look at a huge variety of testing tools and approaches, ranging from freeware scripting utilities to fully-functioned "SAP-aware" testing suites costing thousands of dollars but capable of delivering that much more value.

Finally, in Chapter 17, "Preparing for SAP Go-Live," I culminate all of our work in designing, implementing, staffing, and testing with one final set of exercises—planning for Go-Live. The value of dry runs, locking down the system in terms of change management, verifying data integrity, and performing various SAP Solution Stack final reviews are all explained in detail. I also look at the evolving role of the SAP Technical Support Organization at this stage in the project, discuss various service and support options, and develop a "post-implementation" evaluation useful in documenting what the SAP implementation team did right and where improvement is possible. I conclude the book with high fives, congratulations, and toasts all around, as we walk through the non-event we call "Go-Live," and then spend a few

more pages on the value that Book Two—the next book in this series—will provide in terms of achieving SAP operational excellence after Go-Live.

Special Materials Provided by the Book

Let's face it, if I wrote this book like everyone else, it would simply be an update to a amalgamation of documents you could have picked up a few years ago. Hey, it might even put some of you to sleep! Instead, what I have decided is to wrap up my materials with real-world insights and observations, leveraging a few different approaches:

- You will notice chapter sections containing the words "In the Real World." This is one way that I apply what has been discussed in the chapter and relate it back to what I have observed first-hand in the real world. Doing this lets me inject my own experiences and stories into the dialog. You will find these personal accounts interesting, I'm sure. You might even laugh out loud on occasion—it's cool, I won't tell anyone.
- Each chapter (except this one) contains a "Tools and Techniques" section. As my youngest child, Ashley, likes to add when ordering a complex meal at her favorite fast-food franchise, I like to think that the "Tools and Techniques" section contains "all the stuff"—Checklists, Microsoft Project Plans, Microsoft PowerPoint presentations and Excel Spreadsheets, various tools and utilities, scripts, document templates saved in Microsoft Word format, best practices, approaches, alternatives, and more.
- The *Planning CD* enclosed with every copy of the book is the home of all of these tools and techniques, as well as some other pretty valuable utilities and documents. Enjoy. This represents the work of thousands of hours of consulting. And the best thing is that someone else footed the bill for you!
- Another reference made throughout the book involves the MCL, or *Master Check-Off List*. This is a simple document in design and appearance, but will prove invaluable in tracking progress, planning for meetings, and as a pointer or reference to tools and utilities that may prove useful on the Planning CD.
- Finally, the *SAP Infrastructure Project Plan*, or SIPP, is a comprehensive project plan that may be leveraged in support of the bulk of your SAP Implementation Project or SAP Infrastructure Upgrade.

Figure 1.3 illustrates my approach. In this way, the book not only serves as a reference to guide you through planning for and implementing SAP, but it also makes the book a lot more fun to read. Entertaining, even!

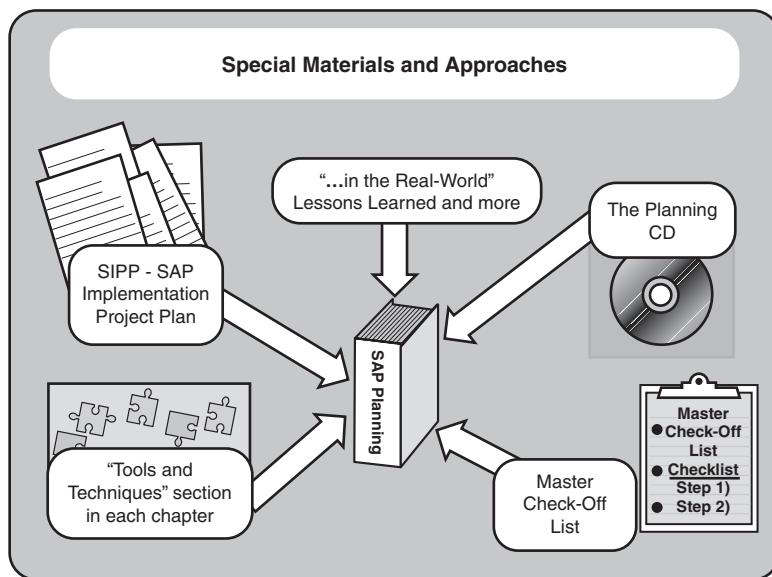


FIGURE 1.3 Special materials and approaches help make this book both valuable and entertaining.

More on Leveraging “Tools and Techniques”

I really liked the idea of sharing all of the “stuff” that I alluded to earlier in formats that you can actually take advantage of—that’s why I included the Planning CD and the “Tools and Techniques” section at the end of each chapter. For example, many of the diagrams and all of my presentations are saved in Microsoft PowerPoint format. In this way, you can simply copy them to your own computer, modify them as you like, and present them as your own. Think of the time you’ll save when asked by your Project Manager to justify, plan for, and present your SAP pre-Go-Live Stress Test approach. Or when you are asked to diagram and explain a number of SAP system landscape training alternatives and present them to your CIO. Or when you are requested to present the pros and cons of various SAP disaster-recovery options to the steering committee.

I also tried to stay away from inflexible file formats as much as possible too, in favor of more space-consuming but easily modified Microsoft Word documents. Again, the time you will save customizing things like my pre-made *SAP Daily Operations Checklist*, or tweaking a high-level *SAP R/3 Installation Guide*, or working through a *Sizing “Rules of Thumb”* document for Enterprise Buyer Professional might even put me on the top of your Christmas list.

To facilitate number crunching, or simply provide an easy-to-read side-by-side comparison document, I have also included various Microsoft Excel spreadsheets. A prime example includes my Sizing Evaluation document, where you can easily rank and stack multiple vendors against each other, thereby facilitating better apples-to-apples comparisons. I have also included a number of other simple sizing exercises, SAP candidate ranking forms, Disk Subsystem/SAN design documents, and more, all in Excel format to make your life easier when inevitable changes to your master plan force you to change, too.

Need a free utility to help you fine-tune or optimize your disk subsystem? Want to compare a RAID 5 configuration to a RAID 0+1 solution in terms of random writes? Curious about what a stress-test script for BW or an R/3 VA01 might look like? Here, too, I have just what you want at your fingertips—where allowed by law and the software vendor, I have included programs, utilities, scripts, documentation, and other such tools on the CD. Where not allowed, I have included Web site addresses, download instructions, or other instructions on how to obtain additional tools and utilities that could prove useful.

Finally, my pièce de résistance includes a number of Microsoft Project plans, designed to let you hit the ground running, not to mention keep you on track. My project plans are the results of years spent on actual SAP implementations and infrastructure-based engagements. They typically map directly to SAP's proven ASAP methodology, and can be easily adopted to the newer ValueSAP and Solution Manager approaches if need be.

The MCL and SIPP

Pay special attention to my *Master Check-Off List* (MCL), an easy-to-use Word document intended to keep my readers on track both while reading through this book and during an actual SAP project. The same can be said for the *SAP Infrastructure Project Plan* (SIPP). I recommend that you use the SIPP and my other plans as a starting point, customizing each as you require for your specific circumstances. And even if you already have plans of your own, use mine for comparison's sake, helping you determine whether you addressed all relevant tasks and milestones. The SIPP is comprehensive, it's real, and it includes detailed tasks like the following:

- Establishing the SAP Technical Support Organization (TSO) as part of the overall SAP Project Team
- Defining a system landscape strategy
- Identifying fourth-tier Internet/intranet requirements
- Architecting each environment in the SAP landscape
- Procuring and building out server and disk subsystem hardware
- Installing SAP instances (including multiple instance approaches)

- Addressing SAP Operations training and support requirements
- Identifying representative SAP load/stress test online and batch transactions
- Defining output requirements (including print, fax, email/workflow, and more)
- Addressing Central User administration
- Establishing transport processes, under the umbrella of Change Management
- Determining an SAP Support Package maintenance approach
- Preparing the Disaster Recovery site

And these are only the beginning! The usefulness of this single project plan alone is invaluable and should more than pay for the cost of this book a hundred times over.

Summary

This first chapter answered such questions as: who should be reading this book, why it will prove invaluable to them, how the book is structured, and how it can be leveraged in support of a successful SAP implementation—which will usher in for you a new age of enterprise integration and information sharing!

To this end, I have also touched upon the value provided through the “Tools and Techniques” section found at the end of each chapter, and the general contents of the enclosed CD. I expect that the checklists, templates, approaches, project plans, and more that I have provided will position you, my readers, to not only hit the ground running, but to do so with the confidence, knowing that thousands of installations before you have laid the groundwork—paving your road to SAP success.

2

What an SAP Project Looks Like

In the Beginning

If you are new to the world of SAP implementations, you're in for quite a ride. And if you're a veteran of multiple deployments and simply putting in another system, you're still in for quite a ride. There are many reasons for this, including:

- No two SAP solutions are identical.
- Planning and deploying SAP R/3 differs in a number of ways from deploying SAP BW, which in turn differs from deploying SAP CRM, and so on.
- SAP solutions solve enterprise-wide business problems, hence the requirement to work with not only technology folks but also the appointed business liaisons assigned to support the project.
- SAP, by its very nature, is used to solve exceptionally complicated business problems. That is, given its role as an enterprise application, and its value in integrating historically disparate data and functions, an SAP solution literally can affect every functional organization in the company. To effectively solve these business problems then requires a daunting array of hardware and software technology as well, further complicating an SAP implementation.

IN THIS CHAPTER

- In the Beginning
- Tactical Reasons for Implementing SAP
- Strategic “Big Picture” Goals of Implementing mySAP.com
- Beware—When SAP Projects Are Less Than Successful
- Setting the Stage for a Successful Implementation
- The Structure and Role of the Steering Committee
- Moving Beyond the “Napkin” Plan
- Tools and Techniques

- SAP implementations inherently impact mission-critical business functions. Thus, grave attention must be paid to creating additional SAP environments beyond the production system, that support key development, testing, integration, and other activities necessary before deploying an application that supports an entire company's financials, supply chain/inventory management processes, human resources management, or data warehousing and reporting needs, for example.
- Finally, given all of the preceding points, SAP affects how the company will actually do business in the future. Business processes and procedures will be changed, and these changes in turn will need to be embraced by the business community to indeed pull off a successful SAP implementation.

Over the course of the next few sections in this chapter, we will take a closer look at the challenges many companies face prior to deploying SAP, including tactical and strategic reasons to do so, and a 20/20 view in hindsight after a successful implementation. To be fair, and put the project's magnitude in perspective, we will conclude this discussion with the potential results of a less-than-successful implementation, too.

The Business Needs Help!

If your company is like most companies, you have a number of Information Technology (IT) systems in place today that handle the needs of your business. Some of the systems might focus on financial management, others might assist in scheduling production lines, some might support your sales team in placing and tracking sales orders, and so on. But if you have picked up this book, it's probably because these systems are not meeting all of your business needs—this book is written under the premise that a new SAP project will be getting underway soon to solve one or more of these business problems.

Thus, a team has probably already been assembled to take a look at different enterprise packages, and this evaluation culminated in selecting an SAP solution. The team certainly looked at the current systems in place too, perhaps with an eye toward patching or modifying them, upgrading to a newer release or platform, or changing some other component of the solution stack. You probably weighed the pros and cons of simply dealing with the existing systems, as well, and found that option unappealing for any number of reasons, such as:

- The current systems are not capable of providing decision-making data to management teams, thereby impacting profitability, time-to-market, and other key tenets of staying in business.

- The current systems represent vertical “stovepipes” of data and processes, with little or no sharing of this data between different systems or business organizations within the company.
- The costs of maintaining the current systems are out of line, perhaps due to the cost of hardware maintenance, database license fees, application maintenance or upgrade costs, or even the costs of the people supporting these systems. Worse, as costs are incurred to simply *Maintain* these legacy systems, there is no overall business process or data integration improvement like that realized by an SAP implementation.
- New business units have been acquired, or need to be incorporated into the current systems, and these current computing systems are simply unable to effectively handle the increased user or transaction load.
- The business community or IT community has grown tired of a “band-aid” approach to maintaining the current systems—they are old, and are overdue for retirement.

The challenge then becomes how to migrate from one way of doing things to a new and better way, in essence reinventing the company along the way, while still keeping the company running. This process has been compared to changing the tires on a car while it's doing 70 miles an hour on the highway. Not only is this type of migration extremely difficult, but doing it with little regard to process and best practices risks putting the entire company—the car, in this case—in grave danger.

Organizations must therefore carefully examine their current suite of information systems to determine whether they are both effective and efficient in supporting the organization's core business processes. They must also look at those business processes themselves, and again determine whether these are also effective and efficient. And finally, everyone must be prepared for change.

The changes will be huge, no doubt about it. In the end, not only will a new way of doing things exist, but a new solution stack—the SAP Solution Stack—will be cemented into place. As you see in Figure 2.1, the SAP Solution Stack can be quite complex in and of itself, covering data center and network infrastructure, hardware platforms, database management systems, SAP core products and components, diverse client accessibility models, operations processes/procedures, and much more.

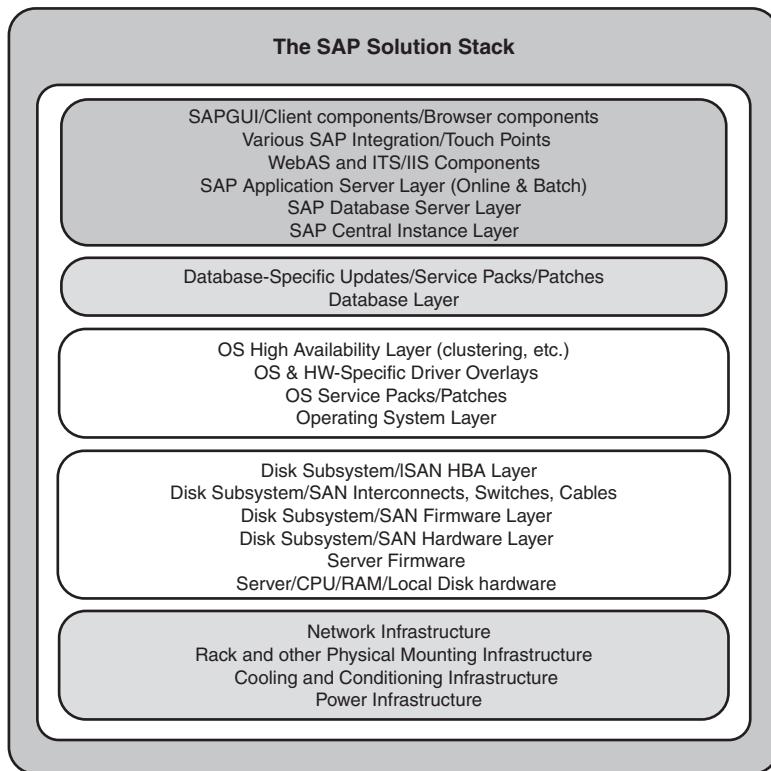


FIGURE 2.1 The individual solution components that make up the all-encompassing “SAP Solution Stack,” loosely divided here into high-level Data Center infrastructure, Hardware, Operating System, Database, and SAP-specific layers.

Tactical Reasons for Implementing SAP

As you saw in the preceding section, there are a number of reasons *not* to remain on a currently implemented IT system. However, the people controlling the financial purse strings are usually much more interested in how an SAP solution will address the nagging issues and challenges that crop up in the course of doing business every day. In effect, they tend to say “Don’t tell me about problems, show me solutions” before they write the big checks. My customers have shared the following tactical reasons why they chose to implement SAP:

- To improve product quality or availability by better managing and reporting on assembly line defects, reducing overall product inventories across warehouses and distribution centers, and enhancing other supply-chain functional areas
- To improve customer service in terms of increased knowledge of customer trends, status of orders, better turnaround on returns, and so on
- To address shifts in customer demand through analyzing buying trends across various geographies, customer demographics, distribution channels, and so on
- To provide one system or portal by which to manage activities in a number of other systems "behind the scenes"
- To increase competitive advantage by supporting rapid changes to business processes as these are deemed important
- To reduce customer billing time
- To increase inventory turns/reduce inventories
- To decrease lead times for production
- To manage cash better though insight into company-wide cash flows
- To address changing government regulations through integration with standard tax and other systems
- To improve resource planning and allocation through better project management and production planning efforts
- To respond to changing market conditions by rapidly analyzing customer and product trends
- To fold all company units into one "system" of record

Countless other tactical issues that may be solved by an SAP implementation abound in most organizations. I welcome hearing about yours!

Strategic "Big Picture" Goals of Implementing mySAP.com

Although tactical reasons for implementing SAP are pretty easy to identify, a number of general "big picture" goals bear discussion as well. As you can see in Figure 2.2, these goals include systems/data and core business application integration, improved operational reporting, improved strategic reporting, and support for creating new or improved business processes. These goals are explained further in the following sections.



FIGURE 2.2 Strategic goals of SAP implementations tend to improve the manner in which the company conducts business.

Application Integration

First and perhaps most obviously, SAP allows islands of data and processes to be collapsed into a single system, like SAP R/3. In other cases, a slew of systems may be collapsed into a few better-integrated systems, like R/3, APO, and perhaps PLM. In my experience, in fact, it is not uncommon to replace five or more existing systems with one SAP solution—I see this especially in manufacturing realms, where each plant or distribution center tends to hold on to a certain way of doing business (and the systems that allow this) until SAP or one of its enterprise competitors is introduced.

Improved Operational Reporting

When it comes to integrating day-to-day operations across diverse functional or business areas, SAP has been king of the hill for some time, in the form of *online transaction processing* (OLTP) systems like R/3. OLTP represents the operational control level of an organization. In other words, it addresses and consists of the huge volumes of transactions that support the day-to-day processing activities of a business. Out of this daily workload come aggregated reports produced in support of tactical or daily management control—the reporting is driven by and reflects the transaction processing data. To summarize this data, Executive Information Systems (EIS) provide a

method of extracting, combining, and summarizing R/3 data into a true management information system, for use at an operational control level. The unique thing about mySAP, though, is that a diverse set of data may be brought together and reported against at other levels as well, and in a real-time manner. Operational data may be pulled from various mySAP components and other legacy systems and dropped into the Business Information Warehouse, for example.

Improved Strategic Reporting

Strategic reporting is also facilitated by a number of mySAP components, including SAP Business Information Warehouse, Strategic Enterprise Management, and a number of reporting systems inherent to R/3. These types of reporting systems are traditionally referred to as *decision support systems* (DSS) or online analytical processing (OLAP) systems, and they focus on supplying information for special reports and analyses that are typically not available from operational data formatted in simple aggregate or summarized form. DSS systems also allow bringing in third-party data (that is, external information on economic and industry conditions) simply not available out of the day-to-day OLTP systems. SAP became especially good at this kind of strategic reporting with the introduction of BW, and SAP's Strategic Enterprise Management component takes these strategic capabilities to the next level by providing an electronic cockpit of sorts from which to manage the business. In both of these cases, implementation of the data warehouse allows managers to access and analyze information that ultimately supports better decision making.

Note that without such strategic reporting capabilities, organizations find themselves in a quandary. That is, without the data-unification enabled by mySAP solutions, strategic reporting is often simply not done. Why? Because when the data must come from many uncoordinated and disparate systems, the time to collect and analyze the data may exceed the window of opportunity for doing it; the cost may be prohibitive; and quite often it may simply entail too much work to do well, only because similar data maintained in different systems is coded differently. For example, I know of a customer that had over 30 different formats for storing customer names alone—it was next to impossible to cobble together a cohesive snapshot of the business without incurring a massive amount of tedious manual labor, until they implemented an SAP solution.

Flexible Business Process Support

Probably the most obvious improvement brought about by SAP involves the opportunity to re-examine business processes, and revise them to reflect industry-standard best practices. Often referred to as *business process re-engineering* (BPR), this long sought-after goal of an SAP implementation seeks to push analytical questioning throughout each business unit. In other words, BPR dictates that the old ways of

doing business should be questioned, and in the case of inefficient or ineffective processes, revised. To this point, I have worked with more than a few companies that, as a result of BPR exercises completed during the SAP planning process, actually injected business process changes to their *current systems* before SAP Go-Live, simply because the changes were so compelling.

For example, one large SAP implementation of mine replaced a number of large accounting and inventory systems deployed throughout a global organization. Inefficiencies were the rule rather than the exception—no integration between the legacy systems meant that a change in one system needed to be manually made in each of the other systems as well. Data maintained in one system was not globally available to the other systems, either. Ultimately, the task of coordinating and executing all of the manual processes needed to synchronize the systems and data across all of the systems grew to be too much for the old IT organization. When SAP was introduced, however, it forced the business folks to get together and agree upon a better solution—re-engineering the business process itself made the difference. Fairly simple changes were made in their current systems, and ultimately the deployment of SAP facilitated an even better implementation of this business process. In the end, it was clear that business opportunities are actually enhanced when re-engineering occurs that is focused on supporting better integration and improved processes by *encouraging communication and questioning*, in accordance with the company's strategic objectives and policies.

How SAP Has Benefited Customers in the Real World

From a post-implementation perspective, I thought it would be interesting to identify how a number of my customers truly benefited from implementing SAP. Unlike specific tactical or strategic reasons, these benefits represent the “real world” impact that SAP has had in these organizations or companies. I have left out actual company names, of course, in the interest of preserving anonymity.

- One company improved its decentralized and uncontrolled procurement processes by implementing SAP’s e-enabled enterprise procurement system (today labeled Enterprise Buyer Professional), saving hundreds of thousands of dollars annually by reducing “after the fact” POs, realizing bulk discounts, and reducing rogue buying.
- One of my customers completely revamped their invoicing system, eliminating the creation and tracking of 175,000 paper invoices per year, while increasing both controls and quality.
- The same company was also able to reduce headcount by 19 Full-Time Equivalents within one year of deploying their mySAP solution.

- Another large company saved more than \$3 million annually in employee productivity by simply moving their expense reporting system to a Web-enabled SAP CATS (time-entry) HR system.
- Compared to their legacy system, another client serviced by my team saved over \$7 million annually by moving its sales/distribution and order entry system to SAP R/3.
- One of my favorite customers was able to retire its legacy UNIX/enterprise environment running another ERP solution, in favor of a Microsoft Windows 2000/SQL Server 2000-based SAP solution, saving \$800,000 a year in database licensing costs alone.
- Although the change was not quantified in dollars, one of my customers talks about their new reporting and decision support systems shrinking their month-end processes by an incremental two days, thereby providing a jump on the current month's production scheduling, improving cash management, and enhancing their competitive position.
- By sharing transactional, master, and meta data via one SAP BW system, another company has reduced clerical data-entry positions by 40% while improving the quality of both their regularly scheduled and ad-hoc reports.
- One of my team's first SAP CRM customers shared some very interesting data through an ROI study. The results indicated that not only did the project break even in 13 months, but that an increase in average order size of over 12% (due to rich data mining capabilities and SAP CRM's campaign-management tools) would serve to maintain better than a 70% internal rate of return over the next two years.
- Attributing their 80% ROI to process cost savings, labor cost savings, and more efficient analysis of their own purchasing activities, another high-visibility SAP SRM customer was able to reduce their number of suppliers from nearly 1,500 to approximately 200.

Over the years I have seen first-hand over and over again how the changes brought about by SAP impact not only an organization's business processes, but also the very political and power structure of the company. In fact, in hindsight it always seemed pretty clear who stood to lose or gain ground in this regard from the very beginning of the project. Individuals who were known as "squeaky wheels" and "difficult to work with" tended to see their position in the company, or their power base, eroded. Individuals who historically created artificial positions within their company by limiting access to critical information found their jobs eliminated. An SAP implementation drives collaboration and partnership between different organizations and

across different teams of business expertise. People who work well in such environments thrive, but those who focus strictly on localized priorities tend to become obsolete.

My customers are often surprised post-implementation by the mix of IT and business personnel working on the SAP implementation. Although historically, most information technology projects tend to be driven by the IT organizations, time and time again I see a 70/30 or 80/20 implementation team mix—70–80% business folks, and 20–30% IT-centric people.

Beware—When SAP Projects Are Less Than Successful

Not all SAP implementations are roaring successes. In the following sections, I take a closer look at some “not best” practices, including in some cases what went wrong, and in other cases how the SAP project may have failed to meet the expectations of its stakeholders—end users, functional business-process-oriented organizations, and the folks controlling the financial purse strings. If you fall back on the scenario of the “car moving at 70 miles per hour,” it becomes evident here that there is truly a great deal of real risk that must be mitigated when tackling an SAP project.

“Not Best” Implementation Practices

Some of the “not best” practices my team has run into may be found in Figure 2.3 and are described in the following list. I suggest that you read these carefully and tuck them away for future reference under a big blinking neon banner titled “Lessons Already Learned by My FAILED Competitors”:

- Not willing to change current business processes. Usually, it is under the guise of keeping things simple so that the implementation goes well and goes quickly. In other cases, I found this to be the result of not having a strong executive order to make changes while simultaneously placing the wrong people on the project. Instead of staffing the project with decision makers, the business staffed the project with people who had merely *requested* that changes be made, often by superiors who found the current status quo comforting. In reality, though, not changing business processes actually represents one of the most common causes of failure for SAP development projects. And when it is coupled with a lack of current-process documentation, the problem is only exacerbated as the business works to figure out how things are done today, so that they can move the unutilized process to an SAP-enabled solution.
- Not learning from mistakes. Implementing SAP is an iterative process, in that implementing the various system landscapes and working on the various development and test clients offers an opportunity to improve upon even the limited IT processes already in place. A phased approach to implementing SAP

(phased in the sense that Development is put in first, then Test, then perhaps Training, and so on) also lets you learn from your mistakes, so as not to run into the same issues as each new implementation wave begins.

- Ignoring sound change control practices. This one scares me the most, because it's so common sense to put good change control processes into place, yet I see the results of not exercising good change control practices both before and after Go-Live. More mySAP Go-Live delays seem to result from this "not best" practice than any other reason, in fact, usually after something is changed in the system soon to become Production, without adequately testing the change first in another environment. Change control needs to be a mindset; without this mindset, it is only seen as an irritating pile of processes and paperwork, until an unplanned downtime event raises change control's visibility again.
- Lack of sound project management. Although this may be considered a broad "do not do," it's actually common in a number of manifestations. To achieve sound project management, first, be sure to understand and document the desired state solution vision. Second, follow a proven SAP deployment methodology. Third, ensure that the objectives of the implementation are clear and managed to. And fourth, engage project managers with leadership, administration, and excellent people-management skills.
- Relying too heavily on third-party consultants. This is another no-brainer, but a very common dilemma. Having a whole lot of consultants on staff is not the problem (it's the norm, actually). However, providing consultants with all of the opportunities or exposure to SAP at the expense of the company staff (who will be there long after the consultants leave) makes no sense at all. I also see a lot of projects where the third-party consulting staff hold too many key roles or positions. Again, the knowledge that these people acquire during the SAP deployment needs to be shared with the home team, not hoarded and carted away.
- Lack of documentation. Like reinventing the wheel, it's expensive and ludicrous to have to learn over and over again how to do something properly. Documentation in regards to the SAP Solution Stack, and the installation, configuration, support, and operations of each layer of the stack, goes a long way toward improving productivity and increasing uptime. Yet a lack of solid documentation still rates as one of the top areas in which a company fails to measure up, ultimately putting the entire project at risk. This is the reason I provide so much in the way of documentation on the Planning CD—I want my readers to be able to take advantage of the many checklists and guides I have developed specifically for this book, or at least to use them as kick-start templates of sorts.

- Lack of buy-in. I have given the process of developing and securing buy-in quite a bit of attention in this chapter. Buy-in not only means gaining executive-level approval, but also the approval and a sense of ownership on the part of the folks who will ultimately use the system. This includes functional organizations and the end users themselves.
- Failing to test adequately. This is more common than most people think. Don't misunderstand; the organizations guilty of this usually *believe* that they have done plenty of testing (and the hours spent in the name of "testing" often bear this out). However, what the numbers fail to reflect is that these failed or delayed projects skimped on either comprehensive integration testing, or end-to-end stress testing. This explains why each of these testing types merit their own chapters—integration testing is covered in Chapter 15, and stress testing is detailed in Chapter 16.
- Scope creep and "bolt-on" madness. Adding extra features, bells and whistles, and other nice-to-haves has pushed many projects beyond their originally published Go-Live dates. Not only does the add-on to the project lengthen time to implementation, configuration, and testing, it also impacts day-to-day operations as well as future support/maintenance needs.



FIGURE 2.3 Best practices? Not quite—I call these "Not Best" practices, or "Lessons already learned by my FAILED competitors."

NOTE

Change control refers to the practice of first testing a suggested or potential change to a production system in a technical sandbox or other SAP system, and then "promoting" this change through the landscape (to the development system, then test system, and so on), to ensure that none of the solution characteristics you required from the solution in the first place are compromised.

The next sections illustrate a few true-to-life Go-Live disasters, to demonstrate how the problem areas I've listed have caused actual SAP implementations to fail or come close to failing.

Ignoring Change Control

This first SAP Go-Live near-miss has probably occurred a hundred times at a hundred different places. Tens of thousands of man-hours had gone into designing a robust SAP system landscape, preparing each system for its role in supporting Production, developing and customizing and testing, and so on. Everything seemed to be on track, and only minor issues had cropped up here and there over the last month or so.

Two weeks before Go-Live, the basis environment was frozen such that no changes would be made. Each server's memory configuration had been specifically set up to take advantage of all 4GB of physical memory. Three gigabytes was to be used by SAP (through the use of Window 2000 Advanced Server's /3GB parameter, which allows a process to access 3GB worth of memory address space). The remaining gigabyte of memory would be available to the Operating System and other processes. This configuration had been validated during stress testing.

The system had apparently been put into "lock down" too early, however, because it was clear that some major tasks still needed to be performed. For example, the technical team wanted to reload the entire OS, database, and SAP system from scratch to ensure one final time that their process for doing so was indeed repeatable. They also wanted to perform some final SAP tweaking, to further optimize a number of database and SAP-related buffers. After backing up the system to tape, the contents of each server's disk drives were wiped clean, the OS was reinstalled, and the database and SAP installations were performed without a hitch. Late that evening, the system started up without incident, the senior SAP Basis analyst verified things looked good, and before sending the technical team home, started the master/golden client copy from development. The final memory-related tweaking would have to wait another day.

The next day, three days before Go-Live, the functional modules were locked down, and the system seemed ready for prime time. That night, however, an issue was discovered with the Production application servers after the memory buffer parameters were changed. It seemed that the SAP application server instances would die soon after being started. After careful analysis and troubleshooting, it was determined that the new memory parameters were not working because each server only registered 2GB of RAM, and the new parameters required more than this.

This was quite odd, though. In SAP stress-testing carried out a month beforehand, these servers had registered a full 4GB of RAM, the maximum supported by Windows 2000 Advanced Server. A quick look at the old profile parameters used in the stress

testing also verified that the system had been set up to work with 3GB of RAM reserved for SAP memory-related settings. And a review of the old BOOT.INI settings confirmed that the /3GB switch was set, and things looked good. Somehow, something had changed though.

Another few minutes of poking around the system revealed the culprit. As it turned out, the (very!) junior SAP Basis analyst who asked to perform the OS reinstallation actually installed plain old Windows 2000 Server. Then he installed the database client, and the SAP application server instance as usual. The installation of the application server instance defaulted to a smaller amount of RAM than previously seen, but our unsuspecting junior team member failed to notice the discrepancy. He brought up the application server instance, which quickly connected to the message server, and walked away confident in his new-found resume-building experience.

In retrospect, several things went wrong:

- Even after the Basis system was locked down, changes were made to the system. What happened to change control?
- No checks were in place to catch the fact that a wrong component of the SAP Solution Stack was installed. A checklist or installation recipe indicating the proper OS to install, or the amount of physical RAM to be seen by Windows 2000 or the SAP R/3 application instance would have avoided this issue early on.
- No one verified the changes to the system in a timely manner. That is, a second pair of eyes was not brought in to review the changes until three days before Go-Live.
- Changes were made too close to the Go-Live date. There was simply too little time left for “what if” troubleshooting between the changes being made to the system, and the date the system was supposed to be turned over to the end-user community.

Bottom line, the end users serviced by this Production instance got lucky. Had the botched OS installation gone completely unnoticed, and the system de-tuned to take advantage of only half the RAM in each server, end-user response times would have suffered. It is likely that the business-driven agreement in regards to response-time SLAs (Service Level Agreements, covered later in this chapter) would have missed the target, too. The new system’s end users would have walked away at the end of the first day with a perception that the new system was slow.

Hiring an Inexperienced Implementation Partner

A recent SAP Business Information Warehouse (BW) implementation I helped to clean up after a less-than-impressive Go-Live seems to be a classic example of “you

get what you pay for." Here, a novice SAP BW consulting organization's approach to implementation was the key to failure. But not all the blame could be pinned on this small implementation company. Consider the following points:

- The internal customer organization within the company implementing BW was not well-positioned to represent the needs of the entire company when it came to data warehousing. A successful implementation of BW requires close collaboration and agreement between and among all of the business areas involved, not to mention clear communication up, down, and across all of these organizations. An experienced BW consulting company would have known this.
- Although BW solutions address business requirements, they are still technology projects, too. Therefore, the implementation team requires a well-staffed technical subteam familiar with both SAP BW and the customer's own unique enterprise environment. The customer thought they were covered, then, believing that their own internal IT organization would supplement the consulting partner's business re-engineering and BW-trained folks.
- Implementing SAP BW benefits tremendously from a proven implementation methodology, like SAP's ASAP (Accelerated SAP) or ValueSAP approaches. These methodologies, and the newer SAP Solution Manager-enabled implementation methodology, historically reduce the time needed to successfully deploy SAP by 10–50%, as compared to implementations performed without a methodology. In this case, no one thought to ask much about what the BW consulting organization actually knew of ASAP, unfortunately, nor whether they brought their own methodology to the table.
- SAP BW implementations are complex in terms of the number of players involved and in the various technologies and requisite skillsets therefore required. A closer look at the consulting organization's "references" (we use the term loosely here) would have clearly removed them from contention. Two of the reference accounts were well-known names, but in reality our consulting organization under discussion had little to do with implementing the actual technology. And where they did have hands-on experience, the individuals who actually gained the experience had long since switched jobs.

This last bullet item is far and away the greatest problem faced by my customer. The other problems with the implementation and the incorrect assumptions on the part of my customer only exacerbated the bigger problem—the SAP BW consulting organization simply didn't have the requisite experience.

To be fair, it should be noted that nearly 60% of firms surveyed in a recent Gartner study gave *internal factors* as the primary reason for failure—my customer in the preceding example is not alone by any means. What this meant in the end, though,

was failure. They blindly followed their implementation partner into a minefield, without even knowing they were doing so. Key deadlines were pushed back several times due to technology issues (SAP BW extractor problems on a relatively new release of R/3, the need to provide reporting capabilities to remote users with slow network links, inability to uniformly stock cubes from which end users could run reports, and more). And even after a pile of budget money had been spent, the only thing the business could show for their efforts was a couple of inadequate cubes accessible only to local users in a single business unit. In hindsight, perhaps they should have characterized their efforts as an extended “pilot,” and moved on from there.

I came in after the fact, after Go-Live. Through a series of meetings and eventually a strategy workshop, followed up by gathering real requirements (again!), the new team helped turn the project around. It took actual experience with deploying SAP BW, knowledge and use of a proven implementation methodology, and a combination of careful listening/attention to detail to make this happen. Unfortunately, it also took a lot more time and money than the customer ever bargained for up front. After sharing this story with one of my colleagues, he remarked “This is the point at which somebody always says ‘why don’t we ever have time to do it right, when we always have time to do it over?’”

With examples of how *not* to approach an SAP deployment fresh in your minds now, let us turn our attention to the tasks awaiting a fresh deployment.

Setting the Stage for a Successful Implementation

With the d preferably in parallel. These things—tasks, analysis, even marketing of the project internally—are covered in the next few pages.

Promoting Buy-in Throughout the Company

At this point, senior-level folks have certainly signed off on the project. However, other executives typically must be convinced that the business units are on the right track. This is where the *Project Sponsor* spends his time initially, gaining consensus within business areas, functions, and the various IT organizations that will ultimately contribute to supporting the project.

As one of a few central figures within the SAP project’s Steering Committee, and generally the person responsible for putting together the membership of the Steering Committee, the Project Sponsor’s role is key indeed.

I discuss the role of the Project Sponsor and makeup and functions of the Steering Committee in more detail later in this chapter. For our purposes here, though, the Steering Committee continues to build upon the momentum put in action by the Project Sponsor, gaining buy-in and “talking up” the project throughout the

company. By working with the various business units to help them understand how important they will be to the project, and how much better the project will address their needs, excitement and buy-in around the project will continue to grow in these early days.

Finally, until a project-dedicated Project Manager representing the business organizations is identified, the Project Sponsor wears this hat, too. Thus, through these important roles, the Project Sponsor must

- Project confidence, presenting the SAP implementation with an air of competence in any forum. To this end, the Project Sponsor must be prepared to give a formal presentation at the drop of a hat, without the presentation appearing too “canned.”
- Be well-versed in the SAP project from Total Cost of Ownership and Return on Investment perspectives.
- Tailor language to his audience, be it the board room, shop floor, IT group, or functional organizations like Accounts Payable or the supply chain group.
- Be aware of the politics inherent to the various organizations involved. This includes determining who the informal decision makers are, as well as the ones granted this authority through organization charts.
- Be truthful and up-front about everything—both personal and professional credibility are on the line at all times. It’s impossible to have all the answers all of the time, especially in a complex enterprise planning project. Admitting “I’m not sure, but I’ll find out” makes a lot more sense than trying to dodge questions or skirt issues.

Nailing Down the Real Business Requirements

Although momentum is growing, a team of key business organizations should be working through the goals of the SAP project. I call this identifying the “must haves,” the “should haves,” the “nice to haves,” and the “blue sky stuff.”

- “Must haves” are core capabilities that can simply not be lived without, like the need to track and manage inventory and finished products, and produce accounting statements, and so on.
- “Should haves” might include the ability to run reports against an integrated information repository. “Should haves” can be worked around and lived without, but often provide the core foundation of value-adds to a solution.
- “Nice to haves,” or extras, might represent the key differentiators provided in certain business solutions, like the ability to access a solution using a standard Web browser. They do not represent requirements by any means, and often add more cost to a project than value.

- “Blue sky stuff” includes extras that would be really exciting to include, but simply cost too much, or are too complicated to implement, or represent other huge risks. The ability to securely make available your supply chain system to customers, suppliers, and vendors still represents “blue sky stuff” to many organizations, for example (even though it’s quite feasible from a technology perspective).

The same list of business benefits put together to get initial approval for pursuing the idea of implementing SAP will suffice as a starting point.

Mapping Business Requirements to Solution Characteristics

Using your list of business benefits as a solution foundation, the business and a few select IT professionals need to address the specific required *characteristics* of the solution, like the level of availability, performance, scalability, manageability, and so on that are desired.

Something I call the “Solution Characteristics Matrix” is often useful in ensuring not only that each characteristic is considered, but that all of the appropriate stakeholders have a say in how important they believe each characteristic is to the business. The matrix becomes documentation in and of itself, too, and will be referenced again and again throughout the project. For starters, I have included a sample matrix on the Planning CD, with the actual characteristics identified as follows:

1. Value placed on high availability/stability
2. Value placed on performance (speeds and feeds)
3. IT administration/manageability (tools/approaches/simple systems-level reporting)
4. Value of IT accessibility (that is, the ability to access the solution or hardware components via the Web, and so on)
5. General scalability (vertical scaling versus horizontal scaling, tiered architecture approaches, and so on)
6. In-the-box scalability (“platform” scalability—ability to add CPUs, RAM, and so on)
7. TCO/pricing (one time costs PLUS annual maintenance costs)
8. Simplification (manage fewer large components versus many smaller ones)
9. Security (vulnerability to security holes versus making access difficult, and the cost to accomplish this)
10. Self-serviceability (ability to service and maintain the solution internally, rather than relying on a third party)

11. Value placed on mature technology versus newer and potentially riskier technology (different technology perspectives, including conservative, close follower, and others)
12. Value placed on embracing new approaches to old problems, vision
13. Value placed on legacy stovepipe computing (where a single solution vendor provides the hardware platform, Operating System, applications, and necessary technical and business-related services for an entire system)
14. Value of access to solution-specific service professionals (like SAP BW functional consultants or Oracle database technical consultants, and so on)
15. Value placed on leveraging your current IT standards (where if your current enterprise systems rely on HP Proliant servers, Windows 2000, and Informix, you might be inclined to implement a new solution that supports some or all of these solution stack components)
16. Value placed on leveraging current IT skillsets

Remember to gather input from every single business group that will ultimately be serviced by the project. By doing so, you'll not only have a better understanding of the business needs driving the solution, but also gain valuable project buy-in through your endeavors, as previously discussed.

Determining Realistic Service Level Agreements

The term *Service Level Agreements* (SLAs) is by no means new or specific to SAP. As with any IT project, though, certain minimum requirements need to be communicated by the end-user community to the IT organization tasked with supporting the project. And this needs to be done fairly early in the project, as the technology base on the SAP project itself will be shaped to some extent by these minimum requirements. Why? Because these minimum requirements become the service levels against which the project will eventually be measured.

Service levels take a variety of shapes:

- Percentage of uptime, or availability

For example, a system might need to be in place that will provide 99.9% availability throughout the year.

- Planned versus unplanned downtime

This is usually expressed in hours per month, or percent achieved per year. A number of our medium-sized SAP customers plan on 4–8 hours of downtime per month, for example, to implement changes to Production or test failover

and other high-availability processes. Database backups, database reorganizations, and the archiving of SAP business objects represent other common planned downtime events.

- Reactive service levels and related support contracts

If the system suffers unplanned downtime, the business expects that the problem will be addressed in a certain time period, and resolved within a specific period of time as well. These metrics must be established up front, and then leveraged to draft service agreements with SAP Solution Stack hardware, software, and systems integration partners.

Work closely with the business groups to nail down SLAs, and then document these, as they will later need to be clearly understood by various SAP Solution Stack hardware and software vendors.

How to Estimate ROI Early in the Game

Entire volumes of books and complicated expensive software packages have been written with regard to estimating *Return on Investment* (ROI). In fact, Chapter 5 covers total cost of ownership and return on investment analysis for SAP in detail. What is important in the earliest stages of the SAP implementation project is “simply” comparing the total one-time expected implementation costs of the project to the in-place systems, and then factoring in recurring costs over something like a three-year time period.

It should be noted that an ROI or TCO analysis examines the following areas (and as shown in Figure 2.4), in terms of costs and expense:

- Processes
- People
- Technology

Ultimately, a number of sources may be leveraged to gather industry-standard costing data on the “future” solution, from excellent objective resources like Gartner and IDC, both well-known market research institutes. More challenging, though, can be collecting useful company-specific information when it comes to these areas.

Underneath this umbrella of processes, people, and technology lie budgetary/cost categories including the following:

- Acquisition (technology and people)
- Facilities
- Administration and overhead specific to the solution

- Break/fix labor
- Downtime
- Help Desk
- Installation/labor
- Management (asset and systems)
- Operations (computer/systems)
- Planning/evaluation
- Scalability
- Standardization
- Training

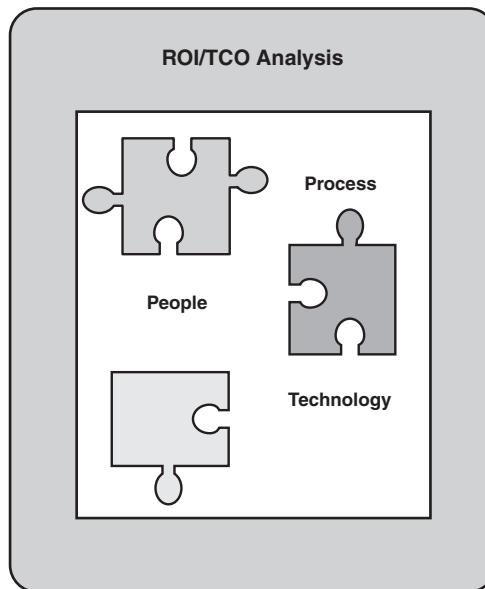


FIGURE 2.4 An ROI or TCO analysis seeks to quantify current and future-state Process, People, and Technology costs.

After data is finally collected on the in-place/current systems, a “first-round” and very rough vision of the future solution needs to be assembled. This exercise, though preliminary in many respects, will prove useful time and time again in the next few months, though. For example, the ROI exercise will be refined via the detailed approaches and processes described in Chapter 5. It will also impact the solution

vision, covered in Chapter 3. And ultimately the combined ROI analysis/future vision will help drive the sizing process (Chapter 7), designing and staffing the SAP Technical Support Organization (Chapters 4, 8, and 12), training (Chapter 9), systems management and operations (Chapter 14), and disaster recovery solution design (Chapter 6).

Translating Solution Characteristics into Technology Drivers

Given the requirements identified by the business units in terms of availability, performance, scalability, manageability, and so on, we are now in a position to begin mapping these requirements into *technology drivers*. They are called this because they drive the configuration of the SAP solution's technical specifications. Nearly all of Chapter 7 is focused on covering this process in great detail.

At this point, a very knowledgeable and experienced SAP Solution Architect or senior Basis/Infrastructure consultant is worth his or her weight in gold. The goal is clear—to analyze all of the data collected via the Solution Characteristics matrix, SLA discussions, and various meetings and so on, and then to come back with an order of magnitude hardware/software design/plan that reflects these requirements/constraints. In the real world, this is an ongoing exercise culminating with formal *hardware sizings* or more comprehensive *solution sizings* (I usually refer to them all as simply *sizings*), which are custom-crafted solution-oriented documents typically generated from SAP's hardware and software partners. The sizings reflect the entire SAP system landscape—the need for a Development system, a Test system, perhaps a Technical Sandbox or dedicated Training system, the Production system, and so on. A comprehensive solution sizing speaks to all of the infrastructure needed as well—network infrastructure, racks for mounting all of the hardware in the data center, power and cooling considerations, administrative or management servers/appliances, and so on. And the solution sizing usually details database layout requirements, the need for bolt-on applications, and more (as requested by a potential customer, or offered by a particular vendor).

The Importance of a Methodology

Obtaining buy-in from every stakeholder, understanding each and every business requirement, and so on can only at best lead to a mediocre implementation costing more than otherwise possible. It's the *methodology* employed, the structured and repeatable approach to SAP implementation, that makes all the difference in the world. SAP AG recognized that their customers would benefit from a standard roadmap to SAP deployment, and developed their ASAP methodology in response. ASAP, or Accelerated SAP, was originally intended for smaller implementations where the primary implementation partner was not necessarily a "Big 4" consulting firm. But the approach proved successful in larger SAP rollouts as well, and continues to

be used quite successfully even today. ASAP consists of five high-level milestones, including

- Project Preparation
- Business Blueprint
- Realization
- Final Preparation
- Go-Live and Support

ASAP evolved into GlobalSAP and ultimately ValueSAP, adding methodologies for Evaluation and Continuous Business Improvement to its core Implementation methodology. The roadmap changed a bit as well, shrinking to four implementation phases instead of five. With the adoption of mySAP.com solutions growing throughout 2001 and 2002, however, SAP sought to release both an improved delivery vehicle and a more comprehensive methodology that better reflected the challenges inherent to mySAP.com deployments. SAP's Solution Manager was introduced in late 2001 with Web Application Server (WebAS) 6.10, and was later updated in June 2002 to the much-improved version found on WebAS 6.20. By the end of 2002, this version was the first Solution Manager truly ready for prime-time, offering not only multiple roadmaps to implementation but also improved content (sample docs, new templates, a repository for canned business processes, and more) and better project management tools.

SAP Solution Manager is clearly superior in terms of capabilities, too. For example, Solution Manager may be used to support ongoing operations as well as implementation and continuous improvement activities. Robust project monitoring and reporting capabilities exist as well. Plus, it provides for a variety of ways to help you manage your project team's educational goals, including *Learning Maps*, which are role-specific Internet-enabled training tools featuring online tutoring and virtual classrooms. And with training and related support of the ASAP and ValueSAP methodologies expected to completely fade out by the end of 2003, the best game in town from SAP AG will soon be Solution Manager.

The only downside is the software and hardware requirements recommended to run Solution Manager—a separate WebAS instance is required, for instance. And best practices dictate that a two-system SAP landscape be created solely for Solution Manager, one system for production purposes, and another to be used for testing updates and changes to the SAP Solution Manager solution stack. Fortunately, server and disk requirements are reasonable. A dual processor server with 1GB of RAM and 50GB of disk space meets SAP's minimum requirements.

Although SAP's Solution Manager may arguably represent the best methodology for mySAP deployments, this is not to say that other methodologies do not exist or cannot be leveraged for successful project outcomes. On the contrary, all of the Big 5 and most of the large SAP technology partners offer a methodology for implementing SAP. The key is the degree of experience that the implementation team has with a particular approach, combined with the methodology's ability to support specific mySAP.com components and the challenges inherent to implementing some of these solutions.

Pinning Down the Initial Implementation Budget

Finally, with all of the data gleaned during the activities described in the last few sections, you can begin to put together more than simple preliminary budget figures. Remember, you're just getting the ball rolling at this stage—the actual budget numbers will become more apparent as you move through the first few phases of the project plan and gather all of the details surrounding people, processes, and technology. Until that time, though, a rough budget will go a long way toward ensuring that the business requirements and potential technology solutions are in line with each other dollar-wise.

Specifically, you can assemble the following costs for the entire system landscape (whether that's a traditional three-system landscape or larger), as well as some of the basic "people costs":

- Data center space capable of housing, powering, and cooling everything below
- Special power requirements (UPS, power distribution units, redundant power feeds, and so on)
- Server hardware (database server, application servers, WAS/ITS servers, other Internet servers, management appliances, infrastructure servers such as domain controllers, and so on)
- Disk subsystem hardware (each system in the landscape requires a database server, and thus a disk subsystem, database, license, and so on)
- Network infrastructure (switches, hubs, routers, and all cabling)
- License fees and ongoing annual maintenance fees for the operating system for each server
- License fees and ongoing annual maintenance fees for the Database Management system for each database server in every system of the SAP system landscape (a three-system landscape will require three database licenses)
- mySAP.com license fees and ongoing annual maintenance fees

- Management system costs (typically an SAP-aware application capable of monitoring the systems holistically)
- Incremental Computer Operations costs
- Incremental Help Desk costs
- Break/fix hardware maintenance contracts
- System installation (server, disk subsystem, OS, database, and each specific SAP component or product)
- Training costs (again, the entire SAP Solution Stack)
- Costs associated with hiring the project manager(s), project coordinator(s), project librarian/documentation specialist, and so on
- Costs related to technology-focused team members like the Solution Architect, database administrators, SAP Basis and other technical specialists, and so on
- Costs related to functional specialists, ABAP programmers, and other development/business-process experts
- Opportunity costs sacrificed by temporarily assigning people to the SAP project, and back-filling their previous line-of-business, technology support, or other roles within the company.

Certainly these numbers will change many times in the next few months. But the value of calculating a budget number, even if only 80% accurate, is worth a lot at this stage.

The Structure and Role of the Steering Committee

Together, we have looked at quite a few planning activities thus far. The SAP Steering Committee has driven some of these tasks, and the Project Sponsor specifically has put in his share of long hours by now, too. At this time, it makes sense to describe the Steering Committee's structure as a high-level group tasked with maintaining a focus on creating a high-quality and relevant SAP solution. Primary members of the Steering Committee include the following (see Figure 2.5):

- The Chair (normally the senior executive tasked with making SAP a reality, that is, the Chief Operating Officer or one of his senior appointed delegates).
- A representative of each functional area to benefit from SAP's re-engineering efforts. For example, this might include representatives from Finance, HR, Manufacturing, Logistics, and World-Wide Sales.
- The Project Sponsor, if not already identified earlier.

- A senior representative of the Chief Information Officer, or the CIO himself.
- The company-internal Project Manager (as opposed to other PMs who will be appointed from consulting and integration partners and SAP itself).
- Manager or Director of Enterprise Computing Systems (or equivalent title, usually responsible for the systems currently in place that will be augmented or retired by the addition of SAP).
- A senior-level SAP Solution Architect, or sometimes SAP's appointed Project manager (someone who can act as the committee's technical liaison). We refer to this position generically as the SA.

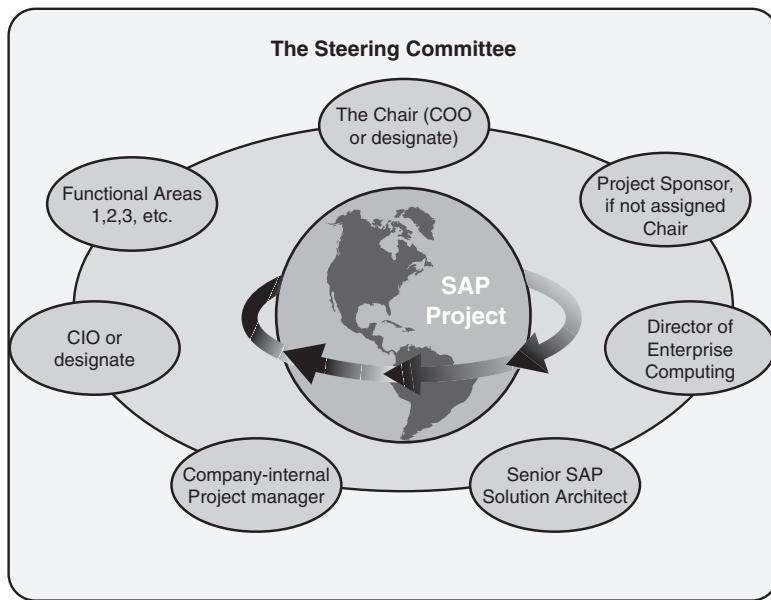


FIGURE 2.5 The make-up of the SAP Project Steering Committee should reflect something like the example illustrated here.

As I said before, the primary role of the Steering Committee is to focus the efforts of the company with regard to how SAP will solve problems of a business nature. It is responsible for setting the scope, time, budget, ROI expectations, and general boundaries for the project. Thus, the committee will be out of necessity heavy with business representatives, who will be required to make decisions that are driven by the nature of SAP and the cost/benefit/capability tradeoffs of technology. Making these decisions will require understanding the impact of the information presented; to make good decisions, the committee will need to familiarize themselves with SAP's

disciplined and thorough approach to designing business processes. Very early in the process, the committee will need to validate decisions made in regard to the very structure of the business hierarchy as represented within SAP, too, including the mapping of business functions against SAP business objects. For example, every company consists of multiple business units, but how will these units be represented in mySAP? As separate clients, or legal entities? As different sales organizations? Through the implementation of completely different SAP instances? The choices, and the impact of the choices, impact the very fabric of the solution, underscoring the fact that the committee's business representation is vital to the project's success.

But to keep the Steering Committee grounded from a technology perspective, specific IT-related representatives are required as well.

In practice, I tend to see a separate smaller subcommittee or team that coordinates the work "handed off" to the Solution Architect or SAP Project Manager, thus allowing these critical resources the bandwidth to focus on strategic committee-relevant issues rather than low-level technical questions. The smaller technology team reports their findings and observations to the SA and/or PM on a regular basis, who then reports back to the Steering Committee as required.

With the structure and role of the Steering Committee explained, let's now move into describing some of the committee's and Project Sponsor's more important activities.

Business Unit Buy-in

With buy-in achieved at executive and senior management levels, the job of the Project Sponsor and Steering Committee is to push this buy-in "down" the organization. In other words, all of the various business and IT organizations need to understand that their full support will be required to successfully implement SAP, and that their managers back them 100% in this regard. Formal and informal meetings and conversations with key contacts in each functional organization, and anyone who plays a key role in designing and supporting the business processes employed by these organizations, are key. I cannot say enough about this—picking up the phone and discussing the project may be good enough for some folks, but this will be the time to really foster and develop relationships face-to-face, too. Success is about embracing change, and then putting in the hours to make it happen. If the business units and their key personnel don't fully embrace the changes that SAP will impose, the implementation will struggle along in two key areas:

- Customization—This is the process of configuring the mySAP components or application modules to align SAP business processes with an organization's workflow processes. Normally this is an adaptive exercise in which the organization's workflow actually *changes* to take advantage of the built-in methods embodied by SAP. But without an organizational subteam in each business area

anxious to assist in deploying SAP, even the best functional consultants and experienced programmers will be at a loss to get this done right, much less quickly.

- Day-to-day business—The pure investment in terms of hours will never materialize in organizations not completely on board with the SAP project. Not only will the project falter in these business units, but accomplishing day-to-day business will be more difficult as well. Why? Because an organization that is not embracing the changes SAP brings with it will instead be fighting hard to keep these changes at bay, and that fight will consume a great deal of time and energy otherwise expended in achieving the company's operational goals.

The “Napkin” Plan—Identifying Major Milestones

While you are developing buy-in of the various business units, you may also begin assembling the pieces of what will eventually become your *SAP Implementation (or Infrastructure) Project Plan* (SIPP). My team has developed so many of these that even our preliminary *Napkin Plan*—the initial high-level plan named for the fact that often it's tossed together by the Steering Committee on something as unlikely as a napkin—is in reality quite comprehensive.

At this stage, the plan is focused on big-ticket items and questions, such as

- When will we nail down the SAP component/solution set that will meet the business's needs? In other words, which mySAP components are we actually implementing?
- When is the right time to share what we know and have, and invite SAP Solution Stack Vendors to either participate in the RFI/RFP process or begin completing their SAP sizing questionnaires?
- When can we get started putting together the SAP Technical Support Organization (SAP TSO)?
- How will we address training our SAP technical staff? What about our development staff?
- How much lead time do we need to fashion an SAP Data Center, and what is on the critical path?
- How soon can we get a development environment and hopefully a technical sandbox in place? What other requirements or tasks need to be addressed first?
- How will we move real data into SAP, to begin the process of configuring and refining business processes?

- When can we begin the Testing phase? Staging? When do we bring in the final Production gear?
- How and when do we perform integration testing?
- Should we run a stress-test or load test? When? How?
- What day do we think we can actually go “live” on our new mySAP solution?
- Do we need to run the old systems concurrently for some period of time? Will we phase in different plants or facilities over time, or will we cut-over to the new system all at once (a *Big Bang* approach)?

All of these questions and more are addressed in greater detail throughout the book. For now, you want to lay out the project plan, and see what kind of time frames you’re looking at. Then, you want to do your best to work “backwards” from Go-Live, so that you can better identify the milestones and tasks that lie on the critical path to Production. This process of working backwards from Go-Live to Planning is covered next.

Working Backwards to Develop Timelines

It’s advantageous to work “backwards,” especially when hard deadlines exist in regard to specific milestones. I often run into hard deadlines around getting plugged into the SAP project management team, crafting/designing the SAP Data Center, deploying the SAP Development environment(s), and getting our Solution Architects engaged. In many cases, I am also engaged to help drive the sizing process with our clients, working directly with our own or other SAP Solution Stack vendors and partners, including their *SAP Competency Centers* (which are organizations within a company dedicated to learning and embracing SAP design, installation, support, and other best practices). When to start technical and incremental “new product” training can also represent hard deadlines for organizations completely unfamiliar with these new technology enablers.

If the time needed to complete each task is understood, and intertask dependencies are clear, it is a simple matter to plug in the numbers and arrive at a critical-path-oriented schedule. With this, you can then look forward again, building and identifying slack-time, concurrent activities, and so on into your Napkin Plan, in effect building a full-fledged SAP Implementation Project Plan along the way.

A Closer Look at the SAP Solution Stack

As was covered earlier, the SAP Solution Stack represents all of the technology layers culminating in a productive SAP installation. But the solution stack applies to each system, and indeed every computer or other piece of gear found in the SAP Data

Center. So the solution stack not only applies to the Production SAP system—the one deployed to actually service the needs of the company—but it also applies to all of the supporting systems that make the Production system possible.

Maintaining a bunch of SAP systems to support a single and perhaps small Production system may seem like a lot of trouble and expense to go to. After all, each system will require its own hardware, software, and so on—each is a full-blown SAP solution in and of itself.

The SAP System Landscape

Before we get ahead of ourselves, then, let us drill down into the specific purpose or roles that each of these systems will play from a change management perspective. As illustrated in Figure 2.6, I have noted both business-process-related changes and technology-driven changes, with the understanding that all of these systems ultimately support and help to maintain a well-performing Production system, but not all are required for every different mySAP implementation.

- Technical Sandbox System—This system is reserved for use by any present or future member of the *SAP technical support team* to practice and perfect configuring and tuning the SAP Solution Stack, especially in regards to software component installations, upgrades, integration with other solution components, setup/testing of data replication, backup and restore processes, high-availability hands-on training, and so on. In the best of worlds, then, it should be identical to the Production system from a topology perspective (same components), though to save money it does not necessarily have to be configured as robustly (fewer number of drives, processors, application servers, and so on may be acceptable). Note that because of its technical support role, it is not involved with development activities per se. Development is initiated in the next system discussed.
- Business Sandbox—Similar in scope to the Technical Sandbox, but used by the *Development team* (SAP ABAP, HTML, and Java programmers) in support of learning, testing, and practicing their trade.
- Development System—This system is created and maintained for continued SAP configuration and/or customization, maintenance, and steady-state updates/bug fixes. This instance also serves as the originator of business-process-related configuration and customization changes that will eventually be “promoted” into Production.
- Test/QA System (also known as the “Quality Assurance” System or the “Integration System” in some circles)—One of the most common systems seen in even the smallest of implementations, Test/QA is maintained for integration and testing of *business process* configuration changes and so on, prior to eventually promoting these changes into the Production SAP system. In other words,

all functional changes are promoted from Development here, and thoroughly tested to ensure that neither it nor other business processes “break” as a result of a configuration change. It should also be noted that *technical* changes are made here first if, for example, a Technical Sandbox is not in place and the Development system is deemed too critical to initiate changes that may impact availability.

- Training System—Usually reserved for larger implementations, this system is maintained for ongoing internal training of SAP end-user personnel (that is, “How to use the SAP GUI,” SAP 101 classes, functional business-area training, and so on). In smaller implementations, this role is often served by the Test/QA system.
- Staging System—Usually identical to Production, this system is used as the last stop for changes in the largest or most mission-critical of implementations. The Staging system is often subjected to stress/load tests and other performance tests that reflect what is expected to be seen on the Production system, so as to determine the probable impact of a change in the actual production environment *before* this change is promoted to Production.
- Production System—This system supports the business groups and provides for the business needs addressed by SAP. It is the system the end users leverage in their daily activities after Go-Live, the reason why SAP was implemented in the first place.
- Disaster Recovery (DR) System—This system is implemented when the cost of unplanned downtime exceeds the cost of implementing, maintaining, and supporting a copy of Production. The Disaster Recovery system (which is therefore usually identical or nearly so to the Production system) is located in a different physical location from the Production system, and used “in case” of a disaster, that is, when the production system fails for an extended period of time, or is otherwise unavailable. Note how business process changes are applied to the DR system—not from development, but typically from Production itself, in the form of a replicated database or replicated transactions applied to the DR system’s database on a regular basis.

Many companies adopt a three-system or four-system strategy in regards to SAP, deploying Development, Test/QA, and Production, along with a combination DR/Staging system or Technical Sandbox. In fact, one of my customers refers to their Technical Sandbox as their “best-kept secret” when it comes to maintaining a *highly available* production operating environment. That is, changes are introduced in the Production system only after testing them in their nearly identical sandbox environment, and then further ratified via the development system. And the entire team at this customer site—SAP Basis, DBAs, even Computer Operations—is very familiar

with the solution stack as they have implemented it, because of their technical sandbox. Further, new-hires and others with the need to familiarize themselves with how the Production system's high-availability cluster operates can do so safely in their hands-on Technical Sandbox rather than trying to schedule Production down-time.

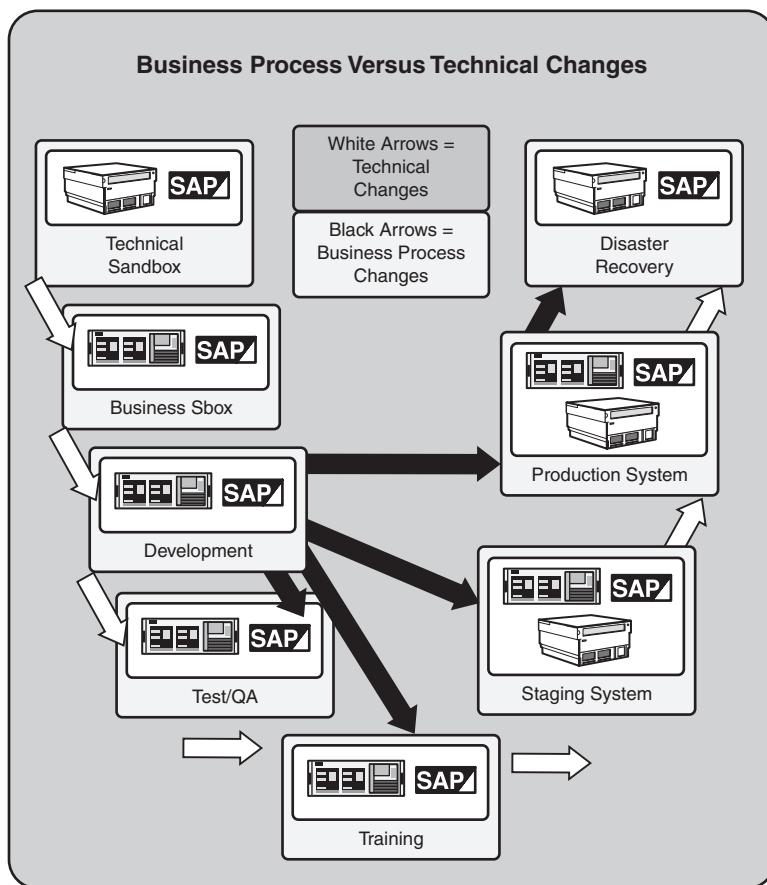


FIGURE 2.6 Changes originate from different systems, and are promulgated differently, based in large part on whether the change is business-process-driven or technology-driven.

Two General Approaches to SAP Solution Sizing

Although the real work of solution sizing is discussed in Chapter 7, the Steering Committee's IT liaison should begin giving thought to how the solution will be presented to the various hardware vendors anxious to compete for their share of the

SAP project dollars. Two schools of thought exist in regard to configuring servers, server components, and computing appliances. In the first case—sometimes called *Future Growth Strategy* or *In the Box Growth*—CPU, memory, disk, and peripheral I/O slot system configurations are designed to meet a minimal supported level of fault tolerance while allowing for future growth of the systems. That is, in server configurations where the hardware needed to meet the user requirements would require filling up or using *all* disk drive bays, or *all* memory slots, or *all* CPU sockets of the selected computing platform, a larger disk drive, higher-density DIMM kit, and so on is used in its place. In this way, fewer CPUs or disk drives are required to provide the same level of capability—and the larger kit ensures that there is built-in “upgrade” capacity within the platform to permit future hardware upgrades, because extra DIMM slots of disk drive bays are left unused and therefore available. Such an approach to configuring allows for “built-in capacity growth” of the individual servers and appliances within a given solution—nothing is “maxed out” from the beginning, so each hardware component can therefore be upgraded in the future.

A second school of thought for configuring individual servers and appliances within a solution suggests that a *Maximizing* strategy allows for lower total cost of ownership. Rather than leaving memory slots, CPU sockets, drive bays, and so on available—and perhaps wasting this potential bandwidth in the years to come before the solution is retired—defenders of the *Maximizing* strategy seek to provide maximum capability in what amounts to a fewer number of servers or disk subsystems. They only pay for what they believe they need. In addition, proponents of this school of thought believe that the typical cost associated with “opening a box”—planning and taking down a server or other hardware component in the SAP Solution Stack, to carry out hardware upgrades—exceeds the delta in cost between configuring it for future growth and “maxing it out.”

Neither school of thought seeks to sacrifice availability at the expense of anything else, however. Availability through redundancy—maintaining “2x” or “1+x” servers when only “x” is required to address a given load—still applies completely to both schools of thought. In the first case, though, three lightly configured servers may be specified, whereas in the latter, two heavily configured servers are preferred.

Different companies and their IT organizations tend to subscribe to one or the other school of thought. In the last few years as hardware costs continue to drop, it seems that more and more of our clients go with the second approach. Experience has also shown many of these organizations that the downtime associated with bringing a server offline for a hardware upgrade is often not worth their effort. Regardless of the approach, though, this needs to be documented and later shared with all of the prospective hardware vendors.

Giving Attention to Change Control

The complexity of an SAP implementation can really create problems in this critical area—not embracing good change control practices. A system-wide approach to managing change must be developed and followed, and it needs to start here and now. There are too many examples of missed SAP Go-Live dates, and too many botched implementations, to ignore change management.

Ultimately, change control (or change management—the terms are used interchangeably) minimizes system downtime while still allowing the system to benefit from enhanced business functionality or increased future stability due to “bug fixes” and other software/firmware patches. One of our SAP clients describes change control as “staying behind the curve enough to keep the system up.” It’s a good definition, actually—if a company decides not to jump on the latest and greatest service packs or bug fixes or SAP support packages in their production environment, instead opting to test these changes elsewhere in a methodical manner, that company will enjoy greater system availability.

The Role of the SAP Technical Support Organization

With a project plan in development, and progress being made toward a general solutions architecture, you can now turn your attention to considering the organization tasked with bringing the technology together, and ultimately managing it—the *SAP Technical Support Organization*, or SAP TSO. A number of forces drive the size and scope of the organization tasked with designing, building, and supporting an SAP implementation. Indeed, the support organization will revise itself shortly before, and again after, the Production system goes “live,” but for our purposes, we will focus on the following “pre-go-live” requirements that drive how and when the SAP TSO is staffed:

- Functional or “capability” requirements
- Accessibility requirements
- Availability/high-availability requirements
- Performance requirements
- Scalability requirements
- Security requirements
- Administration requirements
- Other requirements

We will drill down further into each of these requirements in different chapters throughout this book, too, as the role of the SAP TSO morphs and changes. In

particular, Chapter 4 provides insight into designing the TSO, whereas Chapters 8 and 12 detail when and how to staff it.

Moving Beyond the "Napkin" Plan

With your Napkin Plan starting to grow and fill out, and timelines coming together, it makes sense to begin formalizing all of this into the SAP Project Plan (if you have not already done so). Throughout this book, I reference what I mentioned earlier—the SIPP, or SAP Implementation Project Plan. A copy of a sample SIPP may be found on the Planning CD.

How to Begin Measuring Progress and Success

Tracking, measuring, and charting progress is all accomplished via the SIPP. It is the Project Plan that becomes the central vehicle for tracking changes to the plan of record, too. That is, as newer versions of the SIPP emerge (due to inevitable delays, the impact of authorized “scope creep,” changes or challenges with regard to staffing, and so on), the old project plans serve as documented point-in-time project snapshots.

To help keep the SIPP up to date, my team and I recommend daily, focused 30-minute *daily tasks* project-team meetings, and comprehensive weekly *status update* meetings. The daily meetings are best hosted in the morning, that is, 7:30–8:00 a.m. or sooner, and the weekly meetings seem most effective Monday mornings (replacing the daily meeting) from 7:30–9:00. These times tend to work well for a number of reasons:

- The project team is “fresh.”
- It tends to force individuals to show up (whether in person or via conference call) on time.
- In the case of global implementations, mornings throughout much of North America allow for Europe to attend the meeting/conference call as well.
- It leaves a contiguous block of time—the rest of the day—for work to proceed uninterrupted by status meetings.

To keep the meeting moving along, one of my colleagues likes to hold his daily meetings in conference rooms without chairs. Others leverage a regular daily and weekly format, or a running list of “minutes” and action items from the previous meeting, to stay focused. Chit-chat is frowned upon in all cases until after the meeting is wrapped up.

Evenings are generally bad times for meetings, as the project team will often be “in the middle” of something late in the day, and their time is best spent focused on the

task at hand rather than breaking away for 30 minutes at the risk of losing momentum. Fridays tend to be bad for meetings for the same reason, regardless of how tempting it might be to the Steering Committee or Project Sponsor to have an update before the weekend starts.

The key is to remain flexible, though. For example, one client of ours started with daily meetings and then made a change mid-way through the project. Instead, they hosted their daily meetings Monday, Wednesday, and Friday during lunch, and were quite successful. Why? Because they brought in lunch, and stretched the meeting to an hour. In this way, everyone not only had a chance to share status updates, but everyone had enough time to eat (for free!) as well.

Another client chose to implement meetings from 4:00–4:30 p.m. each day. This time slot was met with grumbling initially, but within a few weeks it became “business as usual” to meet during this time period. It worked for them—they got in and out, with little apparent interruption to the real work of implementing SAP.

Maintaining the High-Level Project Plan

Now that you are in a position to measure progress, track successes, gather updates, and so on, you can begin to effectively maintain your formal (albeit still high level at this point) project plan. Key milestones inherent to all SAP implementations have already been documented in the SIPP. For example, all SAP projects require a development and a production environment, each of which must be designed, purchased, installed, configured, optimized, and deployed. And all SAP projects must take into account a client strategy, a cut-over strategy, and address things like performing the Basis installs, stocking the new systems with data, testing the new business processes in terms of integration with other business processes, and so on.

Additional tasks will become more clear as we move through the next few chapters. For instance, the system landscape requirements of the specific implementation will become apparent, as will timelines for the various implementation tasks to be executed, and much more. We will continue to fill in some of the project plan “holes” that exist with regard to your specific implementation as we move through each chapter together.

Tools and Techniques

With regard to this chapter, the Planning CD contains:

- My sample SAP Implementation Project Plan (SIPP) in Microsoft MPP format.
- A presentation introducing SAP to the Steering Committee, usually customized ad given by the Solution Architect or other technology representative.
- Tools and links for measuring Return on Investment (ROI).

- A sample “Solution Characteristics Matrix,” to be completed by all of the appropriate stakeholders and Steering Committee members so that each has a say in how important they believe every solution characteristic to be to the business.
- Each of the figures found in this chapter, in Microsoft PowerPoint format.

Summary

We have covered quite a bit of material in this chapter, much of which will be explained in further detail throughout the remainder of the book. I started off with tactical and strategic reasons for implementing SAP in the first place, wrapping up the first few sections with real-world stories of success and failure. Then we looked at the tasks and activities that tend to keep the Steering Committee and Project Sponsor busy in the first few weeks after the decision is made to implement SAP. This included high-level critical tasks like promoting buy-in, nailing down the real business requirements to be satisfied by SAP, determining realistic SLAs, an exercise in estimating ROI early in the project, translating business requirements into technology drivers, nailing down something of an initial budget, and more.

Then I spent some time analyzing the SAP Solution Stack, discussed the workings of an SAP system landscape, and covered basic tenets of solution sizing, all of which need to be understood at some point to take us to the next level. A discussion on change control, the emerging role of the SAP TSO, and assembling the high-level project plan wrapped up this chapter.

In the next chapter, we will build upon what we discussed here and address the “SAP Solution Vision,” where the general technical requirements are fleshed out to reflect actual necessary SAP components and products, including factors that influence the design of your SAP system landscape. And I will devote quite a bit of time analyzing outsourcing as an alternative to hosting your SAP solution in-house.

3

Crafting Your mySAP Solution Vision

What Is a Solution Vision?

With the understanding you have gained in terms of what an SAP project looks like, we are ready to begin refining and later communicating a vision of the future-state of your SAP solution, a *solution vision*. Think of this as the “eyes closed” phase of the project, where wishful thinking is tempered by constraints like budget and headcount realities to sketch a design that meets both business and financial requirements, as you see in Figure 3.1.

In the narrow role I have personally played helping my customers to craft such a vision, I give the following advice to executive and senior IT decision makers, and other members of the SAP project steering committee:

- Generally, focus on your core business and how an enterprise solution will better enable that core business to be successful.
- Identify the shortcomings of the systems and processes in place today. For example, is it difficult, expensive, or cumbersome to customize the system? Are employees forced to duplicate entries in multiple systems, or access different systems for different customers? Is the system subject to downtime because of hardware and other solution stack issues?
- Clearly define the value that you believe those systems should provide to the business. That is, should the system be available 24×7? Should it be accessible over the Internet or your company

IN THIS CHAPTER

- What Is a Solution Vision?
- SAP System Landscape Design and Planning Approaches
- Capturing Your mySAP Solution Vision
- Outsourcing—Another Method of Achieving Your Solution Vision
- Tools and Techniques

intranet? Should it tie together different functional and business areas, or enable real-time decision making?

- With the data collected in response to the questions in the previous two bullet points and with your real business requirements nailed down in Chapter 2, step outside of the “box” and consider alternatives and “nice-to-haves.” Enlist the assistance of your own long-time end users and the insight of your current IT staff to begin assembling a new solution vision that describes what the system *should* be capable of.
- Solicit the advice of mySAP.com experts to assist you in identifying real-world product and technology constraints, thereby helping you to refine and document the *characteristics and capabilities* of a mySAP solution that can be customized and implemented for *your* company in support of *your* business objectives.

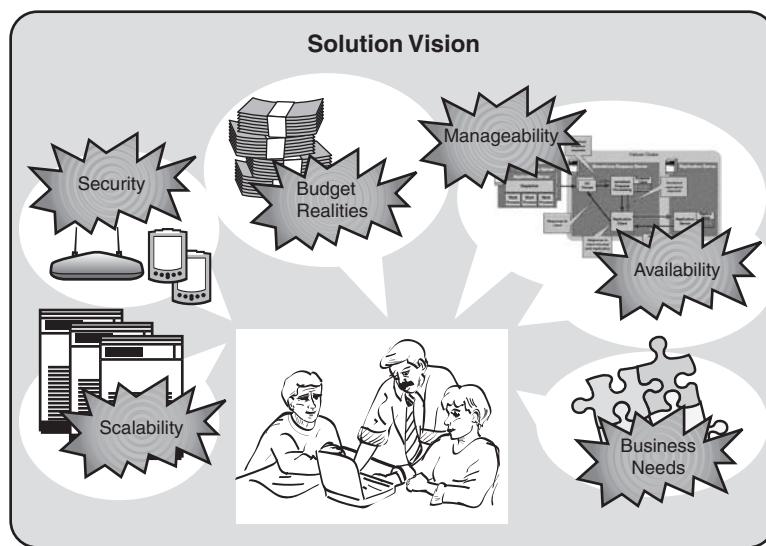


FIGURE 3.1 The solution vision melds company needs and constraints into an achievable business vision facilitated by technology.

Regarding this last point, mySAP.com experts can be enlisted from many places. I suggest creating a focus team of pre-sales consultants from SAP AG itself, from a Big 4 or other capable implementation partner, and from an enterprise hardware and services partner. In this way, most of the SAP Solution Stack is represented while still minimizing the number of players.

After the vision is initially captured, disseminate it in draft form so as to begin a *review process* of sorts—share it with stakeholders, like senior members of the business groups who will use the solution in their day-to-day dealings, the customer-facing groups who will be positioned to better serve your customers, and the Information Technology professionals who will be tasked with supporting the solution. Formally gather and document all of this feedback, so as to better revise and refine your solution vision. To this end, I recommend updating the Solutions Characteristics Matrix first discussed in Chapter 2. Ensure that senior and executive-level management concur with the vision as it evolves, and that buy-in is achieved at all levels of your organization. Only after all of this is accomplished can the real work of planning how to actually “get there” commence.

Impact on the Business

As the business groups begin sharing their thoughts and insight regarding the solution vision, keep in mind the following:

- Business processes will almost certainly have changed with the introduction of your new mySAP system, typically reflecting tighter integration and best practices.
- Therefore, employee roles will change, and jobs may potentially be at stake.
- Finally, the tools and interfaces used by each employee in the normal course of their job will change.

Back to the solution review process—as different folks inevitably demonstrate resistance to the project, consider the points in the preceding list, especially whether their jobs are impacted and to what degree. And just as importantly, consider each individual’s personal resistance to change. These two factors represent the key rationale behind exposing only senior members of the different business and functional groups to the new solution vision. With your senior and loyal employees on board and embracing potential changes as their own, you will be positioned as best you can against pockets of resistance lower in the organization.

Technology Perspective

Before specific software packages and hardware components are purchased, or services contracts are signed, a company must come to grips with its *technology perspective*, which is simply how it views its investment in information technology resources. Why? Because this shapes the architecture, or the very foundation, of a computing solution. Some companies look at IT spending from a long-term perspective, and try to purchase assets with a useful life of perhaps many years. Other companies subscribe to the belief that regular hardware and software refreshes will provide a competitive advantage or a performance advantage over time. Still others

seek to stay on one side of the spectrum or the other, investing conservatively in time-proven solution stack components, or on the other hand investing in the latest and greatest high-availability and performance offerings. And finally, others prefer to outsource technology and its requisite support structure. I like to understand how a company thinks in this regard before attempting to architect a mySAP hardware and software solution; it is important for everyone to understand how risk tends to increase as investments in new technology increase, too, promising greater potential reward in exchange. This is illustrated in Figure 3.2 and detailed in the following different technology perspectives:

- Conservative—As the least risky of all approaches, a company that has a conservative technology perspective places availability above all else. They seek mature technology, mature practices, and tried-and-true solutions that *work*, day in and day out. What they potentially sacrifice, then (though in their eyes this is not a sacrifice at all), is anything new—new approaches to accessing their system, new methods of improving availability or manageability, new solution architectures, and so on.
- Mainstream—Like their conservative brethren, these companies prefer established platforms and products to newer ones. However, the key word here is “company”—they want to have a lot of company when it comes to how they solve their business problems through the use of IT resources. Mainstream companies want to be able to point to a slew of other companies and feel confident that they are not alone, that most of the industry is doing things in a manner similar to theirs.
- Close follower—My favorite companies to work with tend to be close followers. They seem to leverage their IT investments in proven technology, but with exceptions. That is, although maximizing uptime always remains central, close followers are unafraid to try a few new things to gain a competitive advantage or otherwise position themselves better for the future. Therefore, they take an occasional calculated risk and invest in new products, new technologies, and new approaches.
- Leading edge—This is the riskiest of all approaches, hence the more popular label “bleeding edge” assigned to this technology perspective. A leading-edge approach places more value on competitive positioning than anything else—it’s all about getting a jump on the competition in terms of minimizing cost, reducing downtime (through recent technology advances), increasing response times of customer-driven business transactions, maximizing accessibility (for example, through Internet-based vendor/partner access to your order-status system), and so on. Therefore, a leading-edge company must be prepared to spend much more time managing change, as they tend to introduce new products and approaches without the benefit of a “history.” In fact, because of this, leading-edge companies are the same ones that tend to find and work through technology problems first.

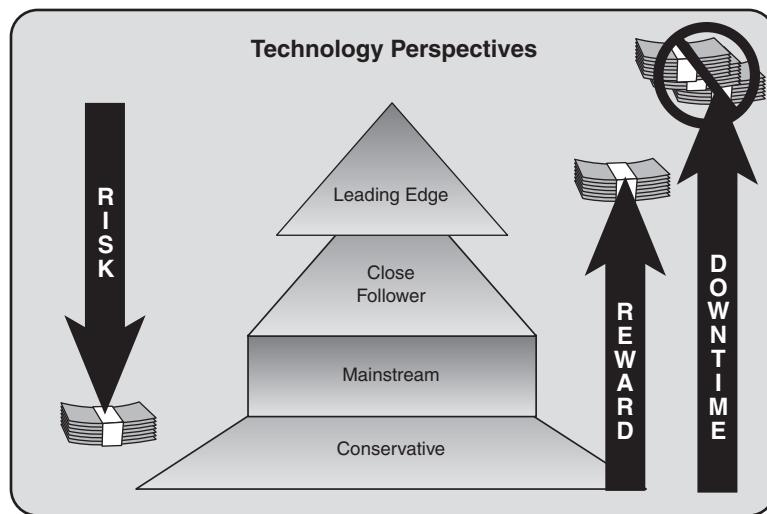


FIGURE 3.2 These four key technology perspectives illustrate how greater risks are related to potential reward as well as increased downtime.

When your technology perspective is clear, we can start looking at individual solution components and how all this fits together to create a custom system landscape for each particular solution. Note that I did not include outsourcing in the preceding list—the topic of outsourcing as a technology perspective is covered in the last section of this chapter.

Considering mySAP Components to Be Implemented

As the different business requirements are hammered out and in turn mapped to the solution vision, inevitably a discussion around various mySAP offerings emerges. Take care to distinguish between *current* mySAP component capabilities, and new features that will soon be released in new versions of a particular mySAP solution. Over the last few years, SAP has aggressively released new versions of current mySAP components, re-badged existing components and technologies, and added quite a few *new* components. So as you begin discussing specific solutions like Advanced Planner and Optimizer, SRM E-Procurement, or Enterprise Portal and the SAP Exchange Infrastructure, it is very important to bring in an expert versed in both the solution's current capabilities and shortcomings, and what lies ahead on the road map.

Considering SAP System Landscape Requirements

As with the mySAP components to be implemented, it's also important to determine the SAP system landscape requirements necessary to achieve your solution vision:

- Do you need a formal training system for end users?
- What about your IT staff—will a Technical Sandbox be required to help them gain a certain comfort level with new technology?
- Will your functional and development/programming team need a Business Sandbox with which to learn and test?
- Will a dedicated load-testing system need to be maintained that is identical to the Production system?

All these questions must be answered soon. This is why figuring out details related to your SAP system landscape plays such a big part in this chapter. In essence, though, evaluating the following will help you answer SAP landscape-specific questions as we move forward:

- The relative strength or weakness of an organization often determines whether an SAP system landscape component is warranted. For example, a “weak” IT team—a team uneducated or unfamiliar with a particular technology platform—will benefit greatly from a Technical Sandbox. Similarly, a development team less than familiar with a unique mySAP component/development tool combination will require a Business Sandbox.
- High availability drives much of the SAP system landscape design, too. The original “SAP 3-System Landscape” discussed in many books and articles over the years evolved out of the need for improved availability, for instance. But your particular needs may drive the creation of a more robust architecture where additional testing is possible.
- The ability to recover quickly from a disaster drives the creation of a Disaster Recovery system. The term “quickly” is relative of course, but a backup tape-based restore performed on a newly installed hardware platform usually represents a worst-case baseline.
- If performance is critical, adding a Staging system to a Development/Test/Production landscape can provide the resources necessary for load-testing or stress-testing changes prior to implementing them in Production (or prior to a change management package or “wave” being promoted to Production).
- If the idea of *simplification* is important to you, there are strategies and approaches designed to do just that—simplify your SAP system landscape.

Other factors like critical security concerns, the ability to manage a particular solution, and so on will drive the adoption of incremental systems, too. All these factors and characteristics are discussed in detail in the next section.

SAP System Landscape Design and Planning Approaches

Remember, a system landscape exists for each mySAP solution—if you deploy R/3, APO, CRM, and PLM, you will in effect be creating four different SAP system landscapes, one for each product. The focus in this and the following few sections, though, is on what *one* of these system landscapes looks like from a design and planning perspective, for example R/3 alone or APO alone.

In the most general form, an SAP system landscape consists of SAP instances (installations of the SAP database and application software) and SAP servers. In the Microsoft world of SAP implementations, there is a one-to-one correlation between instances and servers nearly all the time. That is, the Development instance resides on a dedicated Development server, the Test instance resides on a dedicated Test server, and so on. In the world of UNIX implementations, though, multiple instances can be often found on a single “larger” server. For example, both Development and Test instances can reside on a single server. And multiple application instances can be installed on a single server as well.

Until last year’s release of SAP’s *Multiple Components, One Database* (MCOD) initiative, there was a one-to-one correlation between instances and database systems, too, regardless of the Operating System platform. MCOD is beginning to change this, such that a single “larger” database can be leveraged for multiple instances. However, an important difference between MCOD and multiple instances/one server exists—MCOD ties the same type of databases within *different* SAP system landscapes together. With MCOD, all Development databases used by your R/3, SRM, CRM, and Workplace implementations can be one and the same. Similarly, all Test databases across R/3, SRM, CRM, and Workplace can be bundled together, too, as illustrated in Figure 3.3. Note, however, that in many cases SAP AG frowns on mixing OLTP and OLAP systems, or combining different databases within the same system landscape. In that regard, forcing an MCOD database server to host your R/3 system’s Development and Test databases would therefore be unsupported and contrary to best practices.

As we move forward with our basic understanding of SAP system landscapes, and seek to understand how your SAP solution vision impacts and is impacted by your landscape decisions, my hope is to achieve the following:

- Note the relative importance and relationship of technology perspectives to our solution vision
- Understand why each system landscape is important to fulfilling our vision
- Note how the presence or absence of a particular system within a landscape impacts the other systems and ultimately the overall solution vision

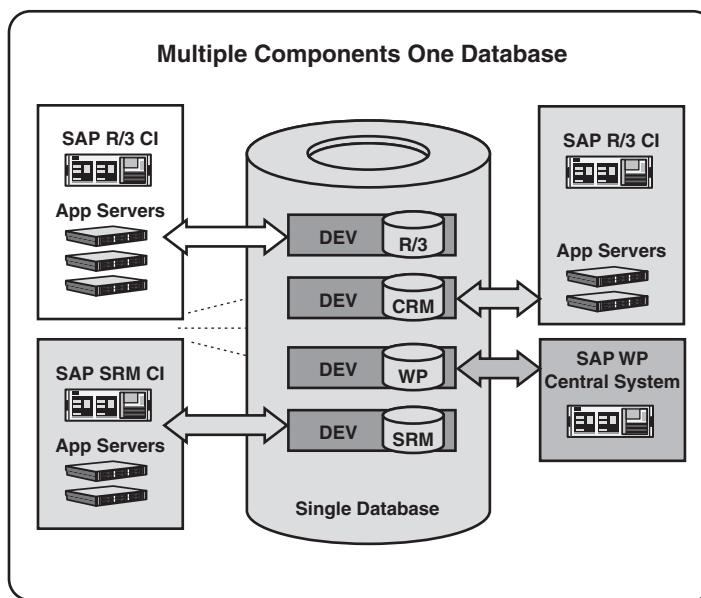


FIGURE 3.3 A properly architected sample MCOD deployment is displayed for a typical mySAP enterprise consisting of R/3, CRM, SRM, and Workplace.

All these design and planning approaches I cover tend to come into play in one manner or another across all mySAP implementations. It's how they are weighted or addressed that makes one system landscape different from the next.

Simplifying Your SAP System Landscape

After spending time with hundreds of customers and SAP implementations, I think it is safe to say that when all things are equal, the desire to *simplify* emerges as an important driver. Simplification takes many forms, too. In the case of the SAP system landscape and how it fulfills our SAP solution vision, the desire to simplify manifests itself in any number of ways:

- First, the pure number of instances will be reduced to the fewest necessary to get the job done “right” for a particular company. An organization focused on simplifying administrative, change management, systems management, operations, and other tasks will deploy a three-system or even a two-system landscape, whereas similar organizations without the same simplification goals can deploy more. There are trade-offs, of course. A system landscape without a dedicated test instance will, for example, be forced to perform testing in the same system used for development. Because of these kinds of limitations, simplification achieved through instance reductions is not as common as it has been in the past.

- Instead, a more popular approach to simplification seeks to reduce the number of physical servers in a particular system landscape, by installing multiple instances on a single server. Consolidation of instances is becoming quite common in SAP customer environments today, as displayed in Figure 3.4.
- Similarly, deploying a shared disk subsystem and tape backup/restore solution also simplifies a very complex piece of the SAP Solution Stack. This is why my colleagues and I have spent so much time in the last two years designing and implementing Storage Area Networks, or SANs—they provide outstanding performance while simultaneously reducing system landscape complexity and allowing expensive resources like enterprise tape libraries to be shared between systems.
- Another customer of mine shared with me why they went with the WebGUI as opposed to the classic SAPGUI approach to system accessibility—to simplify desktop support and maintenance requirements.
- Companies that value simplification will also standardize on a particular solution stack option or approach, too, as this simplifies support and maintenance, and minimizes the need for a variety of onsite/reserved spare parts, the time spent in change management activities, and more.

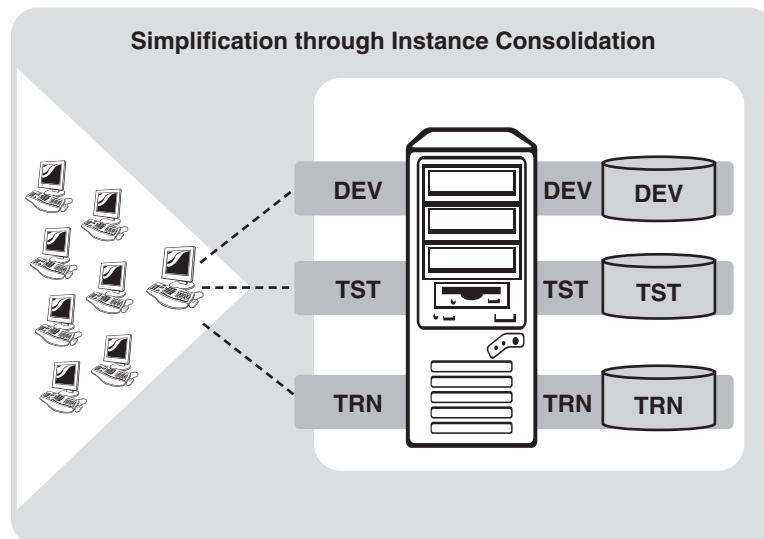


FIGURE 3.4 Multiple SAP instances can be installed and configured on a single physical server, oftentimes reducing both acquisition and systems management costs down the road.

Although simplification tends to work in one direction by encouraging a “do more with less” philosophy, our next topic goes the other route in that it purposefully introduces complexity and differences between various systems within a system landscape—high availability.

High Availability and the SAP System Landscape

When it comes to high availability, many technology professionals automatically think about what it means to improve the availability of a particular system or hardware component—thoughts of basic HA offerings like clustering or redundancy come to mind. With regard to the broader topic of how your solution vision impacts your SAP system landscape, though, high availability equates to the following:

- Business-driven requirements—HA offerings and approaches are normally implemented to satisfy specific business-oriented needs, and therefore form an integral part of your overall SAP solution vision.
- Complexity—HA complicates the SAP system landscape, as HA offerings and approaches tend to only really exist or apply to the Production system and at minimum (hopefully!) another similarly configured system within the landscape.
- Increased support needs—Because HA offerings are inherently complex, a very real need exists to prepare your SAP support organization in how to install it, update it, manage it, and troubleshoot HA issues.

► To read more about how business requirements relate to high availability, see “Availability Planning—Documenting Requirements and Key Drivers,” p. 167 in Chapter 6.

Disaster Recovery Considerations

All companies implement a method of addressing Disaster Recovery (DR), whether or not they actually realize it. Even companies that do not add a dedicated DR system to their system landscape address disaster recovery. That is, their de facto disaster recovery plan simply reflects the challenges and timeframes surrounding rebuilding their SAP system from scratch, restoring from their latest tape backup, and imposing upon their end users to manually rekey any new business transactions lost between the last successful tape backup and the point at which the disaster occurred. This doesn’t sound like much of a “plan,” of course, but it does represent a baseline against which all other disaster recovery approaches and solutions can be weighed.

A host of DR approaches are discussed throughout Chapter 6, from those involving disk subsystem data replication solutions, to various clustering solutions, to database

and mySAP-specific tactics. But when it comes to sifting the potential layout of your SAP system landscape through your solution vision, two general approaches fall out:

- Creating and maintaining a dedicated Disaster Recovery system within your overall system landscape.
- Outsourcing your Disaster Recovery system to an outsourcing provider.

Both approaches are valid, and the first is more traditional. But I believe that the time and expense related to setting up, configuring, keeping current, and managing your own DR system explains the recent increase in outsourcing I've seen over the last two years.

- ▶ To review some of the tasks and considerations inherent to addressing DR internally as opposed to outsourcing it, see "SAP General Availability and DR Best Practices," p. 207 in Chapter 6.

Companies that outsource the DR component of their SAP system landscape help to preserve their data, and access to this data, in that the outsourcer operates a completely independent data center, typically in a very different geographic location. For smaller and mid-size companies with only a single data center, the expense relief is tremendous. On the other hand, if the DR solution is maintained "in-house," so to speak, it will need to be housed in a separate facility. This alone is sure to drive complexity, cost, and even the architecture and makeup of both the SAP system landscape and its individual systems.

Addressing Training Requirements

The SAP system landscape is directly impacted by the potential need to train SAP end users as well as the system's developers and technical support staff. Three different systems come into play, as illustrated in Figure 3.5:

- A dedicated *Training* system is often implemented to assist in teaching users new to a particular mySAP component *how* to actually use the system. This amounts to business-process training as well as SAP user interface training (an excellent alternative to creating multiple training clients on the Test system, which is busy fulfilling integration responsibilities prior to Go-Live—the exact time when end users *need* to be trained!). To provide the most value to its students, the Training system needs to be an exact copy of Production.
- A dedicated *Technical Sandbox* system is extremely useful in helping the SAP Technical Support Organization (SAP TSO) get up to speed on the entire SAP Solution Stack, especially with regard to new components and complex HA offerings (rather than attempting to get time on other systems for what could amount to *crash-and-burn* testing).

- A dedicated *Business Sandbox* or *Development Sandbox* system allows developers unfamiliar with a particular mySAP component, or faced with integrating multiple components and other legacy systems, the opportunity to do so in a pure testing environment (rather than the real Development system).

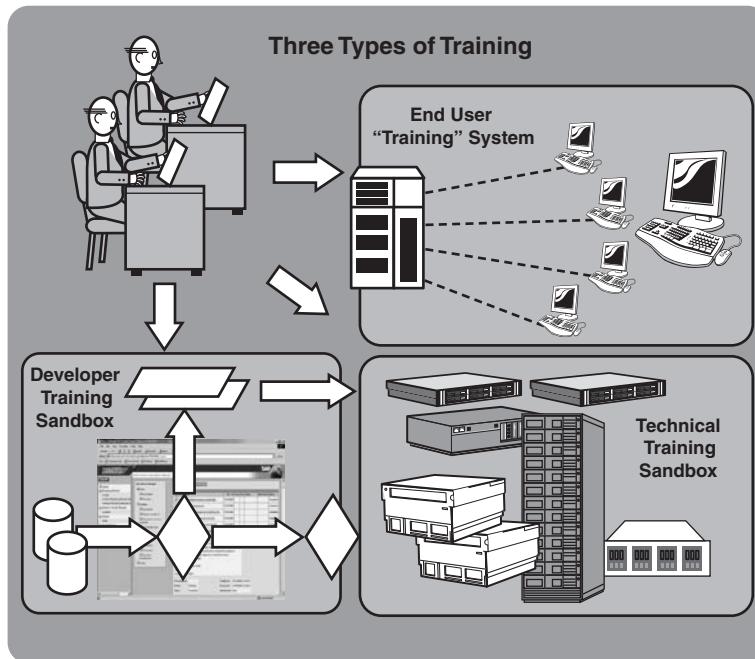


FIGURE 3.5 These SAP training systems support the different needs of different organizations, from end users, to developers and programmers, to technical implementation specialists.

- ▶ For details as to how the SAP system landscape satisfies the training needs of both the SAP Technical Support Organization and the production system's end users, see "Training and the Role of the SAP System Landscape," p. 314 in Chapter 9.

It can represent quite a challenge for the "customers" of one of these training systems to convince everyone that such a system is truly required. In my own experience, I have seen the lack of a Technical Sandbox really hurt an organization in terms of downtime due to botched infrastructure upgrades and changes to DR processes.

Another colleague of mine has more than once had to strongly push for the adoption of a Training system, too. Such a system allows for extensive informal user testing and practice outside of formally delivered training. He believes that this extra

level of hands-on self-directed training is critical because your end-user community is best positioned of *all* groups to find business-process operational errors and limitations. And of course it is desirable to correct these issues well before Go-Live. But a consultant or even a senior super user is typically not positioned to push the adoption and use of a dedicated Training system. More often than not, it takes the SAP Steering Committee, the project's experienced management team, and the prodding of a knowledgeable SAP Solution Architect to do so. I cannot stress this enough—the risk is huge, in that you do not want to find out too late that not every business scenario works as it did during integration testing (for example, all types of contracts, all types of material movements, all kinds of accounting entries, and so on).

The Performance-Driven System Landscape

When it comes to evaluating your solution vision against the layout of your SAP system landscape, it is important to ensure that the performance of the systems meets the needs of their different end-user communities. Most of the time, of course, the focus is on designing, installing, and configuring a well-performing Production system. Performance considerations usually relate back to what an end user will experience while on the system, including

- Business transaction *response times*, or how long it takes to refresh your SAPGUI after pressing the Enter key, for example.
 - How quickly a background or “batch” job will execute, otherwise known as *throughput*.
 - How quickly a report or other query will make it through the system and actually be printed, sometimes called *latency*.
- To read more about verifying that a Production system can meet performance expectations, see “Key SAP Stress-Testing Considerations,” p. 580 in Chapter 16.

However, these same performance considerations apply to all of the other systems within the SAP system landscape, too. The Development system, for example, needs to exhibit excellent performance even while 25 or 50 or more developers are banging away at keyboards trying to build your custom mySAP solution. Similarly, your Test system needs to provide the performance necessary to get through integration testing. Even the Training system needs to provide adequate user response times so as to make the actual training experience more than something to be avoided.

High-performance considerations cover the gamut, touching every facet of every system within the landscape. This means that *everything*—from the performance of the network connecting each system, to each server’s CPU, RAM, and disk configuration, to each system’s OS, database, and mySAP component—must be addressed.

Starting off on the right foot (with properly sized and configured hardware and software elements) is paramount, of course, but tuning all these solution stack pieces to create a cohesive well-running machine is just as important to achieving excellent performance. Like the weakest link in a chain, a single underperforming solution component will only throttle back the maximum performance otherwise obtainable from your system.

Driving Scalability into Your System Landscape

The need for scalability, like high availability and excellent performance, is addressed primarily through the sizing process. Scalability does not pay off up front in terms of improved system availability or better user response times, though. Rather, scalability is all about paying for “headroom” in your system, headroom that is not actually needed at present but might be required in the near future. In other words, scalability addresses future planned and unplanned growth in your system.

This growth can manifest itself in a number of ways. In my experience in the real world, I have seen the results of *unplanned* growth hurt companies where scalability was never addressed, as in the following cases:

- The number of end users increased at one of my new accounts, not due to more hiring than was anticipated when their mySAP.com solution was crafted, but because they unexpectedly acquired their competitor and doubled in size. We had six months to project the delta needed in terms of database and application server processing power and RAM requirements, followed by stress-testing the new design and finally implementing it.
- More than one of my customers' databases grew so fast that they outstripped the results of their comprehensive three-year database sizing methodology in the first year! In most cases, the system we put in place for these customers was scalable—more disk drives could be added, smaller drives could be swapped with larger ones, and so on. In three cases in particular, though, the database growth was so explosive that a whole new disk subsystem platform needed to be brought in, and the recently acquired current platform retired (or redeployed) years earlier than expected.
- When databases grow quickly, the tape backup/restore solution implemented often grows less effective as well. I have seen this most often in relatively small SAP implementations, where an initial investment in tape backup technology needed to be tossed in favor of tape solutions that backed up more data per tape cartridge, and did so fast enough to not exceed the customer's backup window (time allotted to perform a backup, which usually equates to planned downtime in the case of offline full backups).

- It's been a while, but I also had a customer outgrow their network, too. Today, with switched networks and Gigabit Ethernet providing more than adequate bandwidth to every mySAP.com server component, and cheap 10- and 100-Megabit Ethernet prevalent across end-user workstations, there's no excuse for lacking network scalability.

Outstripping the capabilities of your current system such that a new platform is needed probably represents a worst-case scenario. Not only does the current Production component need to be replaced, but to support sound change control principles, so does the same component in your Test, Staging, and/or Technical Sandbox environments.

This is why hardware and software vendors tout things like “highly scalable system architectures,” “enterprise versions” of particular Operating Systems and Database Systems, and so on—though not necessarily needed up front, the headroom that these approaches provide helps an organization feel more comfortable if they wind up growing faster than they expected. And hardware vendors in particular can position their SAP clients for improved scalability by practicing the following:

- Specify server platforms that allow additional CPUs and RAM to be added as needed. In other words, avoid “maxing out” the box.
- Alternatively, design SAP solutions such that they take advantage of SAP’s support for *horizontal scalability*. This is one of my favorite approaches when it comes to SAP Application, Web AS, J2EE middleware, and ITS servers—I prefer to max them out with regard to processors, with the understanding that an incremental number of servers can be added at any time should the environment grow to require it (interestingly, although SAP has successfully tested a system running more than 160 application servers, it is rare to find customer implementations with more than 10 or 12).
- Architect a solution for the appropriate level of vertical scalability. In other words, if a two-tier “Central System” (where all SAP software components execute on the same physical server) approach to sizing meets today’s requirements, perhaps a three-tier solution will provide for unknown scalability requirements. In a three-tiered architecture, one database server and multiple application servers are configured as a single system image.
- Architect a highly scalable database platform. As my real-world examples earlier in this list illustrate, this tends to be where a lack of scalability causes the most problems.

Hardware and software vendors alike spend a great deal of time “proving” how scalable their offerings are. As a first step, I suggest that benchmarks, customer references/feedback, and the results of tests published through white papers and other

technical documents be reviewed by prospective mySAP customers. I also suggest that you begin considering new approaches to scalability. For example, HP's iCOD offering touts "capacity on demand." When a customer buys a server, for instance, it is fully populated with CPUs. The customer pays for only what is needed in the near term, however. Later, if it is determined that more processing power is required, the customer takes advantage of the in-place processors by merely applying for a license; no intrusive field upgrade or service call is required and therefore the need for planned downtime is drastically reduced.

The TCO-Driven System Landscape

More than anything else, Total Cost of Ownership (TCO) drives what a solution vision actually looks like at the end of the day, when a mySAP solution is implemented and really being used. Discussions on TCO might instead be labeled Return on Investment (ROI), or might fall under the heading of investment protection. Regardless, a focus on lowering TCO seeks to find less expensive solution-stack alternatives that still meet the needs of the business.

- ▶ To read more about the relationship between TCO and your SAP solution vision, see "How the SAP Solution Vision Drives TCO," p. 127 in Chapter 5.

When all other things are equal, the following points apply from a hardware perspective:

- A hardware vendor's use of common components like CPUs and memory boards allows flexible sharing of resources between different SAP system landscapes and in some cases hardware platforms, too.
- Similarly, common disk drive form factors reduce cost of ownership by increasing reusability.
- Support for hot-pluggable and/or hot-add hardware components eliminates or worst-case minimizes downtime (can include hard drives, tape drives, power supplies, fans, and even RAM and processors).
- Support for redundant components, like power supplies, disk drives, fans, and so on, also eliminates or minimizes downtime.
- The ability to run mixed-speed CPUs or RAM in a particular platform protects that investment—CPUs and RAM do not have to be tossed aside when additional processing power or memory is required.

Outsourcing your entire SAP infrastructure/operations team is another potential method of reducing TCO. In fact, outsourcing can represent the biggest potential TCO factor that a company will consider. At this level, though, outsourcing becomes more of a strategic business solution that impacts a lot more than simply TCO. True,

outsourcing can cut labor costs by 50%, and enhance flexibility of a technical support organization to easily change as business requirements change, but there are drawbacks and disadvantages as well (discussed later in this chapter).

Another solution vision approach that impacts the SAP system landscape from both a configuration and TCO perspective is the use of an *Application Service Provider* (ASP). ASPs can drive lower TCO by virtue of their application-specific expertise, above and beyond that provided by in-house staff and traditional outsourcing providers. For example:

- An ASP can offer a preconfigured solution stack for the particular mySAP solution you want to implement. This is one reason why they look so good from a TCO perspective—design, deployment, manageability, operations, and other cost factors are substantially reduced due to a high level of both standardization and core competencies in the services they provide.
- ASPs were more or less born out of the dot-com era, and by virtue of this, their data centers enjoy the benefit of fat redundant pipes to the Internet. Thus, mySAP.com applications are well positioned to take advantage of this flexible and powerful accessibility option.
- ASPs offer interesting financing alternatives, in that they partner with various SAP technology partners to make leasing, pay-as-you-go, and other payment methods available.

The ASP provider market shrank over the last few years. The mySAP-focused companies that weathered these hard times seem even better prepared and well-positioned to host SAP solutions, however.

Security Considerations

I know of *no* company that does not envision protecting its corporate computing assets. From a solution-stack security perspective, not all software vendors are created equal, however. Oracle touts its unbreakable database, UNIX vendors tout the robust security features of their operating environments, and so on. In my eyes, security features are very important, but good security is more often about managing and testing changes to a solution stack, by carefully identifying security holes and other weaknesses in new solution stack components before these components ever find themselves in Production.

However, companies that embrace and act upon the idea of protecting computing assets will prove to be better partners in the long run. This is why I believe that Oracle's focus on security will pay big dividends in terms of slowing the adoption of competing databases. And it is why I believe that the *Trusted PC* joint Intel, AMD, and Microsoft vision (once labeled by Microsoft as Palladium, and now referred to as

the “next-generation secure computing base for Windows”) will prove fruitful as well. Its goal is to build security into servers and PCs at a microprocessor level. New initiatives coming out of the Trusted Computing Platform Alliance promise to better secure our processing platforms, ensuring that only authorized applications and program executables can ever be executed by the system, and that all data housed on the system is encrypted so that it is useless to others. To this end, Microsoft considers Trusted PC a significant part of its Trustworthy Computing strategy—we should see something commercially available in this regard by 2004 that applies to server as well as desktop and other computing platforms.

Manageability Considerations

No customers of mine have ever started their initial SAP implementation planning discussions with me by saying:

“George, all that high-availability and performance stuff is fine, but what we *really* want is a nice *manageable* system. Can you do that for us?”

By the time Go-Live looms just over the horizon, though, every single one of them—without exception—has indicated a growing concern for manageability. Sure, it’s there on the project plan, and any number of products can be used to support managing your mySAP environment. But the whole field of manageability is more complex *and more work* than you would imagine. Consider the following:

- Each layer in the SAP Solution Stack must be managed; the risk of not keeping an eye on a particular layer or solution component affects the uptime of the entire system.
 - Because each layer is so different from the others, it’s nearly impossible to find a single management product that can actually monitor and report on more than a few layers, much less the entire stack.
 - Therefore, the next best thing becomes trying to find a product that can at least interoperate successfully with other products.
 - At the end of the day, three, four, or even more tools and utilities must ultimately be fused together to provide a holistic view of a mySAP solution stack. This is challenging, to say the least!
- To learn exactly how challenging piecing together a management approach can be, **see “Systems Management Techniques for SAP,” p. 511** in Chapter 14.

Because of the challenges inherent to managing hardware and software products from a lot of different solution stack vendors, some of my customers have purposely chosen less than “best-of-breed” products for their SAP solutions, so as to minimize

the number of software partners involved. Or they have decided to reduce the number of partners and vendors altogether by selecting one of the big enterprise hardware/services vendors. The obvious partners are clear—HP, IBM, and Sun. For example, if you go with HP and choose to implement an rp8400-based server platform with an HP StorageWorks SAN, running HP-UX 11i, and managed by HP OpenView, the challenges inherent to managing four different vendors' products just dropped tremendously. Similar arguments could be made for going with an IBM or Sun solution stack, too—IBM even throws a couple of databases into the mix.

The System Landscape and Accessibility

The last area I want to cover with regard to solution vision and the SAP system landscape is accessibility. Many companies over the last three or four years have started with a vision of dumping all application-specific interfaces in favor of browser-enabled solutions, so as to ease the burdens and costs associated with desktop/laptop management while opening up new accessibility approaches like hand-holds and other wireless devices. SAP has supported that vision since 1996, with the advent of Internet connectivity in R/3 3.1G. But only in the last few years have I really seen this take off.

SAP AG offers quite a few accessibility options today when it comes to mySAP solutions. The classic SAPGUI and its revamped and more capable EnjoySAP SAPGUI represent one end of the spectrum. This approach is safe, very mainstream, and very easy to implement. And the SAPGUI we have today is extremely comprehensive, supporting all mySAP components through a single interface, which is unlike the approach a few years ago where each so-called "New Dimension" product like BW or APO required its own GUI. But the SAPGUI still represents a typical application-specific approach to accessibility; each end user installs the client on their desktop or laptop, or runs the SAPGUI from a network share, and off they go.

Other accessibility approaches are available, however, as you see in Figure 3.6. The original WebGUI, for example, is based on HTML and provides connectivity via Microsoft's Internet Explorer and so on. And a more recent addition, the JavaGUI, allows native Java-based access to SAP. Both of these approaches fulfill an Internet-based approach to connectivity, and subsequently simplify the desktop (assuming Internet connectivity is a standard desktop offering at your particular company, of course).

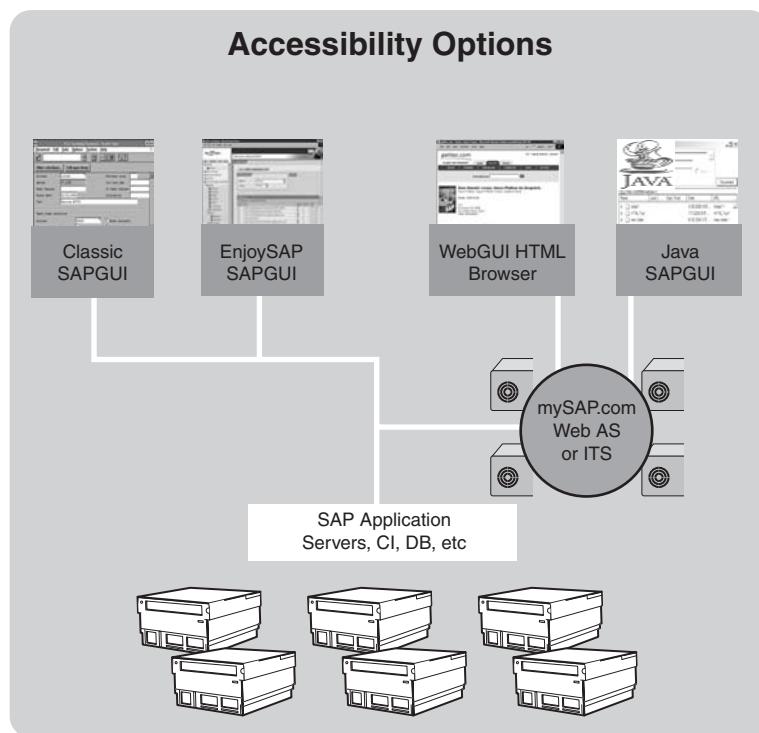


FIGURE 3.6 Access to mySAP solutions is quite varied today, ranging from classic and updated SAPGUI options to newer Web-enabled versions.

Capturing Your mySAP Solution Vision

As you work through all the different system landscape characteristics, considerations, and options, it is necessary to document why a particular approach or product was selected, and how it impacts the vision of the project. This documentation eventually finds its way into a *Knowledge Repository*, which is simply a documentation vehicle where assumptions, constraints, and so on are all maintained. This information serves as a set of boundary conditions and assumptions, useful later as you eventually engage various solution stack partners in fashioning your SAP solution, as you see in Figure 3.7.

In this way, whether you elect to publish a *Request for Information* (RFI) or complete a number of SAP Sizing Questionnaires, all of your hard data and related explanatory reasons for implementing each mySAP component in a particular way will be at your fingertips. The ability to share all this data consistently with all the prospective

hardware and software vendors is important—doing so enables a much better apples-to-apples comparison between different vendors' solution approaches later on, as you will see in Chapter 7. Further, as new information comes to light, or questions are posed by these prospective vendors, the Knowledge Repository will naturally lend itself to collecting and managing this incremental data of evolving constraints, assumptions, requirements, needs, and so on.

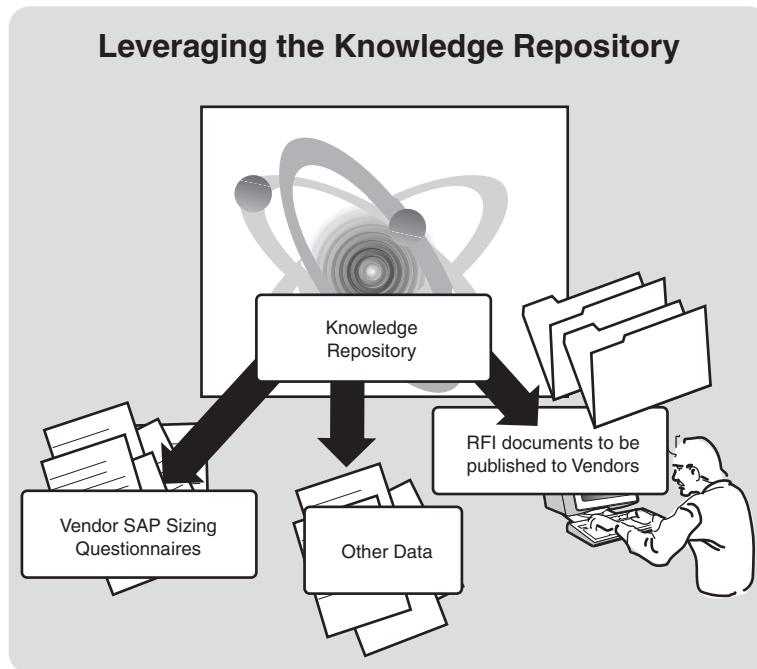


FIGURE 3.7 The Knowledge Repository will eventually provide input into either an RFI process or in answering SAP sizing questionnaires provided by various SAP Solution Stack vendors.

Leveraging SAP Sizing Questionnaires

Although the SAP Sizing Questionnaire is covered in detail in Chapter 7 and elsewhere, a quick discussion is in order here as it pertains to vision. Each vendor's SAP Sizing "Q" can potentially bring to light different areas of concern that you may not have yet considered. Microsoft's SAP Sizing Q is focused on their products and capabilities, HP's on their products and capabilities, and so on. As each organization's SAP Competency Center updates and republishes its respective Sizing Q, it tends to promote new high-availability, performance, manageability, and similar offerings.

Thus, the Sizing Q in and of itself can prove valuable in terms of educating prospective SAP customers. This may in turn help you to fill in gaps in your vision, or facilitate getting you better acquainted with IT and business drivers that otherwise might not be identified until much later in the implementation.

Developing a Request for Information

Rather than completing a whole lot of different SAP Sizing Qs, and going through everything that such an approach would entail after the fact, many companies choose instead to author and publish a Request for Information (RFI). I still promote the idea of going through various SAP Sizing Questionnaires in the name of education, of course, but using an RFI approach can be a much cleaner method for moving a project along.

A good RFI takes time to develop, though—a lot of time, usually. I have included a sample RFI (and related appendixes) on the Planning CD, not only to have something to walk through here, but also for you to use as a template of sorts if need be. Keep in mind that my 20-page sample RFI is quite short compared to what you might need to publish, though. My RFI addresses only two products, R/3 and APO, and contains very little in the way of legal terms and conditions. Your RFP could very well exceed 100 pages if you choose to implement a number of mySAP components or if you employ an ambitious legal department.

An RFI should include or take into account the following:

- General information, such as your company name and contact information, background data, and why the RFI is being published.
- Instructions to each potential RFI respondent as to how to complete the RFI. This includes how to address questions, details on the proposal process, confidentiality details, any disclaimers, and other administrative details.
- Terms and conditions, including the scope of the project, payment terms, minimum integration requirements related to existing/legacy systems, and so on.
- Requirements that must be met by the respondent, including any vendor and account management information you want to capture, the need for other SAP References, details surrounding the pricing model (including lease versus financing discussions), how to factor in maintenance windows or other planned downtime windows required of the proposed solution, hardware quotes, requested professional services quotes (for installation, migration of data, training, and so on), and any other information that may help you make a decision (such as each respondent's relationship with SAP, or the various database vendors, or a particular disk subsystem vendor, and so on).

In addition to covering the basics, an RFI normally contains or references various appendixes, too, which are designed to either share supplemental information or to enforce a certain type of formatted response. Let's walk through the seven appendixes I've included on the Planning CD:

- Existing Equipment Matrix—Documents what is in place today that must either be replaced by the new system(s) discussed in the RFI, or integrated with these same systems. This can also include a breakdown of existing SAP instances, or other enterprise application installations, that are currently productive. And if a current disk subsystem solution is already in place and expected to be leveraged by the respondent, details must be provided here as well.
- Software to be Implemented—Here, the Solution Stack as you imagine it at this point is shared, including expected versions of each mySAP component, database systems, operating system releases, enterprise management packages, and so on.
- Implementation Timetable—Represents an organization's hard requirements or possibly just a "best guess" as to when the new mySAP solution needs to be implemented.
- Reference Form—Applicable to absolutely *all* RFI respondents, though most often focused on potential hardware, software, and implementation partners.
- Cost Submission Worksheet—Consists of a standardized worksheet that forces an RFI respondent to price the project *your way*. This allows you to perform true apples-to-apples comparisons after all RFIs have been turned in.
- Staffing Matrix—Encourages each RFI respondent to give thought to staffing in terms of pre-engagement, during the engagement, and post-Go-Live.
- Sample Agreement with Terms and Conditions—Allows everyone responding to the RFI to understand up front what kind of legal constraints and financial commitments are expected.

Not all of these appendixes absolutely must be published with your RFI. In my experience, however, the kind of information provided in these seven appendixes is exactly what is needed by a respondent to either craft an SAP solution that really meets your needs, or to provide you with enough data to make an intelligent decision as to whether a vendor has what it takes to be your partner.

In Chapter 7, we will go through the remainder of the RFI process in detail, including how to compare and evaluate RFI responses, how to create a short list of RFI respondent candidates, approaches to making your final decisions, and more.

Revisiting the SAP Infrastructure Implementation Budget

With the information gleaned from refining our solution vision and building the basis for an RFI, we should be in a pretty good position to review our SAP Infrastructure Implementation Budget (SIPP) again. The SIPP needs to be updated to reflect new cost models, new business requirement-driven technology drivers, the need for incremental or other skillsets, and any other information that in effect changes our budget.

Based on these new budget numbers—one-time implementation costs as well as recurring license, administrative, operations, and similar costs—you should be in a position to truly consider outsourcing as a viable alternative to hosting everything internally. This is discussed next.

Outsourcing—Another Method of Achieving Your Solution Vision

Outsourcing is what I consider the fifth technology perspective, after the Conservative, Mainstream, Close Follower, and Leading Edge approaches. Most of this book assumes that your company owns and manages the SAP infrastructure necessary to implement your SAP solution. I also assume that the members of your SAP Technical Support Organization are employed or contracted by you, and not by a third-party outsourcing firm. In these final few sections of Chapter 3, however, I open the door to considering *outsourcing* these key assets instead, as illustrated in Figure 3.8.

What drives organizations to outsource? In a recent IDC study, the volatility of our global economy was labeled as the primary consideration. The study put forth the following ideas:

- Making large investments in computing infrastructure is not wise in today's economy.
- A company should instead let experts in the field of enterprise computing resource management make these investments, leveraging *their* core competencies in these areas.
- All non-core functions should be considered for outsourcing, allowing an organization to instead invest time and resources in its own core competencies.

This is really no different than in the past, when companies turned to outsourcing firms to cut costs. But today things are a little different, and cost is less a factor than pure *adaptability*, which is the ability of a company to make changes quickly so as to stay competitive or position itself better with their customers, vendors, suppliers, and so on. In a nutshell, adaptability equates to strategic benefits, rather than the simpler and more tactical cost-cutting benefits realized a decade ago through traditional outsourcing.

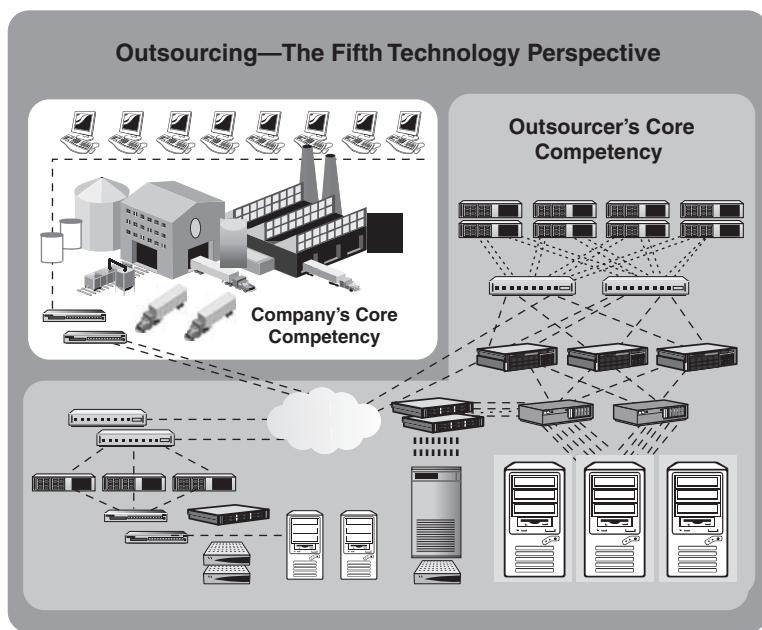


FIGURE 3.8 From both a business and IT point of view, outsourcing represents a unique technology perspective.

Intelligent outsourcing represents one method of becoming more flexible and adaptive, while still cutting costs. Outsourcing can mitigate risks relative to economic uncertainty as well, especially when the outsourcing agreement leverages the core competencies of each party. The really good outsourcing organizations, confident in their ability to execute, are more than willing to assume incremental risk. And with other risk-reward elements coming into play, such as those around meeting service-level agreements and availability targets, the best outsourcers are so convinced that they can do a better job of managing your resources and minimizing your downtime than you can that they're betting their revenue stream—your company's monthly check to them—on it.

With all of this in mind, exactly what should you outsource and what should you keep in-house? The short answer includes anything that is technology-intensive or complex from a process perspective. This easily explains why outsourcing SAP Disaster Recovery responsibilities is growing in popularity—DR meets both criteria in a big way.

Prerequisites of ITO—Information Technology Outsourcing

Although companies today can outsource technology or business processes, my focus in the remainder of this chapter is on Information Technology Outsourcing (ITO);

Business Process Outsourcing (BPO) is the label given to process-oriented outsourcing. A quick list of ITO prerequisites is in order before we move on, however. To really benefit from an ITO relationship with an outsourcing partner, consider the following “must haves”:

- The outsourcer must be flexible and able to adapt to your needs, both short-term and long-term. Thus, a clear understanding of the iterative nature of successful outsourcing is needed, as both tactical and strategic needs will morph over time. A rigid engagement and change-management model will leave you worse off than before.
- Your company’s goals and objectives must align with the outsourcer’s capabilities—if the outsourcer does not specialize in mySAP, or is uncomfortable providing references that otherwise prove their capabilities, walk away.
- A well-defined and articulated set of expectations must be communicated to the outsourcer. For example, your service-level agreements, requirements surrounding any systems management information you want to see on a regular basis, and so on, all must be clearly communicated up front.
- For global outsourcing arrangements, a good cultural fit is very important, too. At minimum, understanding your outsourcer’s culture is essential. But it’s really helpful to understand specific traits and tendencies. For example, in some cultures people tend to avoid sharing bad news with their clients, or in other cultures, it is not acceptable to answer a question with a simple “no” without providing details as to why.

If both parties meet these prerequisites, and you are comfortable with your potential outsourcer, you are a good fit for at least *considering* outsourcing.

Potential Benefits of Outsourcing SAP Infrastructure

The benefits you should reap from an ITO outsourcing relationship, compared to retaining control of your SAP assets internally, include the following:

- Less downtime and better availability. This includes both planned and unplanned downtime, as the outsourcer can presumably leverage their economies of scale, superior maintenance processes, and access to talented mySAP personnel.
- The same or greater level of flexibility. As your business needs change, so too should the system that supports these needs. This should manifest itself in a number of ways, including a full life cycle offering and “one-stop shopping.”
- Better consistency from a personnel perspective. Although employee and contractor turnover is not what it has been in the past—it’s quite reasonable

today—a successful outsourcing provider should still be able to retain its scarce technical resources longer than you can.

- Simplified budgeting and financial management of assets.
- High-quality approach and delivery.
- Reduced IT and supporting costs and little or no up-front capital expenditures on hardware and infrastructure equate to *more cash*. This is especially true when it comes to using offshore outsourcing partners.

To this last point, offshore outsourcing has been described as “counter-recessionary” simply because offshore costing models are so dramatically lower than U.S.-based models—recession or not, you are saving more money than otherwise possible by holding on to SAP assets internally. And with so many other countries beginning to compete successfully with India, which has dominated offshore outsourcing over the last five years (and currently owns 85% of all offshore outsourcing, according to Meta Group), the cost models will only continue to improve over time.

The Shortcomings of Outsourcing in the Real World

Historically, it has been difficult to find more than a few success stories where the company was so enthralled with their outsourcing partner that they could not help but tell everyone. My direct experience with outsourcing is pretty shallow, but from the stories my colleagues have shared with me, the following points seem to hold true:

- Loss of control seems to be the biggest concern. This relates directly back to the flexibility and adaptability that outsourcers today tout as compelling benefits.
- Less than overwhelming cost savings is another. Organizations that expect an order-of-magnitude cost reduction may be disappointed. Numbers like this are possible, true, but only if your own organization is so fat and bloated with overhead that you simply couldn't help reducing your IT bill in half.
- No perceived difference in the amount of time it takes to resolve system problems. This is especially true if your company's IT organization does their job quite well, leaving little room for improvement for an outsourcing partner.
- Outsource contract timelines vary considerably. One of my large SAP customers was persuaded to sign a seven-year outsourcing agreement a few years back. Seven years! That's an eternity in the world of IT, and they are “locked in” until the contract expires, lest they turn over a hefty penalty for early termination.
- Another customer of mine signed an outsourcing deal structured such that incremental processing power required by the customer during the life of the contract would be billed “per server.” Less than a year later, they began to

understand what that meant, as the outsourcer increased its revenue stream by meeting their new mySAP requirements with many two-processor servers instead of fewer larger servers.

- I've been told by customers that they sometimes feel "nickel and dimed to death" by their outsourcing provider. Every unplanned change to their environment, every new addition to the SAP system landscape, and so on add up to incremental and costly fees that were never envisioned by the original contracting team.

In looking back at the preceding list, it seems to me that many of the stumbling blocks stem from contractual issues rather than true outsourcing shortcomings. That is, performance problems were hard to find, and it seemed as though service-level agreements and general system availability were not issues, either.

Analyzing Outsourcing Versus Doing It Yourself

Just like hosting and managing your infrastructure internally, outsourcing touches every facet of your end users' experience with their mySAP solution. However, if an outsourcer can provide the same or better levels of service, responsiveness, and system availability, while successfully retaining the skillsets and expertise needed to keep an SAP solution humming along, and do all of this more cost-effectively than you, by all means outsourcing should be considered the forerunner in achieving your SAP solution vision.

The next step is to verify not only that the outsourcer is built upon a foundation of sound business fundamentals, but that it can demonstrate the following abilities:

- Can be effectively held accountable to deliver what it promises, through penalties and similar service-level-based fees
- Can show you proof of how it has accepted responsibility for its mistakes and shortcomings in the past
- Can point to a clear and time-proven methodology for planning, deploying, upgrading, supporting, and otherwise managing the mySAP enterprise computing resources of *other* customer organizations
- Can show you how its own processes and procedures are subject to continuous improvement

Why are these so important? Because they give an organization a way of comparing themselves to the best that outsourcing can provide. And because there is really no cost savings that will ever make it acceptable to circumvent these basic business fundamentals! In other words, flexibility, service, system availability, and authentic customer-service values mean a whole lot more to your end users than price ever will.

ASP Hosting for SAP

Another approach to managing resources outside the boundaries of your internally housed data center is through an Application Service Provider (ASP). What exactly is an ASP? According to IDC, ASPs provide a contractual service offering to deploy, host, manage, and rent access to an application from a centrally managed facility. ASPs are responsible for either directly or indirectly providing all of the specific activities and expertise aimed at managing software application or set of applications. Different ASPs tend to focus their services in different application areas—hosting traditional file, print, and Web services, and enterprise applications like mySAP make up the bulk of these. This is what tends to differentiate ASPs from general ITO and BPO outsourcing providers.

According to Gartner Group, ASPs deliver application functionality and associated services across a network to multiple customers by way of a “pay as you go” pricing model. As in traditional outsourcing, the value proposition clearly has to do with providing access to customer applications without the systems, staffing, and manageability challenges. After dramatic consolidation over the last two years, the strong remaining ASPs are growing again because they provide the following:

- Access to expensive and skilled IT professionals
- Knowledge in hosting mySAP and other applications
- Better reliability than most customer organizations enjoy, in regards to network and other infrastructure resources
- Alternative and flexible financing arrangements available
- Ability to include value-adds like e-trading, home pages for different organizations, and other Internet-focused offerings

Of course, like ITO outsourcing providers, an ASP’s specific knowledge and experience in supporting mySAP solutions, its reputation, an installed base of customer references, and overall financial stability are important considerations prior to securing their services.

HP’s Utility Data Center

Another very different method of accessing and provisioning scarce hardware resources is promised by HP. Announced in November 2002, HP’s Utility Data Center (UDC) does not seek to simply partition hardware platforms in a different way, or enable software-based workload management. According to HP Software’s Chief Technology Officer, Rick Hayes-Roth, “UDC is a software and services package that creates a data center infrastructure that administrators will wire once, then reconfigure dynamically.” Ultimately, a UDC data center administrator transparently redi-

rects computing resources to any application that needs it, thereby bringing true dynamic *provisioning* to the SAP Data Center by enabling different mySAP applications to access needed point-in-time resources.

In doing so, UDC promises to reduce data center costs by as much as 50%, a compelling proposition not only to large corporate enterprises but perhaps even more so to outsourcing vendors, managed service providers, infrastructure application service providers, and the like. To achieve these goals, UDC customers actually receive the following:

- HP consulting services, to architect the data center infrastructure.
- HP services, which are leveraged to keep the infrastructure up and running.
- HP OpenView's Integrated Services Management (ISM) software, necessary to manage and control resources under the umbrella of UDC.
- Cisco-based networking solutions, tying all assets into one computing fabric.
- An end-to-end solution vision that can easily evolve as an organization evolves. UDC supports multiple operating systems (HP-UX, Windows, Linux, and other UNIX variants), for example, and by tying these all together, enables the creation of huge compute and storage pools.

UDC will deliver on the ROI that all customers hope for but never fully achieve with current solution deployment models, simply because over-sizing and otherwise building scalability into each mySAP enterprise component, for example, is expensive!

There will be challenges that must first be overcome by companies wanting to adopt UDC, however. The most unusual will be related to the fact that UDC is not a single product or approach—instead, it's a marriage of hardware, software, people, and service-level agreements. Bringing all of this under the umbrella of ISM will be nothing compared to the time it takes to simply review current assets and then plan for a UDC solution.

Tools and Techniques

In addition to the sample RFI and related appendixes, I have also included a form for documenting the SAP solution vision, and electronic versions of each figure found in this chapter, in PowerPoint format.

Summary

In this chapter, I discussed the need and importance of crafting a solution vision prior to designing or implementing mySAP. Different technology perspectives were

covered, and the importance of refining the solution vision in regard to SAP system landscape requirements and constraints was covered as well. After the vision began to take form, I reviewed a few methods of capturing all of the data that came together to create the vision—constraints, assumptions, boundary conditions, and so on—including using a knowledge repository, developing an RFI, preparing for the SAP sizing process, and more. I then wrapped up Chapter 3 with a discussion on outsourcing, including leveraging ASPs and newer approaches like HP's Utility Data Center offering to improve systems manageability, provide better resource provisioning, and ultimately create a better customer experience for mySAP end users. In the next chapter, I build upon this foundation we have created to begin identifying and filling key SAP Technical Support Organization staffing roles.

4

Designing and Initially Staffing the SAP Technical Support Organization (TSO)

Overview—What Is an SAP Infrastructure TSO?

At this point, your SAP project is funded, and a vision is in place that addresses business requirements while simultaneously balancing cost, performance, availability, manageability, and so on. You also have your SAP Steering Committee and general project management structure in place. Your goal now is to begin designing an SAP Technical Support Organization (TSO) committed to the SAP infrastructure element of the project. The TSO can be broadly defined as the organization charged with addressing, designing, implementing, and supporting the SAP system landscape and its requisite components. By nature in a technical organization, the SAP TSO generally reports directly to the SAP Steering Committee's senior IT representative (who in turn reports to the company's CIO) or less often directly to the SAP client project manager or Director of Enterprise Computing Systems. Later, sometime before Go-Live, this technical support function will become part of the company's IT organization. I cover this transition later in the book, however.

IN THIS CHAPTER

- Overview—What Is an SAP Infrastructure TSO?
- TSO Overview—Roles and Responsibilities
- Organizational Chart Approaches in the Real World
- Recruiting the Core TSO Leads
- Consultants Versus Training Your Own Internal Resources
- Revising the SAP Infrastructure Implementation Budget
- Tools and Techniques

What exactly does “technical support” refer to? The answer is a bit complex, and revolves around the term *support*. If I were to sum up the technical support role, it would consist of an appropriately staffed and skilled team responsible for ensuring the planning, design, implementation, evolution, and ongoing operations of the technical infrastructure underpinning SAP. Consider the following:

- The SAP solution being implemented, however comprehensive, will not be able to completely manage and monitor itself. Thus, a team of subject matter experts knowledgeable in every layer of the implemented SAP Solution Stack must be constructed.
- The SAP solution will encounter issues that need to be addressed, both reactively and proactively. Again, a broad team of experts that understands *how* to troubleshoot issues, and *how* to leverage resources available from other organizations, plays just as important a role in supporting SAP as does the individual/team solving the issues.
- The SAP solution will evolve or change over time, the result of business and other drivers, and these changes will create issues as well.

Given the preceding, some kind of team needs to be held responsible and accountable for ensuring that the SAP solution is indeed *available* to provide value to the company. Hence, the SAP Technical Support Organization is born.

Given the typical complexity of an SAP implementation from an infrastructure perspective, the SAP TSO fulfills critical functions throughout the SAP life cycle. But the key lies in the attention given to ensuring that the solution addresses the business needs of the company. That is, it’s the business that drives the need for the SAP TSO. The TSO merely responds to the needs of the company to support the “business” of being in business. The business exists to service the company’s *external* customers, whereas the TSO services *internal* customers—the business folks. In this way, the SAP TSO indirectly services a business’s customers, as shown in Figure 4.1.

The SAP Infrastructure TSO includes everyone involved in ensuring that the SAP Solutions Stack is designed, implemented, managed, and otherwise supported. Thus, as shown in Figure 4.2, it includes every technical area or array of expertise needed in support of the SAP infrastructure, or SAP Basis layer.

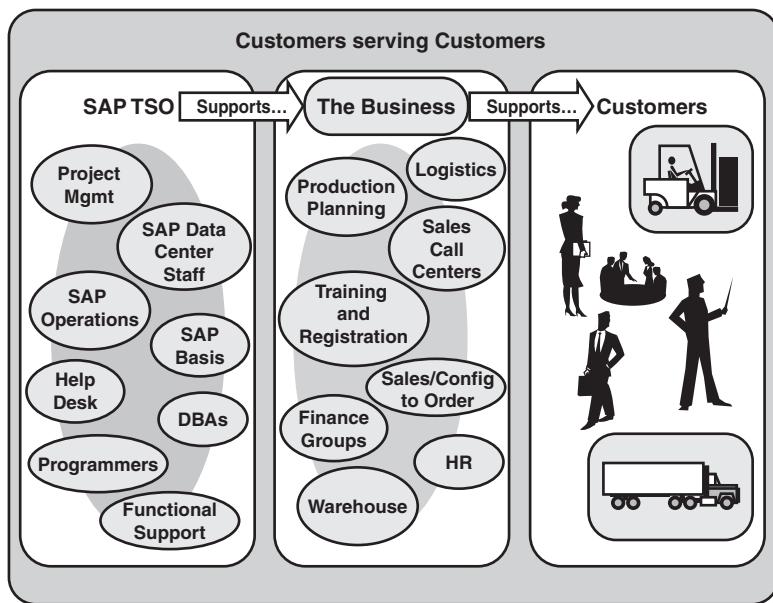


FIGURE 4.1 An organizations' customers are serviced indirectly by the SAP Infrastructure TSO.

The SAP TSO—What It Is Not

The TSO addresses support needs up and down the SAP Solutions Stack. And it covers these needs from inception of the SAP solution through Go-Live, future systems upgrades, and so on. In the context in which we approach the TSO here, though, functional SAP development and support is not included. That is, the programming and development staff charged with customizing the SAP solution for the company at hand is not covered by the SAP Infrastructure TSO. These SAP developers—ABAP/4, Java, HTML, and other technical coding and business-process functional experts—typically find themselves in another “box” within the SAP project organization charts. And, as shown in Figure 4.3, this box tends to align itself at a level equal to the SAP Infrastructure team/TSO, with both organizations reporting to the SAP Project Manager. Together, then, the Infrastructure and Functional teams comprise the entire project’s technical team.

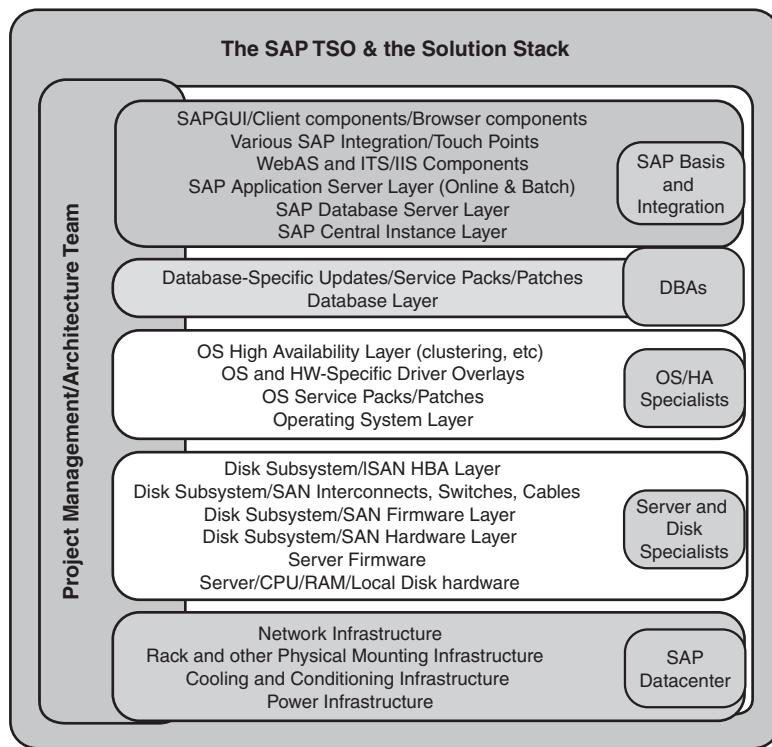


FIGURE 4.2 The SAP TSO touches every layer within the SAP Solutions Stack.

If we say that the SAP TSO is held responsible and accountable for ensuring that the SAP solution is *available* to the company, it makes sense to say that the SAP Functional Team is ultimately held accountable to ensure that the SAP solution *works*. That is, the Functional Team is chartered by the business to implement the business processes necessary to stay in business. This is therefore an absolutely critical function, key to the overall success of the project. It should be apparent by now that neither the SAP TSO nor the Functional Team can be found lacking—everybody loses if either party is not up to the support task at hand.

Fortunately, we find ourselves at an unusual time in the SAP job market. Whereas only two years ago SAP Project Managers, Basis, Functional, and other SAP infrastructure experts commanded hourly rates or salaries approaching that of sports stars (well, not quite), the burst of the dot-com bubble and general state of the economy has brought these rates in line with other high-value IT positions. I take a closer look at how to leverage these conditions in the next section.

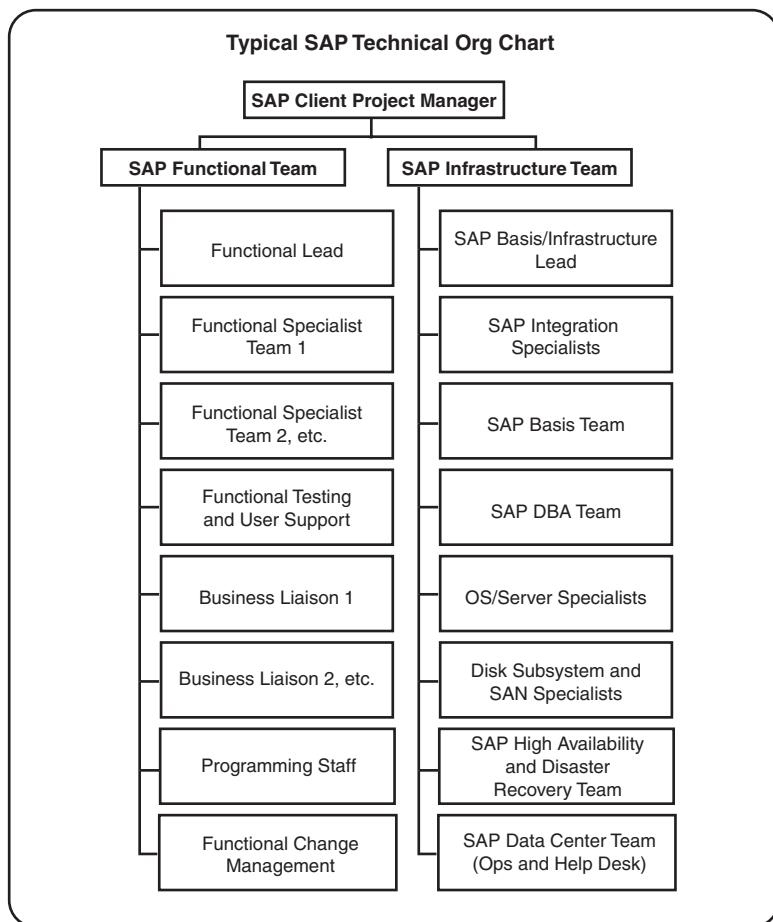


FIGURE 4.3 In a typical SAP project technical organization chart, both the SAP Infrastructure Team and the SAP Functional Team report ultimately to the overall SAP Client Project Manager.

Leveraging the SAP IT Job Market

Today, the SAP job market is saturated with what I refer to as “junior” SAP consultants, functional experts, and Basis/SAP infrastructure technology resources. These junior SAP technologists were typically on the fringe in previous SAP projects, or were simply new to SAP two years ago when we collectively hit the “mature” side of the SAP hourly rate bell curve. However, with less experience than their more senior colleagues, these important team members have accepted positions all over IT, waiting for their SAP ship to come back in.

On the other hand, the market also contains a few folks that were on top of the curve and have simply refused to take substantial pay cuts. These folks jump from project to dead-time to project, struggling to maintain lifestyles that were easily supported two years ago. If you have the budget and absolutely require the best of the best SAP “gunslingers,” these people are right for your project.

Still others, after their current SAP projects no longer needed them, decided to add SAP “bolt-on” experience to their resumes, and pursued extended enterprise projects around specialty or complementary software packages, Web-enabling technologies, and so on. These are the people who work in areas that touch SAP in one form or another, but typically do not represent core SAP support roles. Their value could be immense to your new SAP project, too.

General programmers/developers, network technologists, enterprise systems integration specialists, and other “generalists” fall into the SAP job market category as well. So, too, do people in the middle of retraining and retooling themselves (often at their own expense, at the conclusion of their last SAP project). In my experience, the generalists can bring some very SAP-relevant experience to the table, if you’re weak in say, for example, network infrastructure or basic ABAP coding. Be forewarned, though—their salary expectations might be higher than anticipated. As for the folks retooling themselves in a hot area like mySAP CRM or SRM, I have the highest regard for their proactive efforts to add incremental value to their resumes. If you are assembling a team for a specific mySAP solution implementation, and find yourself in need of a person trained in the solution component, though not necessarily experienced in it, this person’s “go-getter” mentality and personal drive will only add value to the team.

Of course, many SAP professionals are still engaged in thriving SAP implementation and upgrade projects today. These are probably the cream of the crop in the overall SAP IT market, as they possess current skills, and are therefore the most sought-after and sometimes the most difficult to recruit. But I should also point out the obvious about these folks—by the very fact that they are working today, they are harder to entice away from their employers. That is, I tend to assume that a gainfully employed SAP professional probably shows up at work every day and takes care of business, that he is competent at what he does, tends to have few or no interpersonal problems, and so on, regardless of whether any of these assumptions are actually true.

Finally, there are other individuals who simply left the realm of SAP consulting completely, to pursue new careers in enterprise computing or other fields altogether. As shown in Figure 4.4, the market is therefore quite fragmented—SAP expertise resides in pockets across widely disparate technology focuses, careers, companies, industry verticals, and unemployment lines.

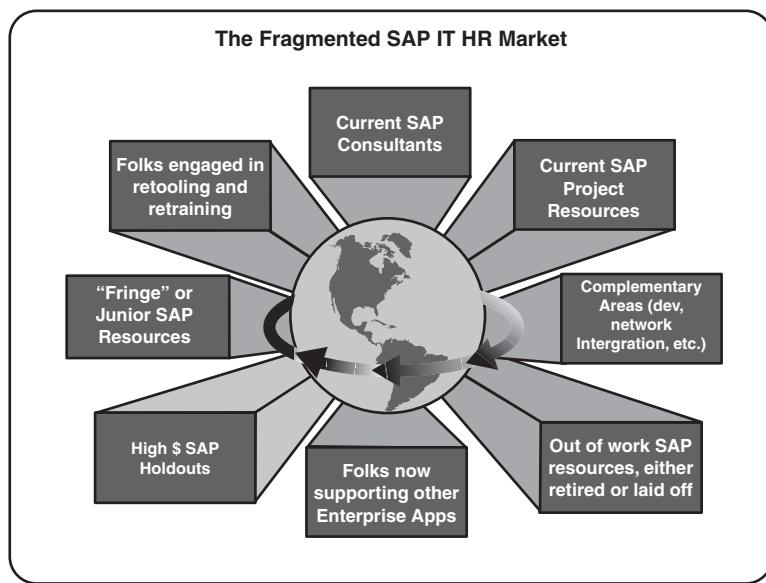


FIGURE 4.4 The SAP human resource market consists of SAP professionals who hold positions across the board, in terms of diverse technology focuses, careers, and more.

Another interesting observation that can be made is that the SAP IT human resources market seems to have more than its share of mature technology workers. Sure, there are plenty of young bright professionals in the world of SAP projects. But when you think about the knowledge and experience required to really support the SAP solution stack, it becomes very clear that the average resume of a seasoned SAP consultant or other professional reflects a lot of time in the trenches. Couple this with the fact that many former mainframe and other legacy IT computing folks moved into supporting SAP in the mid to late 1990s, only to find themselves the most expensive resources of a project (and the first to be let go when other factors were equal), and it's no wonder that the SAP IT human resources market is age-unbalanced today.

Hey, no surprises! The fact that companies and project teams shed their most expensive resources first unfortunately represents what we all see in every industry whenever there is a turndown. Companies consolidate, pile more work on fewer workers' plates, cut expensive contractors in favor of less expensive ones (or to save employee jobs), and so on. Of course, then, an organization's mature workforce usually represents a high concentration of jobs that are eventually eliminated. What's really interesting is that the "job" per se does not actually vanish from importance, though—database administrators still need to continue to monitor and plan for database growth, for example. Instead, the tasks associated with the job are moved to another person, and priorities are shifted between a shrinking support staff.

Think about the skillsets and experience out there in the job market! One project's loss is another project's gain. A senior SAP technologist caught in a downsizing has probably proven himself time and again as adaptable, eager to learn, and capable of doing the job. Consider the following advantages of senior technology professionals, given their tenure and experience in enterprise computing:

- They are equipped. They have the skills and experience necessary to support an enterprise computing solution. With their experience gleaned from supporting legacy environments, these folks have a keen understanding of many layers of the SAP solution stack. They have seen IBM mainframes replaced by Compaq, HP, and Dell server platforms, they have seen IMS and Adabas databases replaced by SQL Server and Oracle, and they have watched transactional systems evolve from green screens to full-featured distributed GUI-based wireless-enabled solutions.
- They are mature. With maturity often comes acceptance of things that can and cannot be changed. Thus, the senior technology professional possesses a wisdom rare in many organizations. He will be willing to take a step backward, or take time to invest in himself again in terms of training/retooling, because he has done it several times before in his career. He has little control over downsizing, but much control when it comes to personal and professional growth—and he knows it.
- They understand change control. In short, they understand *how* to support an enterprise environment, usually as the result of supporting mainframe and then later SAP R/3 and other enterprise environments. Some things never change, and the philosophy behind sound change control practices is one of these.
- They better understand the fact that work is indeed *work*, and unlike newer generations of employees, not necessarily fun all of the time.
- They are flexible and eager to contribute. That is, they are accustomed to personal change and sacrifice. They are also usually willing to negotiate compensation levels and positions within a project organization, to again become part of a project. The root of this is simple—these technology professionals enjoy working and contributing to a project where they are truly appreciated.

Those mature professionals who don't embody the characteristics in the preceding list will probably retire or otherwise remove themselves from the SAP job market. You'll never see them (their resume, that is). I suggest that you take advantage of these experienced folks in your own projects, and leverage the capabilities and cost advantages of the other folks that comprise the SAP job market today, as you begin to think through designing and staffing a balanced SAP Technical Support Organization.

TSO Overview—Roles and Responsibilities

If one thing is certain, it's that the organization supporting SAP will morph and change over time. Initially, the SAP TSO will be engaged in mapping business requirements to technical requirements. They will spend precious time understanding the business and comprehending the vision that the new computing solution will make available. These same folks will begin to put together a high-level project plan, too, and play a role in initially staffing the organization. They will then work hand-in-hand with the business to identify high availability and disaster recovery requirements, helping everyone to understand that high availability simply equates to dollars—the bigger the budget, the more available a solution that may be implemented.

Later, the TSO will need to embrace the exercises and work involved with designing and sizing the various SAP landscapes. Some of the key people will work with the steering committee and business groups to address total cost of ownership trade-offs (where again dollars come into play). Tasks specific to helping to further staff the organization, evaluating the various hardware and software vendors, and eventually developing the SAP data center, will also fall upon the shoulders of the SAP TSO.

- ▶ To read more about at what stage the bulk of the SAP Technical Support Organization is developed, see “Staffing the SAP TSO: Rapid Deployment”, p. 278 in Chapter 8.

Further down the road, the SAP TSO will be tasked with actually installing and managing the various SAP environments prior to “Go-Live.” Processes for managing change and managing the systems themselves will be put in place. Testing criteria, both functional and technical, will be developed in preparation for integration and stress-testing. Finally, the SAP TSO will look forward to the day when the big button is pushed and the company goes productive on SAP. At that time, the SAP TSO will change even more. The bulk of the organization will become something of a support and maintenance organization in many respects, while perhaps a subset of the TSO will stay engaged in strategic matters, working on additions, improvements, and other augmentations to the system.

With all of the activity mentioned in the preceding paragraph, it should be apparent to my readers that the SAP TSO cannot be simply staffed up front and left in a static state. Players will change, core competencies will probably ebb and flow based on one-time project plan milestones, and more. Some members of the SAP TSO will need to be comfortable in the board room, others in the computer room. And everyone will need to be focused on achieving customer satisfaction, even as the teams' customers change over time. In the next section, I take a closer look at building this team.

Building the High-Level SAP Project Team

It is likely that only the business-driven subset of the SAP Steering Committee is in place at this point in the SAP project. That is, the SAP implementation's Client SAP Project Manager (Client PM, or simply PM) has not yet been determined. If this is the case, a number of activities focused on organization and hiring must occur before the project can really gain any momentum:

- The SAP Steering Committee needs to take a hard look at internal candidates capable of taking on the role of SAP Client Project Manager. The ideal candidate will have both relationships and direct experience with the core business functions to be moved from legacy system(s) to SAP.
- If no internal candidates can be identified, the Steering Committee must immediately take on the role of evaluating and working with Big 5 or other SAP systems implementers, in an effort to initially find the right PM. Later, this PM will play a role in selecting the partners and vendors required to complete the deployment, upgrade, or discrete project.
- An organizational structure addressing functional and technical requirements needs to be developed and then staffed. This is discussed in detail in the next section.

How Many PMs Does It Take to Implement SAP?

It is important to understand that the term "PM" or "project manager" is used pretty loosely in SAP deployments and upgrades. In my experience, I have seen as many as *six* people on the same project who were referred to as project managers. Yes, six! In most cases, though, I tend to work with three or sometimes four, labeled different things based on the activities they oversee, or the aspects of project management with which they are held responsible, including:

- Client SAP Project Manager (PM)
- SAP-Employee PM
- Functional SAP PM
- Technical (or IT or SAP Basis) PM
- SAP Training PM
- Hardware Partner PM
- Disk Subsystem Partner PM
- SAP Security PM
- Database Technologies PM

The first three are by far the most common. For our purposes here, unless I say otherwise, the “Project Manager” is the person affiliated with the client actually implementing SAP—the Client SAP Project Manager.

Characteristics of a Good SAP Project Manager

Good project managers come in all shapes and sizes. That said, though, this person is usually either a long-time employee with specific knowledge of the business, or a relative newcomer brought on specifically because of previous SAP implementation experience. The best Client SAP Project Managers also

- Understand that they are change agents, managing change for the benefit of a company.
- Understand the role that technology plays to deliver and enhance business functionality: the big picture.
- Understand that everything is ultimately negotiated, and therefore are adept at getting things done—they are experts at assimilating a never-ending stream of cross-organizational data, and quickly executing decisions based on this data quickly.
- Understand internal and other company politics, especially with regard to how to get things done.
- Are customer-driven, and therefore results-oriented.
- Are accomplished presenters and communicators, comfortable with executives, business/functional folks as well as technical consultants.
- Are above all *leaders* in the true sense of the word, capable of inspiring others to put in the hours and effort necessary to pull off a successful SAP project while still maintaining personal and professional integrity.

In a nutshell, then, the best SAP Project Managers lead by example, leverage the talents and skillsets of others, communicate consistently, and can be taken at their word.

Organizational Chart Approaches in the Real World

As you saw back in Figure 4.3, a generic SAP TSO Organization Chart typically groups all functional folks in one large “box,” and all technology-support folks in another. This makes sense in a lot of respects, and so we tend to see it serve as the starting point for many SAP project organizations. But the real world always proves more interesting, as organizations attempt to leverage or live with existing management structures, communication channels, and so on. The resulting SAP TSO is also

impacted by organization biases, influenced by powerful or charismatic IT leaders, and pulled in many directions by the business.

In the next few pages, I share a high-level view of actual SAP Technical Support Organizations with which I have worked, at some of my own SAP accounts. And where it makes sense, I also go into detail in regard to why certain organizational decisions were made, and ultimately how effective the particular support organization was predisposed to be in each case. To maintain client confidentiality, I simply refer to each organization by an arbitrary customer number. I also provide my overall impression as to how well the end-to-end support model actually works, including my impression of what I perceive to be “Highs and Lows” of each SAP TSO—really compelling reasons why a particular approach should be embraced or avoided, for example. So sit back and read on. These real-world findings and observations are sure to surprise and entertain you!

Customer Example #1, Hardware Versus SAP Groups

SAP Computer Operations, as part of a larger and more general Hardware Group organization, manages and is responsible for ensuring that all servers and disk resources associated with SAP are up and available 24/7. At some level, this organization is also responsible for helping to drive hardware-layer changes (for example, to address growth/capacity issues proactively). All Operating System support, including working with their hardware vendor’s SAP Competency Center for tested and supported configurations of hardware and OS combinations, is provided by this group, too. They are also responsible for completing and verifying tape backups and restores, and implementing basic change control at the hardware and OS layers of the SAP solution stack. It should be noted that half of this group is represented by long-term contractors, and the other half by employees. The average skill level is also higher than most, the result of recruiting experienced support professionals from other locally-based SAP shops. Finally, half of the team also possesses certifications in the OS, hardware, or both.

Another group, called the “SAP Group,” is held accountable for everything above the OS. This team therefore consists of highly cross-trained DBAs, Basis, SAP Internet Transaction Server, and various mySAP.com component professionals. Occasionally, consultants are engaged from their hardware vendor, SAP, and third parties to provide discrete solution knowledge for limited consulting engagements. In the past, these engagements have addressed technical documentation, incremental infrastructure support when implementing new SAP components, and assistance in piloting and stress-testing new SAP components.

A disconnect exists between the Hardware and SAP groups, though, in that they ultimately report up through different organizations. This organization approach manifests itself in a host of different ways, perhaps best characterized as a

communications problem. For example, the Hardware organization tends to develop initiatives and then find support from one set of partner organizations, whereas the SAP Group chases its own initiatives and obtains support from other sources. As a result, strategic directions get blurry, and the two groups tend to step on each other's feet.

Highs: Well-trained and experienced staff across the board.

Lows: Poor communication between groups, little mention of career advancement between the two organizations, and a huge amount of responsibility is placed on the SAP group insofar as breadth of technology coverage is concerned.

Customer Example #2, The DBA Cubbyhole

SAP Computer Operations manages and is responsible 24/7 for ensuring that the SAP environment is available. In addition to performing general tape backup/restore and other typical operations activities, the senior operator is also engaged in the SAP “promote to production” change control processes. For example, each shift may be called upon to verify that transports completed successfully, or to apply kernel patches and perform other SAP basis maintenance (under the careful watch of the SAP Basis team, discussed later in this example). In effect, then, the SAP Computer Operations team is “layered”—the typical operator is responsible for overall monitoring of all computing systems in the data center, whereas the majority of the senior operators address SAP-specific hardware, operating system, and limited database and SAP basis support. The junior operators are typically contractors or consultants, whereas the senior operators are nearly always an employee of the company.

A second organization, the DBA Group, performs all space management and other database-related tasks. This organization represents a cubbyhole of sorts (albeit a common one), as it is responsible for a fairly narrow layer of the SAP solution stack. Turnover in this group has been high in the past. Typically, two to three people work in this role. Interestingly, the DBA Group reports up through a different organization than either the SAP Operations or our next organization, the SAP Basis group.

This third organization, referred to as the SAP Basis team, is responsible for driving all of the day-to-day, change-control, and strategic support for the SAP landscape. They interface day-to-day with the two groups mentioned previously, plus a small fourth group (sometimes referred to as Special Projects, which focuses more on new technology and impending mySAP solutions than anything else), overseeing the growth of high-end mission-critical computing systems at the customer site. To their credit, they have helped to document and support many of the day-to-day processes and management requirements of the SAP environment, offloading this to the senior operations staff. The link between the two groups is still quite strong, though, as the SAP Operations group escalates issues directly to the SAP Basis team, and the SAP Basis team trusts the Operations team to implement the fixes and changes.

Highs: The ability of the SAP Basis team to take advantage of both a qualified Operations team and a dedicated DBA team allows the SAP Basis team to truly focus on strategic initiatives, including spending time with the end-user business liaisons tasked with communicating new business requirements. In addition, the senior operator position is leveraged well, and represents an opportunity for career advancement both within the operations team and into the SAP Basis team.

Lows: The fact that each group (including the Special Projects group) reports to different directors within the overall IT organization really tends to slow down the change control process. And historically, the DBA team has never been granted the freedom to pursue SAP-specific DBA training, which I think is a primary reason for turnover in this group.

Customer Example #3, All Under One Roof

At this complex heterogeneous SAP customer site, the company's general Computer Operations team is actually not involved in ensuring that the SAP environment is "up" or available. There is no interest in managing the SAP environment from this perspective, despite the fact that an operations console had been set up previously to do just that, and all computing platforms are identical. Instead, a very small (four-person) SAP-specific "9-to-5, Monday-Friday" support group has been established to handle all hardware and operating system responsibilities, including monitoring uptime, completing backups, and addressing weekend and after-hours on-call pager support.

Database support is outsourced to the database vendor, rather than addressed internally. The customer has described this as both expensive and "out of hand," to the extent that a plan is being developed with the goal of pulling this support back in-house. Eventually, the goal is to cross-train the SAP Support team (covered in the next paragraph) in DBA activities, and shrink their production SAP R/3 database down to something less than a terabyte (yes, a terabyte).

SAP technical support is addressed by the SAP Basis Support Team, consisting of five SAP professionals. Overall performance tuning is the responsibility of one senior administrator, while the others rotate performing daily tasks such as applying support packages, kernel updates, and implementing other changes. An SAP Basis manager oversees all three loosely knit teams, and is held responsible for meeting business-driven service level agreements and addressing strategic concerns.

Highs: I like the fact that a single person in this flat organization manages all three discrete support functions (operations, database administration, and SAP Basis Support are all under one roof). This allows for decisions to be made and implemented quickly, avoids inter-organizational finger pointing, and affords presumably great opportunities for career progression. I also assume that the business-driven service level agreements require only core business hours coverage, and therefore I like the fact that the team also handles their own after-hours support requirements.

Lows: The duplication of effort between the general computer operations staff and SAP-specific staff seems wasteful, given the dead time that both organizations experience, and the fact that the hardware platforms that each group supports are identical (that is, the file/print, messaging, and other common IT infrastructure platforms are the same server models and platforms used by SAP). I was also quite surprised that anyone would actually outsource just their DBA activities without also outsourcing hardware, operations, and general systems management. Finally, the fact that only one person focuses on performance tuning is contrary to best practices—everyone needs a backup. Besides, the four SAP professionals relegated to applying support packages and kernel updates have got to be pulling their hair out for something more interesting to do!

Customer Example #4, Leveraging Computer Operations

Customer #4 leverages its investment in 24/7 general Computer Operations team better than most, in terms of expectations and responsibilities. Operations is solely responsible for ensuring that the SAP environment is not only available from a hardware and operating system perspective (utilizing a host of tools and utilities), but also in terms of the RDBMS and SAP performance in general. Utilizing an extremely comprehensive daily/weekly SAP Operations checklist specific to each of the three operations shifts, Customer #4's approach is both refreshing and efficient. Tasks such as checking print queues, exclusive lock-waits, the OS monitor, DB free space, the system log, active users overview, SAP housekeeping jobs, and more are performed on a scheduled basis periodically throughout the day via SAP's Computer Center Management System. Other tasks are performed less frequently, including reviewing the status of work processes and server status/alerts, and performing buffer analysis, checking the email routers and fax servers, and more. Operations also is responsible for tape backups/restores, and plays a key role in planning for and implementing changes to the entire SAP landscape. Not surprisingly, the eight-strong operations staff consists largely of employees rather than contractors. Contractors are occasionally brought in for backfill, though, as Customer #4 enjoys promoting from within.

This approach allows a second team, consisting of six to seven individuals in what is called the SAP/Database Group, to focus on strategic issues. High-level strategic issues include capacity planning, hardware and software upgrades, working with business groups to understand long-term requirements (like archiving and the deployment of additional SAP products), improving uptime by researching and implementing high-availability point solutions (for example, they researched methods of minimizing downtime windows, to increase availability of the system before it was demanded by the business), and so on. With the bandwidth afforded this group in terms of day-to-day responsibilities by the Computer Operations folks, the SAP/Database group too is highly leveraged, and offers outstanding ROI from an IT investment perspective.

Highs: This case study represents what I consider to be best in class. They expect a lot across the board, in terms of SAP solution support, and are rewarded with some of the highest levels of system availability and employee retention I have ever come across. They also leverage a general Computer Operations team, rather than investing exclusively in a dedicated SAP-specific Operations team.

Lows: Although their paper-based approach to SAP operations checklists has much merit (it forces the operations team to become and stay very SAP-literate, in terms of knowing and understanding transaction codes and how to pull data out of CCMS), it doesn't easily allow for cross-component or big-picture systems management. Implementation of an enterprise SAP-aware management system like that provided by HP Openview or BMC Patrol needs to be considered. Such an approach will provide this kind of cross-system monitoring capabilities, while still allowing detailed drill-down into each discrete SAP server or instance.

Customer Example #5, Multiple Production Instances

Given this global company's six distributed productive SAP instances (down from three times this after a number of mergers and acquisitions!), Customer #5 spends much of its time focused on ensuring that all SAP system resources are simply up/available—the fewer number of instances equates to more logged-on users per instance, and therefore the number of people that might be affected from downtime has increased dramatically. They leverage both individual server-based and an SAP-aware enterprise management product to monitor hardware availability and performance, while also proactively monitoring for impending hardware failures. A large SAP Operations center in the United States oversees all of this activity, and is staffed 24/7. Additionally, smaller support teams distributed throughout the globe are tasked with monitoring "their" productive systems as well. Finally, a subset of SAP Operations manages a single large instance supporting a few thousand users, as this resides on a hardware/OS platform quite unlike most everything else.

Database support and space management is handled exclusively by a six- to seven-person Database Team, also responsible for maintaining and testing the disaster recovery environments (which amount to log-shipping-based approaches). The members of the Database Team are experts in their field, and highly regarded within the overall IT organization.

The SAP R/3 Infrastructure/Basis Group reports to the same director as the Database Team, and is not coincidentally staffed in large part by former Database team members—more than half the team hails from here. Other than staff reductions resulting from their instance consolidation strategy, turnover has been almost nonexistent over the last few years. This basis team focuses on R/3 (as their name implies), BW, and the many interfaces to legacy applications and newer e-enabled applications across the company that focus on customer relationship management and e-procurement (all non-SAP).

Other teams are built specifically to plan for, install, and support other mySAP.com components (except BW), including Advanced Planner and Optimizer and Product Lifecycle Management. Although these different focused teams all ultimately report to another director, who in turn reports to the same VP as the SAP R/3 team ultimately reports to, the entire SAP TSO organization leverages the general approaches and practices developed by the R/3 team. In essence, then, the individual technical support teams are pretty consistent across the board.

Highs: Given what they had to start with, Customer #5's approach to managing multiple productive instances makes sense—I like the fact that they rolled up all high-availability monitoring to a central location and team of support folks. I also like how they have staffed their R/3 support team with former DBAs—this is the kind of career progression and promote-from-within philosophy that builds loyalty and ultimately higher system availability due to less employee turnover (they know their systems inside and out). Finally, I also appreciate the fact that the other SAP groups have not reinvented the wheel, that there is a general shared approach or standard as to how SAP is installed and managed throughout the company.

Lows: The duplication in SAP operations effort is probably costly (though it can be argued that this approach provides the “backup” that I am fond of as well). Just the fact that more than a couple of productive instances exists seems to be a nightmare, too, at first blush. On the surface, there appear to be some huge cost efficiencies that could be realized if the businesses were better integrated—fewer hardware platforms, for example, running fewer varieties of OSs would be an excellent place to start trimming operational, training, and management budgets. Customer #5 is well aware of this, but has pointed out that the cost of collapsing the environment into even fewer than six productive instances is expensive, to say the least. They acknowledge that the ROI gained from SAP instance consolidation is real, though, and despite the expensive functional and programming resources required, a number of integration projects are currently underway.

SAP TSO Organization Best Practices and Approaches

Given the actual customer environments presented in the previous sections, and generally accepted best practices in regards to staffing an SAP project, I assembled the following lists of pros and cons. What I really liked in the five different approaches includes the following guidelines:

- Create flat organizations, where the various SAP-related support organizations tend to report to one or very few managers. This approach facilitates better communication, clearer strategic vision, and faster resolution of issues.

Flat organizations tend to have another great side effect. That is, a flat organization makes it easier to promote from within—with fewer organizational issues with which to deal, and none of the “quit stealing my people” perceptions

that can arise in other organizational designs, I believe that flat organizations ultimately increase employee retention. That is, people grow quickly because their day-to-day tasks and work teams are not restricted by organizational boundaries. As people grow, their potential to take on new responsibilities is recognized across this focused organization; promotions are the recognition of this growth.

- Regardless of the organization model that must be employed, make it a common practice to promote from within. And “advertise” these promotions widely within your organization, to show everyone that not only do they happen, but also to highlight the skills, contributions, and attitudes that are important to the organization.
- Build a certain amount of redundancy into your organizations, such that every position has a “backup.” These backups are critical when people leave your organization, go on vacation, take off a week for training, or are otherwise unable to perform their duties. Create a certain amount of well-defined and controlled overlap, so as to effectively distribute tasks between closely aligned organizations (like between the DBA and SAP Basis teams, Data Center Support and Hardware team, SAP Operations and Systems Management group, and so on), and equip each team with the training and leadership to back up other teams.
- Leverage general teams, especially in smaller SAP implementations, rather than creating, hiring, and managing dedicated SAP-only organizations. This seems most effective in the areas of database administration and operations/help desk.
- Expect a lot! Hold organizations accountable for the goals given them, and for developing expertise that overlaps other organizations. But be certain to equip these teams to be successful, too, via formal training, cross-training, access to the proper tools and systems management approaches/methodologies, and so on.

On the other hand, the organizational approaches looked at had quite a few inherent problems. And there were other issues, too, that could be overcome with a bit of organizational redesign and thought into reducing turnover and promoting career advancement. I suggest the following:

- Avoid creating dead-end jobs and cubbyholes, where there is little or no chance of moving into other positions. If you don’t provide career progression, your competitor running or implementing mySAP down the street will be more than happy to. You may argue that some folks are happy in these kinds of positions—that’s true, I imagine. But creating a career path into and out of each

technology and support role helps mitigate the risk of not knowing for certain whether one of these types of people is employed in *your* organization. In other words, if they're happy in their current position indefinitely, they won't leave you regardless of whether you provided a career path for them or not.

- Minimize the number of managers involved in the organization, as this tends to slow down communication channels and generally confuse strategies and goals. Note that many of my most successful SAP clients prefer to fill the lowest levels of SAP TSO management positions with technical team leads and managing consultants, rather than pure managers.
- Avoid staffing too much with contract or other temporary help. This is especially true for long-term positions, where a lot of knowledge tends to walk away when the contractor or consultant finds a better position.
- Of course, when you find yourself in the position of bringing on a consultant or other temporary help, make sure that they are required to document what they have accomplished and learned while on the job. In this way, when they leave, a legacy of their SAP TSO tasks, responsibilities, and role is left within the organization.
- Enforce self-documenting approaches to managing and maintaining the systems. For example, leverage online or paper checklists in operations and enterprise management roles, and use systems management/monitoring tools capable of archiving data related to systems performance and availability. Such approaches not only best serve the organization day-to-day, but (more to the point of this section on organizational design) foster organizations where new workers can get up to speed as fast as possible.

In the end, designing a good organizational structure amounts to placing value in things like how easily the organization will be able to respond to issues and changes, how much opportunity will be given to people in terms of career advancement, and how the politics of the existing IT infrastructure will need to be overcome or otherwise sorted out. Just as important, the culture of the company implementing or supporting SAP will need to be taken into consideration—the value placed on bringing in experienced resources versus internally training your best people, for example, needs to be addressed. Before reading further, I suggest that you give a great deal of thought to what *will* work, and what *might* work, in terms of SAP Technical Support organizations. Document these approaches, and then walk down the list of pros and cons, best practices and so on, making notes and changes where appropriate. Discuss your conclusions with your steering committee and other key players. Finally, come to a consensus as to what the SAP TSO organization should look like given all of your own constraints and timelines, and then read on.

Recruiting the Core TSO Leads

Now that the high-level SAP Project Team is in place, and an organization structure is nailed down, the real work of filling in the initial key organizational “holes” can begin in earnest. In nearly all cases, as shown in Figure 4.5, this includes the following SAP professionals:

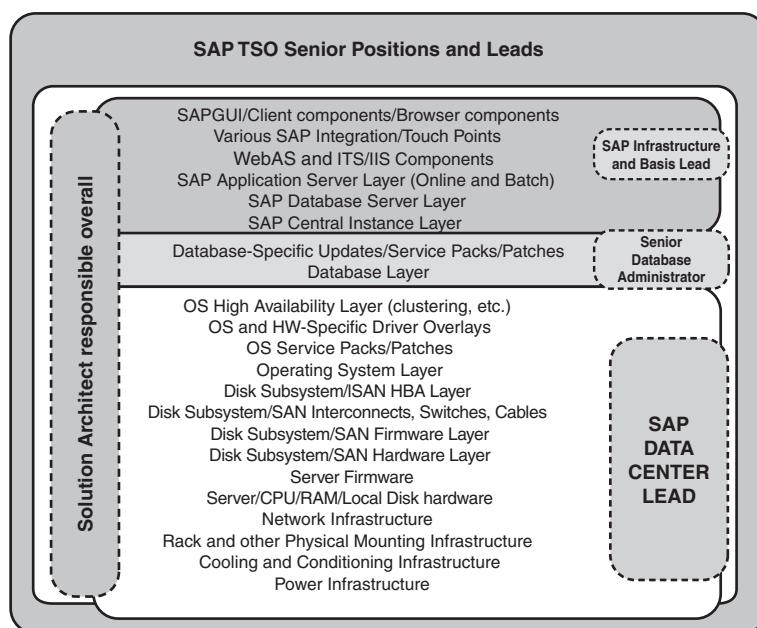


FIGURE 4.5 The core SAP Technical Support Organization Leads provide seamless coverage up and down the entire SAP solution stack.

- Solution Architect (SA)
- SAP Infrastructure/Basis Lead
- Senior Database Administrator (DBA)
- SAP Data Center Lead

The Solution Architect, or SA, holds a truly critical and unique position from both a team leadership and technology position. The SAP Infrastructure or SAP Basis Lead, like the Senior DBA, also both hold chief positions within the hierarchy of the SAP project, addressing both day-to-day and strategic concerns. Finally, the SAP Data Center Lead typically owns the SAP Solution Stack “below” the SAP Basis and Database layers—everything from the Operating System to the server and disk

subsystem hardware, to network, power, and cooling infrastructure. Like the Basis Lead and Senior DBA, this person also tends to put out fires as much as they spend time strategizing. The nice thing about this circle of senior SAP TSO Leads, though, is that they can all offload tactical matters to their teams when the time is necessary.

Let us take a closer look at probably the two lead positions of greatest responsibility and widest influence—the Solution Architect and SAP Infrastructure/Basis Lead.

What to Look for in the Solutions Architect

Perhaps the most critical position to fill at this time in the SAP project is the role of the SAP Solutions Architect, or SA. In a nutshell, the SA is responsible for converting the SAP project's solution vision into a core set of technologies capable of delivering on the vision—he maps business into technology requirements. Thus, the ability to architect a solution, or pull together the resources to architect the solution, is paramount. Additional hard and soft skillsets required of the SA include:

- A thorough understanding of what it means and what it takes to determine business requirements and translate these requirements into a workable SAP solution stack, or design
- Ability to set and achieve strategic vision, working hand-in-hand with project management and various technology-focused groups
- Competency in the design and sizing of the SAP components to be implemented or upgraded, including accessibility options like SAP's Web Application Server (Web AS) or Internet Transaction Server as required
- Characterized as a self-starter; a goal-oriented individual
- Ability to work in a fast-paced enterprise arena, balancing the needs of multiple stakeholders against project budgets, timelines, and more
- Ability to create and manage project schedules, meet deadlines, embrace strong time management and organizational skills, and react quickly to shifting priorities
- Ability to make and implement lasting decisions, and an aptitude in gathering consensus to make these decisions when time permits
- Excellent oral and written communication skills
- Solid negotiating and influencing skills
- Flexibility to travel to wherever the key stakeholders reside

As should be evident from the preceding list, the Solution Architect shares quite a bit in common with the Project Manager. Both are leaders, responsible for achieving key

project milestones. Both must be comfortable managing a team of specialists. Both need to understand the business behind the solution being architected, too. And finally, both are ultimately held accountable for a huge part of the overall success of the project. If the vision does not map to what was implemented, the project will certainly fail. It is therefore up to the PM and SA to work together in lockstep, pulling in key stakeholders when necessary, to ensure that the solution implemented indeed solves the company's business problems while simultaneously meeting ROI and other goals.

A good Solution Architect is not just involved with solution design and other "front-loaded" project management tasks. He is also often held accountable for designing and implementing processes to ensure the integrity of the SAP solution moving forward. That is, the SA plays a key role in helping everyone figure out how to manage changes in the environment as the system grows and morphs with the passage of time. Further, the SA ensures the completeness and accuracy of the documentation supporting the solution, usually directing knowledge transfer efforts to both the technical support groups and the ultimate users of the system.

From an operational perspective, the SA also provides operational training to the data center team tasked with supporting the solution stack implemented by the SA. This might include arranging for skills gap coverage of specific solution stack layers. For example, if a Storage Area Network was implemented "behind" the SAP solution, the SA will be held accountable for ensuring that the data center operations and help desk teams understand how to support this key component of the SAP solution stack.

The Solution Architect also normally is tasked with keeping an eye focused on where the SAP solution stack may be heading. He may lead research and evaluation efforts into new SAP component, database, operating system, or other layers of technology, or new/evolving functionality soon to be delivered by SAP with respect to what has already been implemented. The SA is a gatekeeper, too, communicating decisions that constrain business choices and the business implementation. In all of this, the Solution Architect acts as a "feedback loop" into the solution, looking ahead at the changes required of the solution to better meet the needs of its user community and business overall. And he is a feedback loop back to the business, too, helping the functional organizations understand how technology limitations and changes may impact the business itself.

Qualities of an SAP Infrastructure/Basis Lead

Whereas the Solution Architect is tasked with providing vision and design, the SAP Infrastructure/SAP Basis Lead is tasked with implementing this vision and ensuring that the vision "delivers." In doing so, then, he is held responsible for leading and coordinating a group of technical resources specialized in SAP technical operations and support. This includes identifying and resolving issues with anything that impacts high-availability, performance, scalability, and other key business drivers.

Real-world examples include tape backup systems, disaster recovery solutions, systems management challenges, the SAP change management strategy designed by the SA and embraced by the team, and more.

The ideal SAP Infrastructure team lead therefore is comfortable participating in a number of mini-projects or tasks simultaneously. He is comfortable implementing and managing the actual technical solutions dreamed up by the SA and others. In essence, the SAP Infrastructure team lead is one of the folks most responsible for ensuring that the company's business needs are indeed addressed today and tomorrow. He also performs the following:

- Leads any technology-focused initiatives or projects, often at the direction of the Solution Architect or any number of SAP Project Managers.
- Assists in staffing processes (covered in detail in Chapter 8); therefore, a certain amount of project management or general management experience is a plus. This includes administrative, interviewing, reporting, controlling, decision-making, and employee-evaluation skills.
- Constantly seeks to understand the "big picture" and how to leverage the resources of the company to achieve the big picture. This includes internal IT resources, standards, business processes, and generally how to get things done.
- Is adept at creating procedural installation, maintenance, and operations documentation, and knowledgeable in leveraging various documentation vehicles (for example, checklists, installation "recipes," building simple data repository Web sites and file shares, and so on.)

What does an SAP Infrastructure/SAP Basis lead look like? He's often a very senior technology guru steeped in real-world SAP deployments. But he is also the rare technology-focused individual responsive to the business drivers pushing the technology in the first place. He is a people person as much as a technology person, potentially capable of stepping into the Project Manager's shoes when asked to do so. Thus, he is adept at setting up and facilitating meetings, and following up on action items.

This technology-oriented leader is a *Subject Matter Expert* (SME) in business operations, too, with keen insight into how his understanding of the solution impacts the ultimate availability of the solution. He therefore represents the core of the technical support organization from a strategic technology support perspective. And he is accountable then in large part for the success of the project.

Such a key player is hard to find! Like the Solution Architect, sometimes they are already part of a company's IT organization, perhaps in a "new technology" or enterprise support position. More often, though, they are hired guns, coming out of the Big 5 and a few of the SAP-literate hardware solutions vendors like IBM and HP. In the next section, I take a closer look at different approaches employed by companies to procure and effectively use these kinds of high-end SAP professionals.

Consultants Versus Training Your Own Internal Resources

Two extremes exist in regards to staffing your SAP project. At one extreme, a horde of expensive money-hungry double-billing consultants may be brought in. You can find them quickly, and just as quickly start accruing enormous consulting fees. On the plus side, though, if they are managed and motivated properly (a really important “if”), a group of experienced consultants can take you through project preparation, planning, blueprinting, and implementation fairly quickly—many months perhaps, instead of more than a year.

On the other hand, pouring gobs of time and dropping maybe \$20,000 in training on *each* of your high-potential, intelligent, and investment-worthy internal IT professionals can turn these folks into inexperienced though now certified and suddenly sought-after SAP professionals. This latter scenario is unfortunately even *less* appealing than the first, though. Why? Because you have practically no hope at all of actually achieving your solution vision, regardless of how much budget and time you’ve been granted. Certainly, a middle ground of sorts must be achieved.

Thus, it became business-as-usual many years ago to employ a staff of perhaps 4–9 consultants for every employee or long-term contract IT professional engaged on an SAP implementation. To put it into perspective, a mid-size company with 10 resources tasked with implementing and supporting SAP might also have another 40–90 consultants, depending upon the mySAP solutions or components being implemented, the complexity of the solution (like the need for the highest levels of availability, or the inclusion of a discrete disaster recovery system at a remote site), and the number of functional areas (like materials management, sales/distribution, financials, human resources, and so on) for which business processes must be defined and deployed.

In the next few sections, I will examine textbook reasons why both consultants and trained company employees make sense, and then delve into educational real-world examples.

Training Your Own Staff—Values and Headaches

Bottom line, when it comes to training your own staff of SAP professionals, what you lose in terms of time you gain in terms of employee satisfaction, increased loyalty, and long-term return on investment. Let’s look at the particulars, though, using the approach I used earlier when discussing organizational approaches in the real world—highs and lows.

Highs: Training your own staff (and doing your best to retain them!) serves to keep key technology and business knowledge inside the company. That is, consultants walk away after the implementation is over, but employees can be motivated and otherwise leveraged to provide maintenance and support changes to the SAP solution even after Go-Live. This practice directly impacts career progression, too, while simultaneously making the organization a better place to work. And as I said

initially, the loyalty gained from investing in your people, and therefore the return on investment, becomes compelling indeed.

Lows: Training your own staff will not eliminate the need for consultants. And training does not provide the much-needed experience that truly makes an SAP IT professional valuable in his position on a project. Also, everyone needs to understand that training simply takes someone to a certain SAP release level, or level of functionality, or type of hardware platform, or version of a database, and so on. Training becomes a very expensive option if, in the middle of an implementation, a newer release or version of a particular SAP solution stack component becomes necessary. Think of it this way—if a consultant's knowledge becomes obsolete, it may be pretty easy to replace him with a different consultant knowledgeable or experienced in the newer release. But this isn't true with employees, where the employee then must be retooled, only to return to the project with a head full of knowledge but no practical application of that knowledge. This takes us to my final point. Resources that are trained but not experienced will typically take much longer to accomplish the same task than their experienced colleagues, often reworking the solution more than a few times until they get it right. If time is abundant, this could possibly be acceptable. But if, like most projects, timelines are pretty tight, the trade-off in time versus cost may simply not make good business sense.

Hitting the Ground Running—Consultants Versus Your Budget

The training that your internal folks lack (and therefore their ability to meet stringent project deadlines) is more than made up for in general costs. A typical SAP Basis or infrastructure consultant will range from \$60/hour for a junior self-employed contractor to maybe \$125/hour for someone out of a fairly reputable SAP systems integrator, to over \$300/hour for a senior enterprise architect from a Big 5 or "Enterprise" hardware or software SAP solutions partner. Figure that the least expensive resource is not capable of really meeting your needs, and even an inexpensive consultant will cost you \$250,000 annually. And this does not begin to include the expenses typical of this kind of arrangement, which normally add another 10–15%, or \$25,000–\$38,000 annually in costs.

Highs: Using consultants will speed up your time to implementation, no question. But there is also a certain amount of flexibility that can be enjoyed in leveraging consulting resources, too. For example, niche technology areas can be staffed quickly via consultants and short-term contractors, allowing you to continue to meet deadlines even when priorities or business requirements change. Further, if you ever have a problem with a consultant—bad attitude, a history of showing up late, or a consultant incapable of actually doing the work, and so on—you have great recourse. Let him go! Fire him! And push hard to not even pay for his "time" that was wasted, or deliverables that were "under-delivered."

Lows: Budget early, and budget accurately—as I have said before, assume that 65–80% of your overall costs will go toward funding your consulting staff alone. The costs are high, no doubt about it. But in the long run, thousands of companies have found that the costs of *not* implementing an integrated enterprise solution like mySAP are even greater. Can you really afford not to implement such a solution?

Compromising and Finding Organizational Balance

Compromising somewhere between staffing exclusively with consultants or exclusively with newly trained internal resources equates to how much you will benefit from keeping more of your SAP implementation knowledge inside the company, versus how quickly you will actually implement. That is, at a particular mix of consultants and internal staff, you will still maintain some level of control (via your own hand-picked staff) and at the same time leverage experienced experts in their fields (consultants).

Consultants Versus Internal Resources in the Real World

As we have seen, there are lots of reasons why it makes sense to use both consultants and internal resources on your SAP Project. But the real world provides infinite reasons why you need to achieve a certain balance between the two extremes, as I alluded to previously.

Bring on the Consultants!

I had one customer who started down the path to an SAP implementation two years ago. They brought in experts from one of the big consulting houses to cover most everything—project management, project coordination, blueprinting and design, functional consultants, ABAP programmers, SAP Basis, and other infrastructure consultants, and more. The customer named one of their senior managers as the Client Project Manager, but he still had other duties to oversee relating to the business, and could therefore be considered a part-time resource at best.

Within a year, this small company was averaging about \$50,000 day in consulting fees and related costs. Annualized, this amounted to over \$18 million, and did not even include budget money spent on hardware for development and test environments, and the SAP and database licenses themselves. For a project that never really got out of the blueprinting phase, they had literally spent a small fortune.

The project was way over budget and nowhere close to being on time. What happened? Changes in scope gradually crept into the project, unchecked by the customer and Client Project Manager. What started out as an R/3 implementation grew into parallel R/3, BW, and APO implementations. The business continued to feed requirements to the consulting team throughout the year. No one was truly in control, especially the client, and only the consulting firm was arguably coming out ahead.

In the end, the project was actually scrapped! Months later, SAP licenses and a pile of hardware in hand, I was asked to propose a new approach to implementing just the core R/3 component. No dice. Sadly, economic hard times and probably a deep-seated wariness towards SAP in general served to put this project “on hold” indefinitely.

Do It Yourself or Die Trying

Another customer of ours actually *did* successfully implement SAP R/3 with “barely” any consulting staff. In fact, the ratio of consultants to client staff was something like 2 to 1. The client took on 90% on the ABAP programming, all of the Basis and infrastructure, and most of the project management themselves. They brought in a second-tier consulting firm to help them for the first few months with general project planning and blueprinting, and leveraged free consulting resources from three different hardware and software vendors. Note that in all three cases, new and therefore not necessarily mature technology was introduced into the SAP solution stack—in exchange for providing references when presumably the project went live successfully, the client was granted consulting assistance gratis. It paid off for everyone quite handsomely, actually.

The project was not without its share of pain on the client side, however. The normal work week quickly grew to 60–80 hours (and then some), as the client painfully learned and relearned how to convert their business processes into ABAP and HTML code. New hardware, OS, and database platforms presented their own challenges, too, as the team scrambled to quickly learn and adapt their own operational and support processes to these.

Go-Live came and went without much more pain, and it came on schedule, which I believe amazed most everyone. Of course, the client’s team was exhausted from all of the hours put into the project. And they continued to struggle, suffering from “we don’t know what we don’t know.” Case in point, the production database quickly got out of hand, and then SAP tuning consumed all waking hours for nearly a month. All of this could have been avoided with carefully placed consultants tasked with providing good knowledge transfer before they hit the door on their way out. Nonetheless, the client struggled through all of this, and while it is debatable as to how much time they actually wasted, and what the state of the ABAP code is today, the end result was a live system. My team and I just hope they don’t have to upgrade for a long time!

Revising the SAP Infrastructure Implementation Budget

Now that an SAP TSO organization structure is in the making, it should be evident how and when the key technical resources need to be brought in. Rationalize and review the various organization charts I have included on the Planning CD, and other organization layouts available from your implementation consultants, hardware and software partners, and your SAP project manager.

Next, refine or pull out your most recent solution stack “best guesses.” Review the solution vision developed in Chapter 3, to ensure that any changes are reflected. Leverage these documents, and the knowledge of experienced SAP implementers, to get a feel for the level of technology support that each layer in the stack may require. The point here is not to try to detail the low-level positions, but to get a handle on how big or complicated the four key teams may become—and use this data to help determine the level of experience and background required of the four Lead positions.

Finally, pull out your internal company employee list, and work with the steering committee, IT liaison, Director of Enterprise Computing, and so on to get an understanding of who might fit this criteria. In parallel, work with your ever-growing advisory team of SAP, hardware, software, and systems integration partners to begin to understand the costs of external resources. Unless a clear internal candidate is identified early on, and get a thumbs up from both the steering committee and his or her own management team, plan for worse case and budget the position based on external consulting costs. Work through the other numbers found on my SAP Implementation Project Plan as well, and revise your own project plan’s numbers accordingly.

Tools and Techniques

On the Planning CD, I have incorporated a number of different organization chart approaches, job descriptions for the core TSO roles discussed in this chapter (including the SAP Project Client PM, Solution Architect, SAP Infrastructure/Basis Lead, Senior DBA, and the SAP Data Center Lead), and a Microsoft PowerPoint slide that may be used to justify both internal training and bringing in external consultants. I have also included the figures found in this chapter, in Microsoft PowerPoint format.

Summary

In this chapter, I focused on what it takes in terms of a technical support organization to pull off an SAP project. I looked at different approaches to organizing the SAP TSO, key staff positions, and the pros and cons of both consultants and internal staff. I also spent some time analyzing how real companies have structured their SAP support teams, including what tends to work well and where the challenges lie.

With this groundwork in place, coupled with a clear vision of what an SAP project entails and what needs to be accomplished, we are ready to move on. Now that we have effectively wrapped up the first section in this book, “Preparing for Your mySAP.com Implementation,” we can begin focusing on the next project phase—“Getting Started—Sizing and Blueprinting.”

PART II

Getting Started—Sizing and Blueprinting

IN THIS PART

- 5** Total Cost of Ownership Analysis in Preparation for Sizing
- 6** Blueprinting: Identifying High Availability and Disaster Recovery Requirements
- 7** Sizing: Evaluating the SAP Solution Stack Vendors
- 8** Staffing the Technical Support Organization
- 9** Critical SAP Infrastructure Training Requirements
- 10** Developing the SAP Data Center
- 11** Starting the SAP Implementation
- 12** Rounding Out Support for SAP

5

Total Cost of Ownership Analysis in Preparation for Sizing

Purpose of This Chapter

This chapter takes a closer look at *total cost of ownership analysis*, or *TCO analysis*. The idea at this stage in your SAP project is not to justify the project so much as to determine where and when the costs are incurred within the context of the SAP Solution Stack and ongoing operations. You also need to be able to compare various solution stack options and alternatives from a holistic cost perspective. By calculating rough costs before embarking upon major technology purchases, or before fleshing out the SAP Technical Support Organization with incremental and expensive human resources, the composition of the solution stack you eventually go “live” with will not only meet your business needs, but also reflect your budget needs.

A few years ago, one of my SAP customers put the whole TCO analysis process in perspective—TCO analysis limited their (solution stack) choices to better reflect their real-world budget constraints. Of course, in the end, if the choices that have been made fail to address the real needs of the business for which SAP has been employed—business process, availability, performance, and so on—the project will fail. So TCO analysis is also about balancing these requirements against their costs as well. The goal is simple—ultimately, a trade-off or compromise must be reached where the end-user organizations and financial bean counters see eye to eye.

IN THIS CHAPTER

- Purpose of This Chapter
- What Is Total Cost of Ownership?
- How the SAP Solution Vision Drives TCO
- TCO and the SAP Solution Stack
- Lowering TCO Through People and Processes
- Updating the RFI, RFP, or Internal Knowledge Repository
- Tools and Techniques

How you achieve that goal is addressed throughout this chapter. The first half of the chapter looks at the decisions made in regard to specific solution stack components: hardware platforms, the OS and database selections, high-availability and performance options, and so on. These are one-time implementation costs. Afterwards, I take a closer look at recurring costs, such as annual maintenance fees, upgrade/update fees, the cost of downtime to perform such updates, and more. I also look closely at tools and approaches offered by software vendors, hardware partners, and SAP AG alike, designed to reduce TCO and maximize ROI.

By the end of this chapter, through using the simple tools and approaches I have developed over the years, many of the solution stack decisions that you may be currently struggling with should be clearer. The only questions remaining should be those involving the complex details of high availability and disaster recovery, which are addressed in more detail in Chapter 6. By the time you begin Chapter 7, you will be in a position to fully engage your SAP Solution Stack vendors and partners and either publish an RFI/RFP or begin answering solution stack vendors' questions related to each mySAP.com component you intend to implement as part of your mySAP solution strategy. That is, based on all of the business, technology, people, process, and budget requirements and constraints identified through the first six chapters of this book, you will be well prepared to move forward into the realm of solution sizing.

What Is Total Cost of Ownership?

With your vision for 80-90% of the SAP solution decided, core Technical Support Organization players in place, general availability and performance requirements nailed down, knowledge of your technical organization's core competencies, and an initial budget approved, you are ready to perform the first of many TCO exercises.

But first, what exactly *is* TCO? Rather than attempting a long-winded all-encompassing definition, consider the following, and refer to Figure 5.1 for a comprehensive visual look at TCO:

- TCO analysis is all about determining how to get the best *business solution* for the least money. But it's much more complex than simply determining the relationship between computing power and budgets. Instead, a key goal of TCO analysis is to understand the cost-benefit trade-offs inherent to different solutions.
- TCO exercises must first of all focus on one-time initial costs, or *acquisition costs*. One-time costs include both product procurement and product installation costs, including training dollars spent in support of installation.

- A valid TCO analysis therefore factors in *technology*, including the SAP Solution Stack infrastructure and any hardware and software tools that make it up or aid in managing the stack.
- An incomplete and misleading picture is painted if only acquisition costs are considered, however. *Recurring costs*, including the cost of SAP operations and management, maintenance costs, the business cost of downtime, and other day-after-day costs must be factored in as well.
- A TCO analysis must also factor in the cost of *people* qualified to deploy and manage the SAP solution. In some cases, this might represent incremental people, though most often this instead reflects the cost of bringing in people with the necessary unique experience and skillsets.
- Finally, a TCO analysis must consider the impact that various operational and other *processes* have on the overall solution costs.

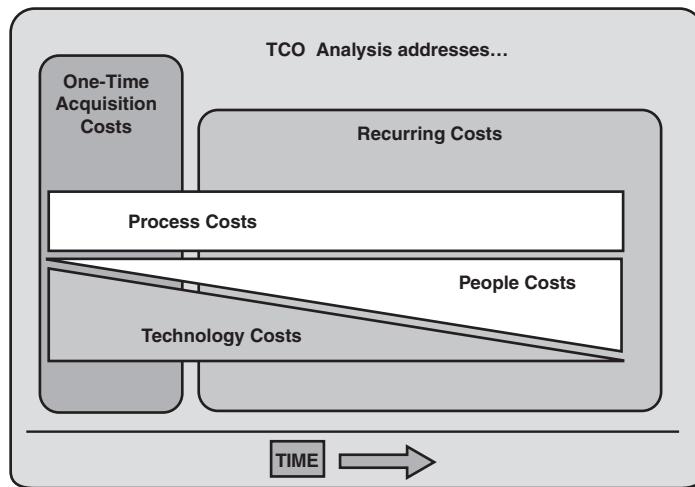


FIGURE 5.1 Both acquisition and recurring costs in regard to technology, people, and processes all play a role in determining a solution's true total cost of ownership.

Given the preceding description of TCO, a TCO analysis can help you evaluate different solution stacks, and even help you weigh diverse people/process alternatives like hosting SAP internally versus outsourcing it. This is possible because TCO addresses both internal (direct salaries) and external (outsourcing contracts) costs, one-time as well as recurring.

NOTE

In addition to the discussions of outsourcing found in Chapter 3, a “TCO SAP Outsourcing Analysis.xls” document can also be found on the Planning CD. This spreadsheet facilitates comprehensive comparisons between either outsourcing or hosting your own SAP landscape, which is sometimes called “in-sourcing.”

And on top of this, a TCO analysis can be both all-encompassing, or quite discrete. That is, TCO can be expressed in either absolute dollars, or in percentage differences (deltas) between various choices and alternatives. I use both approaches, depending upon the situation, but when I can I lean toward delta comparisons. Why? Because a delta comparison is simpler in that only the solution components that differ between two alternatives need to be examined. A delta analysis doesn’t cover the absolute costs of one solution over another. Instead, it seeks to identify the *difference* in costs between two different solutions. This usually yields enough information to make a smart decision one way or the other, especially with regard to procurement and people costs. In other cases, though, it becomes a judgment call as to whether one particular approach or solution stack is significantly more expensive than another *in the long run*.

For example, in the past when I was asked to build and analyze TCO models of a Sun/Oracle solution versus a Windows 2000/SQL Server 2000 solution, each capable of addressing 1,000 concurrent users, I only had to consider a handful of technologies, people, and process issues that differed between the solutions. In the end, the customer deliverable was shorter, more concise, and more applicable than a full-blown total cost of ownership analysis, because it was customer-specific and only took the deltas into consideration.

Consider Figure 5.2, where you see two very similar solutions paired in a delta analysis. Because the solutions are similar, only the *differences* need to be identified, analyzed, and charted here, making this approach simpler and more practical than a complete TCO analysis. Why? Because solution components with identical costs regardless of the solution alternative cancel each other out. In other words, tracking everything in the solution stack would be a waste of time—only the deltas need to be identified.

Solution alternative #1 looks to be more expensive than alternative #2 for a number of reasons. For example, #1 requires twice as many DBAs to administer the same size database for each solution stack, and the acquisition cost of #1 is four times that of the second alternative. However, from a database operations perspective, each is on par with the other. And the costs associated with the remainder of the solution stack are the same, too, in this case, and therefore not analyzed.

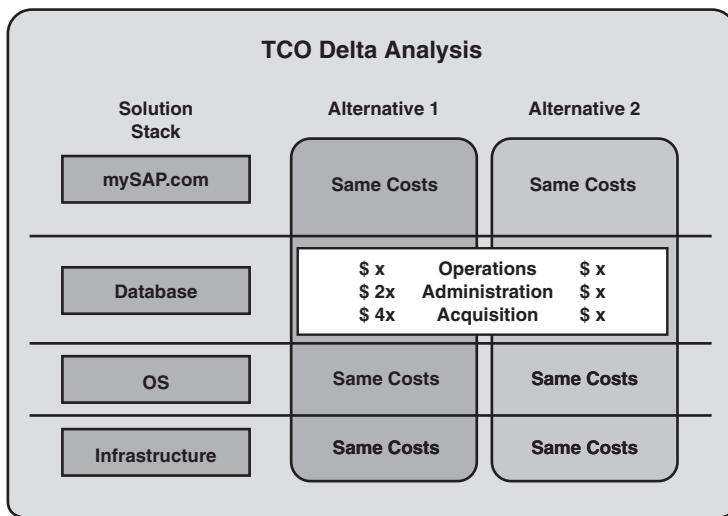


FIGURE 5.2 A TCO Delta Analysis represents a quick method of determining, for example, which solution alternative is least expensive from an operations and systems management perspective.

How the SAP Solution Vision Drives TCO

Total cost of ownership analysis seeks to measure the life cycle costs of a particular solution stack—the end-to-end complete costs incurred to own and operate a mySAP solution over its useful life. Thus TCO serves to highlight the relationship between cost and performance, illuminating how quickly a particular solution can claim a return on investment.

Returning to Chapter 3 and the various solution characteristics that need to be envisioned and planned for in advance, a number of characteristics were discussed. I have collapsed these into a few key areas or considerations:

- High Availability
- Disaster Recovery
- Performance
- Scalability
- Security, Manageability, and other Operations areas

Throughout the next few pages, I will focus on these areas as each pertains to the SAP Solution Stack from a technology perspective. Later, I will identify people and

process considerations inherent to each layer in the stack, like ongoing operations, systems management, and other processes that are subject to continuous improvement, therefore lending themselves to reducing or increasing TCO.

The Impact of High Availability Requirements on TCO

High-availability (HA) requirements refer to the need of your SAP solution to suffer from only a limited amount of unplanned downtime. In other words, the higher the level of HA, the more available a solution is to its end users. This availability is often expressed in percentages related to the total number of minutes available in a year. Over time, these percentages have been labeled, giving us the infamous “3 nines,” “4 nines,” and “5 nines” of availability (which equate to 99.9%, 99.99%, and 99.999% availability, respectively).

Generally, the higher the level of availability required by the business, the more costly it becomes to procure, implement, and support the system in question. And the relationship between cost and high availability is not linear; rather, it grows exponentially as we strive to achieve something closer and closer to 100% availability, as you see in Figure 5.3.

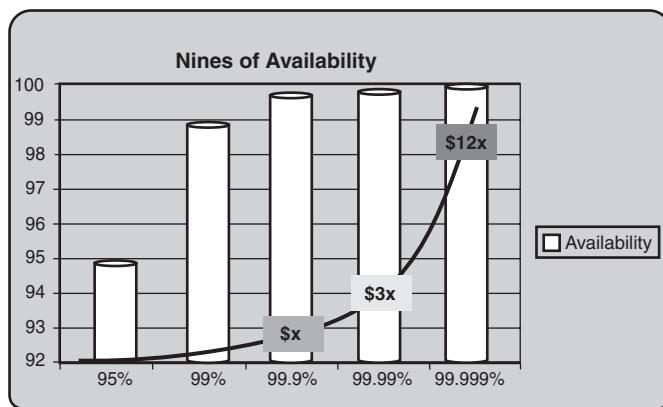


FIGURE 5.3 Small incremental percentage gains cost more and more as we inch closer to 100% availability.

Figure 5.4 really puts this into perspective. As we move from 99.99% availability to the famous “5 nines” of availability, the incremental cost to the business to achieve another 47 minutes of availability is \$2.9 million, nearly *five times* the cost required to jump from 3 nines to 4 nines (\$590K). And yet in terms of raw numbers, we only add a fraction of a percentage to our availability targets—a number measured in mere minutes. For businesses that lose millions of dollars for every minute of

unplanned downtime they incur, five nines is the way to go, of course. In my experience, though, the incremental cost is usually not worth the nearly negligible difference in uptime.

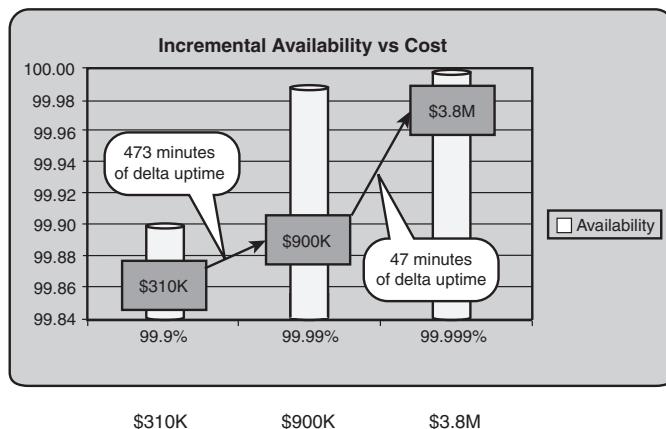


FIGURE 5.4 For this fictional enterprise, note the small amount of time or availability gained as we increase our SAP solution's availability from 99.9% to 99.999%.

I have always told my customers that I can provide them pretty much any level of availability they'd like—the only issue is money, of course. Many of my customers would commence our initial discussions on high availability by telling me that their business demanded “the highest levels of availability” or “little to no downtime.” Many of these same companies had failed to do the “5 nines of availability” math, however. Their need for this level of availability was little more than a perceived requirement, a desire. After we worked out the budget and ROI numbers together, it was amazing to watch the perceived business requirements nose dive to embrace less expensive technology solutions more representative of the real business’s needs.

I like to structure the end result of these types of ROI calculations such that they reflect how much the business will pay for that extra few hours or minutes of availability. The real-world numbers have been pretty significant in my experience, in one case running up to another \$500,000 over the three-year life cycle of the project for seven hours less unplanned downtime per year. In other words, giving 21 hours back to the business would cost the business nearly \$24K per hour. I then had to pose this question to the customer: “Will you suffer more than \$500,000 in lost revenue or productivity if you are down an incremental seven hours a year?” In this one particular case, the cost of downtime escalated pretty quickly with every hour of downtime incurred by my customer, and they indeed required the solution. Most of the time though, in my experience, the answer is usually more like “Oh! I guess not”

and my TCO-educated client settles for a lower level of availability that may not be the highest, but is *good enough* for the business.

Good enough—what a great phrase! In these days of redundant and clustered everything, companies often seem to forget that going after that extra wall-clock tick of availability can really add up, and yet not significantly impact their ability to do business. *Good enough* is all about compromising, settling for 95% of what you desire, but paying only 50% of the cost of a solution that could satisfy 100% of your desires. In so many cases, for so many SAP implementations, it's the right attitude. The decision boils down to identifying business *requirements* versus desires. As I've said before, just do the math. There's nothing like a logical discussion around ROI, TCO, and HA to convince a business that *good enough* will serve most companies quite well (not to mention leaving a little money in the corporate budget for things like Go-Live bonuses!).

Disaster Recovery Requirements That Drive TCO

Similar to High Availability discussions, determining a company's Disaster Recovery (DR) requirements also boils down to how much it costs the organization when the SAP system is unavailable. But where HA addresses timelines of minutes and hours, DR discussions focus most often on hours and days.

- ▶ To learn about Disaster Recovery options in depth, see "Determining Your Required Level of Disaster Tolerance," p. 167 in Chapter 6.

Figuring out the real cost of downtime after an extended period of time—days—can be quite complicated. Consider the following:

- Business processes must usually be capable of failing over to a completely different physical location. Therefore, the technology ramifications are huge, to the point of requiring completely redundant data centers or hosting sites in some cases.
- Ownership issues are huge, too—every member of the SAP TSO needs to understand exactly what their team is responsible for providing in the event of a disaster, and staff/plan for this accordingly.
- Communication issues are therefore paramount. A communications plan must be developed, tested, and continually retested as the solution stack evolves over time. Similarly, communication vehicles like escalation plans and even system current-state/as-is and process documentation must be maintained and updated at both the primary and DR sites, and tested regularly.
- Technology issues abound, including backup/restore concerns, data synchronization between the primary and DR site, access to the DR site by the system's end users, access to the site by other computing systems (integration touch points), day-to-day operations/management of the DR site in the event of a

disaster, and issues with how to fail back over to the primary site (or another site) after the disaster has subsided.

- Additional people-issues exist, too, like determining contingency and other backup plans should key people be unavailable to perform their duties in the event of a disaster. Here, the importance of consulting agreements (that is, “consulting on demand” or “reactive services” contracts) is underscored, as is training a backup for each key role, maintaining excellent documentation, and so on.

All of these issues impact cost, and therefore present opportunities to minimize or increase TCO. Most often, my involvement in Disaster Recovery-related TCO exercises has amounted to doing the math between different DR alternatives. In some cases, the math supports building redundant data centers. More often, though, establishing a mini-data center or putting into place a support agreement with a third party to provide such an environment is more appropriate.

In the best scenarios, a customer is able to distribute their SAP environment across two existing sites, though. For example, it's quite common in my experience to house the Production and Development systems at one physical location, and a Staging or Test/QA environment at another physical location. In this way, the site with the Staging or Test/QA system becomes the de facto DR site. This is generally an excellent approach to addressing disaster recoverability. Consider the following, however:

- The system that takes on the role of the DR system must be sized appropriately, based on what the business considers “appropriate” performance after a failover. It is not uncommon to size a DR system for half or perhaps three-quarters of the concurrent SAP end users typically hosted by the production system. In some cases, though, a DR system capable of hosting all production users, batch processes, and so on is mandated by the business. And in worst-case situations, both the Production system and a fully-functioning Staging or Test/QA system need to be available concurrently, even in the event of a failover.
- In a similar manner, the *high availability* built into the DR system must be considered. That is, the DR system may need to be clustered or configured with redundant components to achieve a certain level of availability should it become the Production system for an extended period of time.
- Along with maintaining the system for its primary purpose of testing or quality assurance, this system must also be kept up-to-date from a DR perspective, consistent with the service-level agreements and other requirements of the business. Therefore, this could entail anything from weekly SAP client refreshes to 15-minute database snapshots replicated across a WAN or other network link, in addition to the load already placed on the system.

- SAP end-user access to the DR site must be addressed, because a DR site is no good to anyone if no one can access it. This often involves redundant network links between the client public networks and the DR site. In the best of cases, separate carriers (for example, AT&T and Southwestern Bell) are employed, to avoid single points of failure specific to the carriers themselves. But most often the real challenge here is more a matter of explaining to the end users how to get to the production system should it failover to the other site—which SAP logon group to use, or special DR-only SAPGUI icon to double-click, or ITS server to access, and so on.
- The location of each site is critical. For example, there is more risk of a single disaster taking down both the primary and DR sites if they are within a few minutes of each other rather than if the sites are separated by a hundred miles or so. This location issue is common in campus or single-building environments, where the backup site might be only a mile (or a few floors) away from the primary site. Of course, the economics of such a decision are easy to understand—a DR site that is close by will probably be easier to manage and maintain. My point is this, though—identify the lack of critical “distance” between the two sites as a risk, and either mitigate or minimize this risk.

When it comes to Disaster Recovery and TCO, it's important to focus on the business areas that will cripple the business if interrupted or unavailable. Therefore, core transactional systems responsible for generating revenue, maintaining minimum levels of customer service, and keeping the production lines rolling are generally most often addressed. Reporting systems, internal procurement systems, and other such functions typically fail to garner enough business support to cover the expense of a dedicated DR system or site.

In addition to focusing on the core business areas, it's also wise to establish exactly what it means to say that a business function is interrupted or unavailable. Does this mean eight hours? Three days? A week? Bottom line, when does unavailability become unacceptable? The answer to this question drives failover timelines, impacts service level agreements, and more.

And finally, as you see in Figure 5.5, the most effective TCO Disaster Recovery analyses benefit from a sound costing model or baseline—such a baseline should reflect how much revenue or productive time may be lost prior to moving business processes over to the DR site. And it should also cover time as a function of dollars, so that the relationship between the two clearly illustrates how losses grow over time.

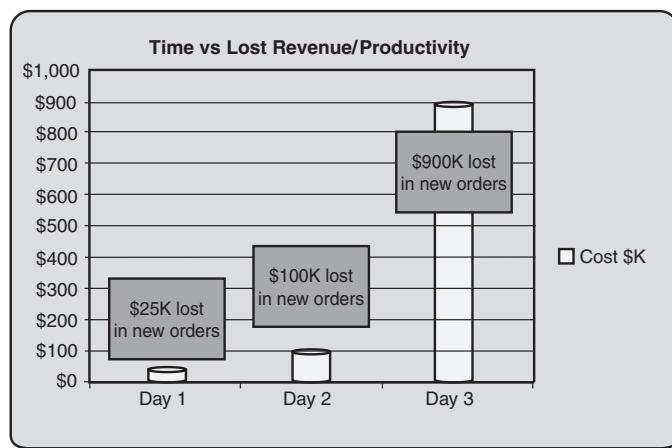


FIGURE 5.5 The relationship between time and dollars can help characterize and justify the relative importance of a DR solution, as well as serve in delta comparisons.

With this information, you naturally have a better understanding as to how long the business can actually tolerate disasters, including different levels or thresholds of pain. Such a keen understanding will serve to further refine the DR solution/approach down the road, as cheaper and more capable technology solutions continue to evolve and therefore rate delta TCO analyses in their own right. In my opinion, this whole area is probably one of the best places for something beyond a simple delta analysis; understanding the complete DR picture can conceivably mean the difference between surviving a disaster and going out of business a few months later. And with so many people and process considerations, building a good technical solution is simply not good enough.

Performance Requirements and TCO

Like so many other areas of TCO analysis, the more performance sought, the larger the IT budget typically needs to be. The key lies not so much in understanding this, though, as in understanding how to *measure* performance such that it can be factored into a TCO performance delta analysis of solution stack options. In the past, I have characterized SAP performance in terms of

- End-user response times as related to average and peak periods (like end-of-month peaks, or peaks observed during a seasonal cycle).
- Dialog steps processed during the peak hour (as a reminder, dialog steps represent units of work in SAP, where one dialog step equates to a user pressing the Enter key or otherwise completing a transaction against the database).

- Average number of fully completed transactions processed in an hour (like dialog steps, this is another way of measuring *work* completed by the SAP system).
- Average number of concurrent processes executing in the system while it's under load (easily gathered point-in-time by SAP CCMS transaction SM66, or historically via any number of SAP-aware management applications).
- Average disk queue length while under load (captured via PerfMon, similar UNIX-based utilities, or most SAP-aware management applications). This kind of performance measurement is most applicable to changes in the disk subsystem or database layers of the SAP Solution Stack.
- Average CPU utilization while under load (again, using PerfMon or a similar UNIX utility), most applicable to changes in server infrastructure, but also valuable when new disk subsystems are introduced (to observe how a bottleneck moves from the disk subsystem to the server, for example, due to a faster subsystem).

After a measurement is embraced, it's fairly easy to redefine it in preparation for a TCO exercise, in terms of dollars. Thus, transactions per hour becomes "x\$ per transaction," "% CPU utilization per dollar," and so on.

Further, it's usually easier to perform a delta analysis rather than a full-blown performance analysis. Some of the most common areas of TCO analysis when it comes to performance involve testing new versions of SAP, new disk subsystems, and configuration changes to existing systems. Another area might include a change in operating system, or the application of a Service Pack or patch to a particular layer in the stack, or the impact of installing multiple ITS instances on a single server. In all of these cases, testing the before and after scenarios simply makes the most sense. And it simplifies both testing and success criteria, as absolute numbers tend to not be as important as the delta between the two.

How Scalability Impacts TCO

The need for scalability in an SAP system also impacts total cost of ownership. Scalability is historically addressed by purchasing more than you need when it comes to hardware resources. In this way, headroom is available should it become necessary, for example, in the event of heavier-than-usual end-of-month processing or after new processor- or disk-intensive functionality is added to the system. These "in the box" scalability considerations include

- Buying a server with additional CPUs, RAM, or disks, in effect *supersizing* the solution to address unknown what-if needs at a later date. In the meantime, such an approach will naturally provide better-than-expected performance of the current solution as well.

- Buying a server *capable of scaling* or growing in terms of the number and speed of CPUs, RAM, disk capacity, I/O slots, and so on—without actually adding these components. In this way, the oversized system is ready to grow with only minimal downtime required. Unfortunately, you might wind up buying a more capable platform than you ever need.
- Buying a disk subsystem or supporting infrastructure capable of easily housing more data, or supporting rapid expansion of capabilities, instead of buying the exact capacity required for the time being. Such incremental capacity might include a SAN’s switched fabric containing a higher available port density than currently required, or a disk subsystem with a few extra empty disk shelves (and therefore ready to add disk drives).

Scalability can also be addressed by virtue of the architecture of the solution. In a classic example, it’s common to see a large number of relatively cheap servers employed as SAP application servers, rather than deploying fewer larger but more expensive boxes. In other cases, clustered resources are introduced into the SAP system landscape, providing for both improved scalability and availability.

Scalability goes way beyond hardware, though. Even the OS, database, and SAP application components can benefit from buying more than you need. I have many customers who have implemented Windows-based operating systems as the foundation for their SAP systems, for instance. The capabilities of different versions within the same family of operating systems differ, though—to address greater than 2GB of RAM per process, NT 4.0 Enterprise Edition, or Windows 2000 Advanced Server is required, for example. Of course, these more capable versions are priced at a premium compared to their less capable brethren. Customers who make the decision to purchase the more capable versions without a current need for the capabilities inherent to that version are actually investing in scalability.

What all of the aforementioned cases have in common boils down to a simple delta TCO analysis, and the benefits it can provide. Each scenario includes an opportunity to spend more money on something that you may or may not actually need. Depending upon the raw dollars at stake, a quick comparison between the two cost models can make a lot of sense, though.

As you will see in Chapter 7, these kinds of scalability considerations play a common role in sizing the SAP solutions within your system landscape. It is insane to buy exactly what you think you need, because invariably a need will evolve over time that was never addressed during the SAP vision phase—appropriate attention to scalability and sizing a system for a period of time (that is, a three- or four-year life cycle, taking into account growth in terms of users, database, and so on) can mitigate this risk.

Other SAP Solution Vision TCO Drivers

Other SAP solution vision drivers impact the solution's total cost of ownership. The most prevalent are security, manageability, and operations. That is, the design and architecture of a particular SAP solution may cost substantially more (or less), depending on how these areas are addressed. The tightest security constraints, for example, will dictate deployment of firewalls, lock-down of services and ports, implementation of virus protection, and so on. Each has a particular cost that must be factored in to the total solution cost. And because a variety of firewalls and virus protection methods exist, delta analyses may be appropriate as well between different competing solutions.

A need for implanting the most manageable system, or system that is inherently "operations-friendly," will also impact TCO. There are a multitude of management/operations approaches and software packages on the market for SAP, for example. Some are operating system- or database-specific, not to mention mySAP component-specific. Others are supported only on a particular hardware vendor's product line. Still others vary in how much time is required to learn the products, whereas others are more difficult to work with when it comes to managing changes in the landscape.

- ▶ For detailed Operations and Systems Management information, see "Systems Management Techniques for SAP," p. 511 in Chapter 14.

TCO and the SAP Solution Stack

According to recent findings by the Harvard Business School ("ERP: Payoffs and Pitfalls," *Harvard Business School Working Knowledge*, October 2002), companies poured \$47 billion into enterprise systems in 2001 (not just SAP—all enterprise applications). This did not even include the expense associated with acquiring hardware, training, and implementation services. Therefore, with more than petty cash at stake, an exercise in TCO should represent a "minimum requirement" prior to purchasing the key components of an SAP solution. To this end, a number of general factors contribute heavily to lowering solution stack TCO. These include

- Procurement or acquisition costs, or the cost of initially purchasing SAP Solution Stack technology like a server, operating system, database, systems management tool, and so on.
- Installation costs, including configuration of the installed product such that it's usable.
- Management and maintenance costs, like those related to systems and process management, as well as costs related to scheduled maintenance activities.
- Training, which includes the cost of time off to pursue training as well as the actual cost of attending the training necessary to install and later support a particular component or layer of the SAP Solution Stack.

Another interesting factor that can heavily impact the overall TCO of a particular SAP system landscape and the solution stack(s) inherent to this landscape is the attention paid to *standardization*.

Standardization and Total Cost of Ownership

The goal of standardization is to minimize variety, thus simplifying troubleshooting while facilitating rapid component replacement when required. Standardization can be applied to individual systems, particular solution stacks, every component within the solution stack, and characteristics of the entire SAP system landscape.

- ▶ To read more about the role that standardization can play in reducing total cost of ownership, see “Optimum Server Configuration Best Practices for SAP,” p. 362 in Chapter 10.

Information systems evolve over time. As the demands of end users and other stakeholders force this evolution, care must be taken to ensure that the same standards imposed to create an effective enterprise environment are still enforced. This is essential to minimizing future support costs, so its importance cannot be underestimated.

However, in the real world of managing mySAP solutions post-implementation, maintaining a static SAP computing environment over any real length of time is highly unlikely. Therefore, expending extra effort maintaining exacting standards may very well cost *more* than evolving via changes to the solution stack. One area in particular—hanging on to key aging hardware assets with the false hope of keeping TCO lower—comes to mind. Does your organization plan to grow, either through new business processes, additional functional needs, or simply through acquisitions? These kinds of changes are exactly the change agents that not only improve productivity, but impact standardization.

How? Consider this: As more end users are added to the organization, or business processes are updated, or new mySAP.com components are added to the mix, incremental changes to the solution stack are almost inevitable. New high-performance hardware with new features, upgraded/new software with its own set of new features, and so on are introduced. Examples abound everywhere. For example, SAP’s Internet Transaction Server is only supported on a Windows/Intel, or *Wintel*, platform. So UNIX shops have been historically forced to introduce completely new server platforms, OSes, and so on just to Internet-enable their SAP enterprises—so much for standardization. Early adopters of new mySAP.com components typically must go with *Wintel*, too, until the various UNIX ports are completed. At the other end of the spectrum, beyond pilot projects and the like, SAP APO’s liveCache is nearly *always* implemented on a UNIX platform to take advantage of the greater RAM that can be addressed by these platforms. APO also relies on SAPDB for its liveCache database, which is a fine database by most standards, but is completely different from the

Oracle, DB2, and SQL Server databases implemented across most SAP system landscapes. Again, there goes standardization. And other factors come into play, too. The catalog server used by SAP's Enterprise Buyer Pro, for example, was limited to older releases of SQL Server and Oracle for the first year after release—during this time, I had customers who needed to plan for, staff, and manage these solution stack variations even while implementing brand new mySAP solutions.

If you are running any enterprise application today, this is not news to you. The very standard, very manageable environment you painstakingly assembled and maintained over the course of the first year or two of your implementation today probably looks nothing like the original implementation. Today, if you're like many companies, you probably run a hodgepodge of different hardware, different operating systems, and perhaps different database and other application-specific components.

My point here is simply that *standardization* should be looked upon as a goal, not as a black-and-white ironclad reality. SAP support organizations that strive for standardization, but understand that managing change within a standards-based environment represents the real world, will be the most successful at meeting the needs of their end users and customers. The good news is that ultimately these SAP-driven companies will emerge at the forefront of their industries, as their SAP-enabled business units increase both productivity and profitability compared to their non-SAP-enabled competitors.

Hardware TCO—Server Considerations

Comparing and evaluating the total cost of owning different hardware platforms has long been a mainstay in the world of TCO analysis. To this end, the solution stack is filled with areas where different platforms and standards can be pitted against one another in terms of acquisition, installation, management, operations, training, consolidation, and other costs.

NOTE

The manner in which computing assets are *financed* represents another common area of TCO analysis. I have included a "TCO Lease vs Purchase Model" spreadsheet on the Planning CD, to assist you in this regard with comparing and analyzing these two very different methods of acquisition.

In the past, I've assisted my customers with evaluating everything from the latest in network infrastructure hardware, to KVM (keyboard, video, and mouse) solutions, to even various power and cooling approaches. But one area in particular has consumed most of my time when it comes to TCO analysis—server alternatives.

A quick trip down memory lane is in order. Many companies running large SAP enterprises today invested in high-end enterprise computing gear from companies like HP, Sun, IBM, and Digital a number of years ago, leveraging each vendor's respective UNIX operating system. Although historically this type of hardware/software solution had its place in mission-critical environments, it has never been either cheap to own or cheap to operate. A number of years ago, as Intel pushed the envelope in terms of propelling inexpensive processing power into the data center, the world of supporting enterprise applications like SAP turned a corner. Suddenly customers had a choice when it came to meeting the performance requirements of a large processor and disk-intensive application. Scalability was limited initially, however, and availability did not approach the level achievable in UNIX-based clustering solutions.

Enter into the picture Microsoft and its Microsoft Cluster Server technology. Around since the end of 1996, and pioneered in support of enterprise solutions by companies like HP and Compaq, MSCS brought a higher level of availability to Intel-based solutions. In 2000, with the advent of Windows 2000 Advanced Server and Data Center Server, Microsoft gave us the most stable and easily managed clusters ever produced out of Redmond, and a large number of Fortune 2000 enterprises began seriously migrating their enterprise applications away from expensive UNIX platforms onto these less expensive platforms that performed just as well. There were and continue to be trade-offs in terms of server consolidation and manageability, but by and large the TCO story has only improved over time.

Today, Intel and other processor-based server platforms vary in every way imaginable—obvious variations include raw processing power, support for various OSs, built-in high-availability features, internal architecture considerations, form factor (size), support for various memory footprints, general upgradability, ability to add server components and peripherals “on the fly,” and of course price. These and many other factors are incorporated in the “TCO Solution Stack Considerations.xls” spreadsheet I have assembled and included on the Planning CD. Use this to help you collect apples-to-apples data that in turn allows you to compare different platform features, functions, and other capabilities in a comprehensive manner.

This spreadsheet will also assist you in terms of identifying basic benchmarking data to collect. Server manufacturers often submit their platforms to standard testing batteries like NetBench, WebBench, WebStress, SPECint (CPU2000 benchmark), SPECweb99, and so on. On top of this, SAP-certified platforms usually are benchmarked via SAP's standard R/3, BW, APO, and other benchmarking kits. I recommend collecting this data as well, in support of a server delta TCO analysis. Why a delta analysis? Because no “standard” benchmark truly represents the genuine measure of a platform's ability in the real world; this is accomplished via custom benchmarking and proof-of-concept exercises, which are covered in Chapter 16. But a standard benchmark provides excellent data points for server-to-server comparisons.

Finally, the “TCO Solution Stack Considerations” spreadsheet also identifies typical custom benchmarks that can be run to exhaustively test different server platforms. Many of the tools and approaches are free or nearly so; others can run into the tens of thousands of dollars pretty quickly.

- ▶ For more information on testing tools and approaches, including a concise “Test Tool Comparative Matrix” spreadsheet, see “System-Level Stress Testing and Pre-Tuning,” p. 574 in Chapter 16, and the relevant directory on the Planning CD.

Hardware TCO—Disk Subsystem Considerations

Similar to server considerations, we must be able to evaluate varying disk subsystems in terms of capabilities, performance, scalability, and so on. So I have included a “Disk Subsystem” tab on the “TCO Solution Stack Considerations” spreadsheet mentioned earlier. Here, you can also record the results of both standard and custom disk-based benchmarks, like those derived by Intel’s Iometer (rhymes with “thermometer”) utility, or Microsoft’s SQLIO.

I have spent nearly as much time working with comparing and testing various disk subsystems as I have with server platforms. Beyond raw performance and high-availability options, the most compelling less obvious TCO considerations that I have uncovered include

- Setup/Installation GUI—The user interface required by the storage can really make a difference in how quickly an installation or change can be completed. I prefer browser-based or Windows-based GUIs, as they tend to take a lot of the guesswork out of *how* a storage system is actually configured. The flip side of this is terminal sessions and command-line-based approaches to configuring and managing storage.
- Density—The pure number of disk drives and controllers that can be housed in a storage system also makes life easier for the SAP Infrastructure or DBA team. With more options and flexibility “in the box,” and fewer cabinets to manage, density impacts TCO appreciably. That is, a disk subsystem that can be easily expanded by simply adding more disk drives, rather than requiring the addition of new cabinets, controllers, and disk drive shelves, will prove less costly to procure, manage, and upgrade.
- Management appliances or utilities—This becomes more critical if additional disk cabinets must be added to an SAP system landscape. A management approach that allows a collection of storage to be managed holistically saves time, simplifies change management processes, and affords greater flexibility.
- In-place upgradability—A disk subsystem that allows in-place upgrades impacts TCO significantly. For example, the ability to upgrade to larger or faster disk drives, or enhanced disk controllers, or add/replace high-availability features

like pluggable fans and power supplies, helps an organization leverage its investment in the disk solution by extending its life cycle.

Some of my colleagues also submit that Network-Attached Storage (NAS) allows for lower total cost of ownership. NAS requires a dedicated storage network of its own, however, to help guarantee minimum service levels, because doing otherwise means impacting users or other services running on existing network infrastructures. This costs money. And the fact that they require up to twice the CPU processing power to deliver only a third of the throughput of direct-attached storage devices equates to higher costs, too. Finally, because NAS solutions are often “black boxes” with little to no real tuning capabilities, and are not supported by all OS/database/SAP combinations, the highest-performing disk subsystems will probably be found elsewhere.

But my own NAS experience is limited in this regard. What I’ve seen in the field indicates that there *could* be immense potential in products like HP’s StorageWorks NAS Executor E7000, which marries the benefits of both NAS and Storage Area Networks (SANs) in a common, networked storage pool. This flexible storage pool promises real interoperability, by providing both file-level (NAS) and block-level (SAN) access to SAP implementations. For companies that have successfully deployed enterprise-wide storage organizations, where a single organization is responsible for providing for the commodity storage needs of different business and technology groups, this approach may be right on target.

- ▶ For details regarding SANs and deploying other disk subsystems for SAP, see “General Storage Considerations,” p. 369 in Chapter 10.

High-availability and disaster-recovery options are much more limited today (with a few interesting exceptions related to snapshot capabilities) when it comes to NAS, however. This, combined with limited support for Microsoft SQL Server (see Microsoft Knowledge Base Article 304261), potential bandwidth issues, and general lack of performance tuning for complex SAP solution stacks, in my opinion will serve to keep NAS out of enterprise solution stacks, and back in the corporate file and print data centers, for the foreseeable future.

Operating System TCO

The OS layer gives us another opportunity to affect TCO. I have already shared one of the obvious ways to increase or decrease TCO—by purchasing an Enterprise version of a particular OS, versus the standard version. More often, though, my clients have been interested in comparing the virtues and capabilities of one vendor’s OS to another vendor’s OS in terms of the following:

- Support for various SAP products and components
- Acquisition license costs/fees
- Annual maintenance fees

- Ability to support the performance and scalability needs of the solution, including features like multiprocessing (and the number of processors that can be supported by a particular version of the OS, that is, 4, 8, 32, 64, and so on), massive RAM footprints, and so on
- Ease of administration or management, often impacted one way or the other by the skillsets of the organization responsible for deploying and managing the OS, more so than factors inherent to a particular operating system
- Ease of patching, data migration, and upgrading, especially in regard to how much downtime must be incurred
- Innate OS-level support for high-availability features like clustering, redundant network cards and disk controllers, robust security, ability to add new (*hot add*) or replace existing (*hot replace*) hardware components without rebooting, and so on
- OS support for features that further reduce planned downtime requirements, like the ability to expand disk volumes on the fly
- Value-add capabilities, such as the ability to support dynamic partitioning, directory services, network services like DNS and DHCP, and more
- Third-party support for robust backup/restore solutions, disaster recovery solutions, enterprise management applications, other hardware components that may be important to your particular solution, and so on

Additional people/process factors impact TCO, too, like the time it takes to train a new hire, or the ability to recover from configuration mistakes. And finally, the ability to support a particular SAP Solution Stack's preferred database solution is critical, as you see next.

Relational Database TCO

Quite a few database choices exist for most SAP components and products. Historically, though, Oracle databases underpinned the bulk of SAP implementations before the turn of the century. That is why even to this day I continue to be astounded by the fact that 51% of all new SAP sizings that left the doors of Compaq's SAP Competency Center back in 1999 were for Microsoft SQL Server. It is during this period that my colleagues and I saw quite a few changes with regard to database selection, most of it TCO-driven. Oracle's share of new sizings fell to 48% of Compaq's SAP customers and prospects that year, and the remaining 1.5% was split across DB2 and Informix. Compare this data to information published by Gartner in 2001, and it becomes quickly apparent that what my SAP Competency Center colleagues and I witnessed was not a fluke; TCO was driving a huge shift in demand.

Figure 5.6 captures some of the relevant findings from Gartner's study, and highlights this trend clearly.

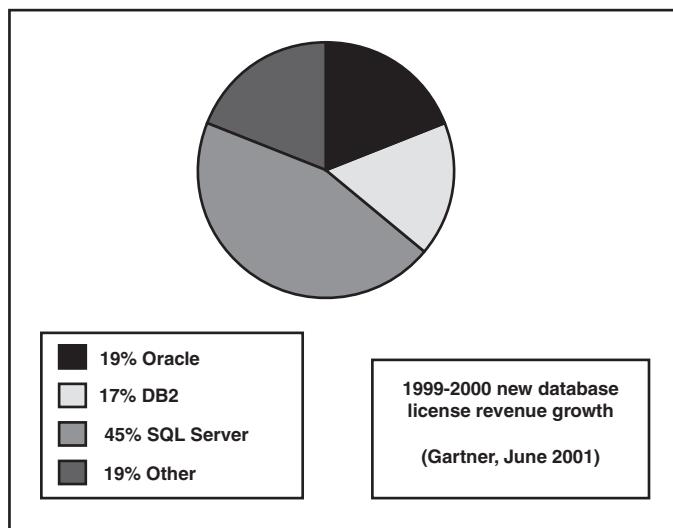


FIGURE 5.6 Beginning way back in 1999, growth in overall new SQL Server licenses began to really outstrip their competition, consistent with trends in the overall SAP enterprise space as well.

Today, according to my new colleagues in the HP SAP Solutions Center, the mix is still pretty close to equal. This is not surprising given the huge installed base that HP enjoys in the world of SAP on HP-UX; without this installed-base activity, I imagine the numbers would favor Microsoft by a wide margin. Regardless, SQL Server continues to attack Oracle's hold on the largest SAP customer environments while maintaining the lion's share of small/medium SAP sites (those with less than 500 concurrent users, or something like 2,000 named users).

Why the huge interest in SQL Server over these last four to five years? Simple—lower acquisition and management-related TCO at nearly the same performance levels of other database products. More than a handful of studies and my own experience bear these facts out. I'm a big fan of Oracle, but if we're talking pure dollars, the numbers don't lie. At something like a quarter of the acquisition costs, half the implementation costs, and half to two-thirds of the management costs, SQL Server is substantially less expensive and in most cases just as capable when it comes to SAP implementations supporting 2,000 concurrent users and less. Check out the latest TPC numbers for the HP Proliant DL760G2 if you're skeptical relative to performance(www.tpc.org)—numbers like 115,025 used to be achievable only on expensive

UNIX platforms with 64 processors. Today, eight Intel Xeon MP 2.0 GHz processors running Windows Datacenter Server 2003 and SQL Server 2000 Enterprise Edition can do the same thing, at literally a fraction of the cost per transaction. Small industry-standard servers like these support 99% of the large SAP-on-SQL Server customers *in the real world*. And larger SQL Server platforms are available as well (the Unisys E7000 comes to mind!), though a compelling TCO story for these larger platforms requires a bit more work.

I've spent the better part of the last three years supporting some of the largest SAP-on-SQL Server shops in North America. SQL Server 2000 is supporting 2,000 concurrent users (more than 6,000 named users) at one of my sites, 1,600 concurrent (7,000 named) at another site, and 1,000 (5,000 named) at yet another site. And I support quite a few other SQL Server shops running between 400 and 800 concurrent users. Because of their decision to run SAP on SQL Server, these SAP shops from a TCO perspective have quite a bit in common beyond simply their database choice, including

- These shops employ fewer Database Administrators (DBAs), when compared to other SAP/non-SQL Server shops that I support.
- They also benefit from simpler disk layouts, configuration options, autotuning, and so on, which tends to make upgrades later on less painful, and day-to-day operations less complex.
- About a third of the shops take advantage of the fact that SQL Server 2000 supports multiple database instances on a single server, thus reducing hardware-related TCO for testing, training, and development environments.
- Great attention is paid to change management, however, especially given the history of Windows operating environments' tendency to succumb to memory leaks (the result of poorly written third-party drivers, usually). As long as the solution stack remains static, and is changed only after adhering to a tightly controlled "Promote to Production" policy, there is simply no issue running W2K/SQL2K in the enterprise.
- More time is also spent addressing security holes inherent to the solution stack (the OS more so than SQL Server itself, though exceptions have been noted and addressed by SQL Server 2000 Service Pack 2 and 3, and its most recent Security Update).
- Finally, they all enjoy better than 3 nines of availability, and in some cases 4. Remember that 99.9% equates to less than nine hours or so of unplanned downtime per year—outstanding metrics in most anybody's book. These companies are successfully using SQL Server for mySAP solutions, and saving a boatload of money for the most part.

I've had experience with a few flavors of Informix, a dated version of DB2, Adabas/SAPDB, and every major release of Oracle and SQL Server since 1997. Without a doubt, I highly recommend that SQL Server be at least *considered* for every new SAP implementation where TCO and performance are the biggest factors—run through the math and see for yourself. And trust me when I say it's robust, simple to install, simple to cluster, and easy to administer.

Perhaps SQL Server's greatest TCO advantage comes under the guise of "support," however. As I mentioned previously, my largest SQL Server/SAP customers require fewer DBAs than non-SQL shops. But even more compelling, as SQL Server continues to grow in popularity across all enterprise applications (per Gartner in June 2001, SQL Server was the most popular installed relational database on Microsoft Windows platforms, with a 38% market share), access to skilled SQL Server resources will only grow. This will help keep salary rates at a reasonable level. In fact, Microsoft released a study early the next year, indicating that there were more than 85,000 trained SQL Server Database Administrators by the end of 2001.

Meanwhile, other DBAs who remain focused on the other relational database technologies will continue to become scarcer. The shrinking pool will skew supply/demand economics, as fewer new people enter the job market supporting these other database technologies. In some cases, these non-SQL DBAs will actually become *more* expensive. More often, though, I expect to see more and more Oracle and Informix DBAs shift their attention to supporting SQL Server (and perhaps SAPDB as well, given its potential to appeal to the same folks that embrace the Linux OS, for example). Regardless, if you are a good Oracle DBA, you'll have no problem picking up the knowledge you need to become an excellent SQL Server DBA.

Application Layer TCO

At an SAP application layer, beyond licensing and most other acquisition-related technology costs, total cost of ownership is most often impacted by people and process issues. Three areas in particular that are key in my experience include

- Architecture considerations, especially in regard to distributed three-tiered architectures versus central systems and other architectures
 - Deployment considerations, like *stacking*, where multiple SAP instances are installed on a single server rather than deployed across multiple servers
 - Heterogeneous versus homogeneous implementations
- Many of the approaches to minimizing TCO can be traced back to architecture, sizing, and configuration; for more detail **see "General Sizing Best Practices and Approaches," p. 230** in Chapter 7.

SAP's support for a distributed architecture allows for multiple low-cost and/or industry-standard servers to replace single-server central systems. These latter "big-box" solutions are inherently easy to deploy and manage, but can cost an order of magnitude more than their distributed solution counterparts. Of course, a few trade-offs need to be taken into account when considering a distributed SAP system. First, the management of many server, disk, and tape resources adds up to more people- or process-related costs in a distributed environment. I have one customer in particular who standardized on a relatively low-end two-processor server for all SAP application servers, ITS servers, and other infrastructure servers. Compared to the acquisition costs of more capable larger servers, their approach paid off. However, managing 40 servers instead of perhaps 15 consumes a bit of their "savings" every day. The same argument would apply to SAP clients deploying the latest "blade" servers rather than larger ones.

However, managing the SAP system landscape in these cases tasks the SAP Technical Support Organization more than a more consolidated-server approach would. This is true from the SAP Operations team up through the Database and SAP Basis/Infrastructure organizations. And in my actual customer's case, even with their highly automated SAP monitoring and reporting system, they still incur significant time penalties when ad-hoc uptime and other reports need to be generated for their SAP system landscape—not surprisingly, it takes longer to collect and analyze the raw data relevant to 40 servers than 15.

In addition to the extra management overhead, change management and subsequent upgrades to the solution stack take more time, given that each individual server is subject to firmware, hardware, OS, Service Pack/patch, database client, SAP kernel, and other per-server updates. Licensing is also a bit more complex than it might otherwise be, again given the sheer volume of OS and database client licenses that must be maintained and managed. Finally, tape backup/restore processes tend to morph into more complex "server restore" processes, as it is easier to rebuild an application server through an automated disk imaging or similar process than to actually back it up and restore from tape in the event of an emergency. This then adds another burden to the SAP TSO, though—maintaining and updating server images for each type of server after each change release or change wave.

The potential TCO benefits of a distributed architecture are many however, including

- The overall solution stack is oftentimes not only cheaper to acquire, but still cheaper to manage despite the drawbacks.
- High availability through redundancy (less impact to the overall solution if a single application server or redundant Web AS, ITS, or other server suffers from unplanned downtime).

- Scalability is easily addressed at the application layer by simply adding another application server (and calculating/addressing the performance hit passed on to the database and central instance servers, of course).
- Excellent load balancing approaches can be put in place when multiple application servers, or ITS Wgate servers, and so on are present.
- Specific workloads can be easily addressed by multiple servers. For example, many SAP shops employ two or three dedicated batch servers in their mix of application servers. Other shops break out update work processes onto separate servers, or in some other way dedicate specific functions to a particular set of servers.

The second key area for minimizing SAP application-layer TCO is in regard to stacking. *Stacking* refers to the ability to install and run multiple SAP instances on a single server. For example, in one recent project of mine, my colleagues and I installed three pilot instances of R/3 on a single HP ProLiant DL580 running SuSe Linux and Oracle. You might also want to consider virtual partitioning or virtual machine approaches, like that provided by VMware. This alternative to stacking lessens the number of servers and is just as manageable as any environment where one SAP instance is installed per OS installation. In addition, though, virtual partitioning allows you to chop up a large server in terms of hardware resources. Thus, it provides the flexibility of allowing you to install an SAP instance based on resource need, rather than based on the maximum capacity of the platform.

Either approach can save a bundle in hardware costs when it comes to deploying pilot, sandbox, development, testing, and training systems. Stacking or virtual partitioning is usually NOT the preferred way to go for production environments, however, as immediate or low-level performance tuning, future capacity planning, and general reactive troubleshooting become more complex.

Finally, whether an SAP system landscape consists of the same or different server environments represents the third area affecting TCO. It probably surprises no one that designing, deploying, and managing a diverse landscape costs more time and money than doing the same for a homogeneous system landscape. However, if the SAP TSO is skilled in supporting both UNIX platforms and Microsoft-based OS's, for example, implementing a heterogeneous solution for SAP in the past has significantly decreased both acquisition and recurring maintenance costs. This type of solution seems to manifest itself most often in shops where a large investment has been made in high-end UNIX or IBM S/390 or AS/400 gear—or where the customer may simply be locked in to the technology and want to put it to use in the form of an SAP database server. Surrounding these large database servers with relatively cheap Wintel servers can be quite effective and a good TCO move. And given the fact that the SAP connection between the database and application servers is only a simple

TCP/IP connection, integration and management of the solution poses no real challenge, either.

Today, given the power of Winet and low-cost UNIX-based solutions, I see more and more customers actually moving *away* from heterogeneous solutions, however. I believe that this approach served a need quite well for a number of years—the need to reduce the total cost of ownership of SAP R/3 and to a lesser extent BW in the largest of environments—but now that need is served equally well by homogeneous systems. One interesting twist to this trend might include the role Linux plays in the near future, though. That is, I won't be surprised to show up in more than a few data centers some day soon and see brawny Sun, IBM, and HP mainframe-class database servers surrounded by highly capable though very inexpensive SAP-on-Linux application and Web servers. The benefits in terms of OS standardization and access to massive processing power will further reduce TCO while simultaneously elevating Linux higher in the enterprise data center. It is this kind of activity that will certainly drive SAP infrastructure and overall solution stack upgrades, too, covered next.

Solution Stack Upgrade Considerations

While on the subject of solution stack TCO, consider the following. Upgrades are inevitable, and well-timed and well-executed upgrades can be a wonderful method for extending the life cycle of a stack layer or component. But upgrades get expensive quickly. Every upgrade, according to industry experts, equates to a minimum \$200-\$400 in delta service expense, or incremental expense associated with “touching” a production resource rather than replacing it. Add to this the costs associated with the following:

- Determining *which* upgrade hardware or software part numbers to order adds to the upgrade cost.
- Placing and tracking the upgrade order. Because upgrade orders tend to be one-offs, they typically cost more than placing and tracking a new order.
- Paying for the update or upgrade component(s), compared to a new component—this nearly always favors the upgrade in terms of cost.
- Coordinating the overall upgrade effort, compared to coordinating a simpler “replacement” effort, costs more.
- Managing the change control process associated with upgrading or updating a solution stack component, including increased costs associated with mitigating the risk of dealing with unforeseen compatibility issues not uncovered during testing.

- Safeguarding valuable data during the actual process of upgrading, and making that data available again as soon as possible (as upgrades are typically more time-consuming than employing a replacement strategy).
- Installing a piece of hardware like an additional processor usually costs more than buying the server already fully populated with CPUs. Why? The cost of service upgrades (that is, buying a processor for a two-year-old server) can be expensive, added to the cost and complexity of scheduling and taking down-time.

It becomes quickly apparent that the costs of a seemingly simple upgrade can well exceed the cost of a straightforward replacement strategy, even when the procurement costs of the upgrade are substantially lower than the procurement costs of a replacement. My recommendation is to be wary of upgrades and “do the math”—identify and recognize the costs of an upgrade versus a direct replacement, evaluate the deltas, and in the end make an informed decision.

Other Solution Stack TCO Considerations

Finally, before leaving the world of technology-based SAP Solution Stack TCO behind, a quick look at some final technology considerations is in order:

- Backup and restore capabilities continue to top my list of “things that are too complex.” I am amazed at how many of both my new and long-time SAP customers are unsatisfied with their backup and restore products and strategies. Total cost of ownership is certainly important, but given the critical role that a good backup plays in regard to disaster recovery, I’m inclined to say count yourself lucky if your backup strategy works well at all. Beyond that, disk storage systems that support centralized tape backup (like SANs), and the efficiencies that can be derived by investing in one or a few large tape libraries rather than discrete one-off tape drive solutions, decrease TCO and simplify life with SAP in general.
- A variety of SAP-aware management applications and tools exist. I cover many of these in more detail in Chapter 14. As far as TCO is concerned, however, the goal is clear—to replace the mundane tasks of operational reporting and tracking with a package that doesn’t require vacation time, fail to show up on Mondays until noon, or forget to collect data one day. In addition, the best enterprise packages will collect and filter against *events* up and down the entire SAP Solution Stack, both reactively and proactively looking for problems and failures. Finally, the infrastructure required to perform these management tasks should be reasonable, and in my opinion the management application itself should refrain from making internal changes to the mySAP.com component being managed.

- Tools that provide connectivity to other systems will eventually become de facto “production” interfaces. In my experience, this means that attention needs to be paid up front to bringing in tools from reputable vendors with support organizations that have an understanding of how their tool or utility works with SAP. Free or “shareware” tools might be nice for a few months, but inevitably some kind of support issue will crop up. Again, do the math—a single support issue, including the time and potential revenue/productivity lost—can drastically impact TCO, especially after changes have been made elsewhere in the solution stack.
- The deployment of SAP’s user interface, the SAPGUI, can impact TCO substantially as well. One of the most compelling solutions lies in leveraging SAP’s WebGUI rather than installing a copy of the traditional SAPGUI on each user desktop. In this way, the desktop is never changed and therefore many fewer user support calls are initiated. Another compelling solution could include Citrix MetaFrame and other similar services designed to host the SAPGUI remotely back at the data center, rather than at each user’s desktop.

With the technology factor of TCO wrapped up, we can now move to the next two major TCO analysis components—people and processes.

Lowering TCO Through People and Processes

Although the procurement and deployment of technology plays an obvious role in total cost of ownership analysis, you must remember that people and processes play equally key roles. No one area is greater than the other two, as is clearly illustrated in Figure 5.7. Instead, a balance must be achieved.

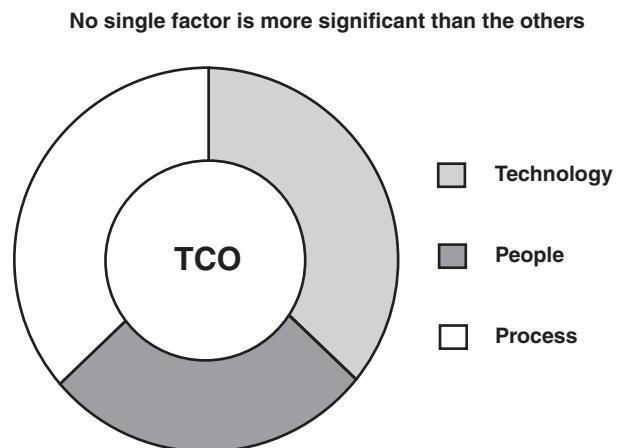


FIGURE 5.7 Technology, people, and processes all play key interdependent roles in evaluating the total cost of ownership of a particular solution.

It is not uncommon, unfortunately, to be inclined to carefully estimate people and technology costs, only to downplay the process costs of a particular solution. Why? Because it's *easy* to pull together the one-time implementation costs associated with technology, but usually a bit more difficult to quantify people expenses, and seemingly impossible to quantify process costs/savings.

NOTE

To make life easier when tracking people costs, I've included a "Simple Project Staffing Costing Matrix.xls" document on the Planning CD. Though not a TCO tool in the strictest sense, it has served me well in tracking costs, timelines, and contact data for small projects.

Attracting and Retaining Talented SAP TSO Members

A key TCO concern when it comes to people is the cost related to employee hiring and retention. In a company where the SAP Technical Support Organization lags dramatically behind IT in general, especially in terms of technology, the ability to attract and retain high-quality IT people can be severely hindered. What highly motivated PC Technician or Systems Engineer wants to continue to support the SAPGUI on a Windows 3.x desktop? What database administrator wants to work in a SQL Server 6.5 environment any longer than necessary? True, there are plenty of qualified individuals, but as in any other IT organization, the most motivated and talented individuals typically propel themselves forward as they have always done, embracing and learning new technology along the way. Meanwhile, those individuals happy playing with the legacy solution stack components I just mentioned are probably the same people you had problems training and moving out of your IBM MVS/XA data center 10 years ago.

Not only is finding these older skillsets sometimes difficult, but the annual recurring costs to retain these skills can actually be higher than the costs of more mainstream solution components. For example, as you see in Figure 5.8, if you are adamant about maintaining a truly vintage solution environment (like SQL Server 6.5 or perhaps an aging server platform), or insist on bleeding-edge technology (like the latest release candidate of a new 64-bit enterprise version of SQL Server, or the latest server released last week), the people-related support costs will be higher than the costs associated with supporting the mainstream product (like SQL Server 2000, or a server platform that has been available for 6–12 months).

So take heed—it is nearly always cheaper to keep the incumbent DBA (or SAP Basis guru, client deployment specialist, and so on), rather than hiring and training a replacement. My recommendation is nothing short of common sense. Keep your talented IT people as long as you can, and have a backup plan. The backup plan should include developing a good relationship with local contract staffing and

consulting firms, and leveraging the assistance of hardware and software manufacturers as needed.

- ▶ For more information on retaining your valuable SAP Technical Support members, see “How to Retain your SAP Staff!” p. 298 in Chapter 8.

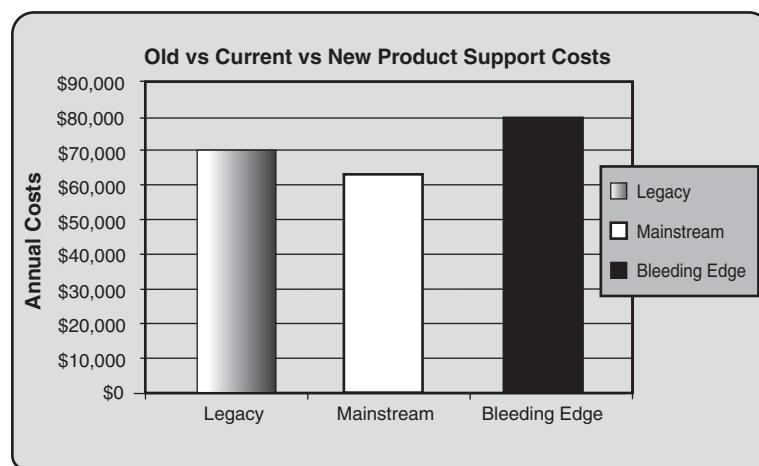


FIGURE 5.8 From the SAP customer’s perspective, the sweet spot for SAP IT skills reflects expertise in stack components that are mainstream; both older and cutting-edge expertise tend to cost more.

Maintenance Costs

Maintenance costs exist throughout the SAP system landscape, and up and down the entire SAP Solution Stack. Server and disk subsystem hardware, operating systems, databases, and mySAP.com components all usually incur annual maintenance costs (one interesting exception, of course, is SAP’s SAPDB database).

Also, consider the costs of hardware spare inventories, especially after five years, the period when many hardware manufacturers quit manufacturing or remanufacturing spare parts. And then consider the fact that, as your ability to stock these spares decreases, the probability of a component-level hardware failure increases. This explains in a nutshell why so many enterprise and other solutions tend to undergo technology refreshes every three years or so—not to mention that most manufacturer’s warranties expire after three years as well.

When it comes to hardware spares, any component or part that is deemed critical to production needs to be carefully considered for “sparing.” Usually this means onsite stocking of any part that represents a single point of failure or is otherwise deemed critical.

Instead of stocking a part that sits in a cabinet, however, other methods of stocking are often employed. A key benefit of standardization is the use of identical components throughout test, training, and sandbox environments, for example. Some of my customers even go so far as to house onsite and offsite spare servers and disk subsystems, sometimes “cold” (thus requiring manual intervention to introduce these assets into production should a failure occur within the production environment), but more often “warm” (ready to be pulled into production automatically upon failure of a critical component).

Stocking is also often handled as part of the service-level agreement process, too. In these cases, fee-based arrangements are made with hardware or third-party hardware sourcing partners to provide a part onsite within perhaps four hours after being contacted. Such an approach makes a lot of sense in metropolitan areas where the overhead costs borne by the sourcing partner related to maintaining such a facility or service can be spread out over hundreds of enterprise customers.

Financial Terms

Financing alternatives, including leasing, offer an attractive method to retain cash in-house for profit-generating activities that would otherwise be tied up in fixed assets. Leasing in particular allows a company to not only stay near the technology edge, but also to let them treat the expenditure as a tax-deductible operating expense (usually).

For companies with little capital, leasing ensures that they have access to the newest technologies today. Even today, leasing remains an attractive method of financing SAP infrastructure implementations throughout businesses of all shapes and sizes.

It should come as no surprise that when it comes to TCO, leasing versus purchasing costs must be considered. As I said before, to assist you in this analysis, I have included a cursory Excel spreadsheet geared toward this exercise on the Planning CD.

Operations and Systems Management Costs

Although it’s admittedly difficult to quantify the operations and systems management costs of a solution that is only being planned at this stage, you still need to take a stab at it. Fortunately, questions like the following can help you put together a delta analysis specific to management costs, as discussed previously:

- Does each layer in the SAP Solution Stack support “manageability” of some degree or other? That is, are there tools, utilities, or management applications available to monitor and manage the various hardware, OS, database, and mySAP.com solution choices you are leaning toward for this particular solution stack?
- How much does each tool, utility, or management application cost to acquire?

- What annual maintenance fees can be expected of each tool, utility, or management application?
- Do the various management tools and approaches work well *together*? For instance, is there a common management framework that can be leveraged, allowing each tool to “snap in” to the framework? Or do any of the tools require specific agents or other components that conflict with other tools’ agents?
- Does each tool require a dedicated hardware platform upon which to run, or can a common management platform or console be used?
- How long will it take for the SAP Operations and other SAP TSO team members to learn each tool, utility, or management application?
- When it comes to access to professional support/services particular to each management product or approach, what is available? And what are the costs of these subject matter experts?

As for other costs, support for the following needs to be understood:

- Backup/restore-related tasks and responsibilities, like the time required to learn and actually use a particular backup solution, monitoring the backup/restore processes, swapping and managing tapes, and so on
- Testing disaster recovery and high-availability processes
- Supporting change management processes as necessary
- Connectivity to other systems
- Client access approaches via the SAPGUI, WebGUI, and other user interfaces
- Incremental facilities costs (if, for example, a particular server or disk subsystem platform requires power, cooling, or other facilities infrastructure different from the data center’s norm)

For additional management and operations tasks that may need to be evaluated, refer to Chapter 14.

TCO Risk and Risk Factors

A value is often placed on the risk inherent to a particular solution stack or process. Given that risk is usually difficult to quantify, a delta analysis is appropriate here, too. In this way, you can compare the risk of one method or approach or solution to the risk of employing a different one, and quickly determine the cost difference. I like to perform this exercise using something I call *risk factors*. Risk factors are

numbers weighted to reflect the relative risk inherent to changing a particular solution stack layer component.

For example, consider the following scenario with Global Process and Chemical Corporation, or Global, a fictional entity. Global is in the process of evaluating a number of different solution stack alternatives when it comes to their planned APO solution. They are running SAP R/3 4.6C and BW 3.0B today, so they have expertise in the following:

- A number of different mid-range HP Unix server platforms
- Both EMC (used by SAP R/3) and HP StorageWorks (used by SAP BW) storage systems
- Two recent versions of the HP-UX Operating System
- Two recent versions of Oracle's database product
- SAP Basis layers 4.6 and 6.20

The preceding list represents a baseline of sorts, and will be used to calculate how different changes to the solution stack affect risk. Obviously, Global incurs the lowest risk if they decide to leverage their talent and expertise represented by this baseline. Risk increases as Global deviates from their core competencies—perhaps not a whole lot in some cases, but the idea behind a delta analysis is to identify the impact that each change has on the project's cost. Thus, as Global considers alternatives perceived to be lower cost, like a Windows 2000 platform, or implementing IBM DB2 on HP's ProLiant line of servers, they need to factor back in the risk associated with changing their standards and everything that such changes demand.

What is *out of scope* for this exercise is an analysis or comparison of the procurement or management costs associated with each solution stack layer/component. Remember, this should have already been covered during the solution stack implementation analysis.

In Global's case, let's take a look at different solution stack combinations and alternatives, and assign a risk to each solution layer from 1 to 10, where low numbers are less risky than high numbers. Note that a zero is possible, too—in this case, there would be no change to a particular solution stack layer or component.

- Overall, in my experience the risk in changing server platforms equates to something like a risk factor of 1 if the same server line is brought in, a 3 if a new server line is introduced from the same hardware vendor, and a 5 if a completely new server vendor is called upon to deliver the new platform. If the server platform is changed, risk is increased more if the server vendor changes as well. That is, it's less risky for Global to bring in one of HP's UNIX-based servers than, say, IBM's or Sun's UNIX offerings.

- If a new storage or disk subsystem is introduced, more risk is incurred than in introducing a new server platform. I go into quite a bit of detail in this regard in Chapter 10. Suffice it to say here, though, that the disk subsystem represents the key performance and availability challenge within the solution stack. Risk inherent to introducing a completely new disk subsystem rates a 10 in my book, due to potential issues related to complexity, support, performance tuning, and troubleshooting. Marginal differences in storage (like a new disk drive standard, or incrementally updated disk array controller) rate a 2. A new storage line from the incumbent storage vendor rates a risk factor of 6, however.
- If a new operating system is introduced, risk ranges from 0 (same OS with the same patch or service pack levels), to a 2 (for introducing a new flavor of UNIX similar to the current UNIX standard, like HP-UX and Tru64 or Solaris), to a 4 (for introducing a very different UNIX flavor from the standard, like AIX), to a 6 (for completely different OSs, like Microsoft Windows 2000 Server compared to HP-UX 11i).
- The risk inherent to introducing a new database is also one of the highest. Version changes (that is, Oracle 8i to 9i) represent only a 1, but completely changing vendors (from Oracle to Microsoft's SQL Server or IBM's DB2) rates an 8. This is because the database selected for SAP impacts so many other areas—the SAP basis layer changes, backup/restore processes and utilities are changed, DBAs require retooling, the physical layout of disks differs and therefore needs to be addressed, and so on.
- Finally, establishing a new SAP Basis standard introduces its own risks. Minor release changes are nearly negligible (like 4.5x versus 4.6x). Major release levels incur more risk, though, as new approaches to installing, integrating, and managing the basis layer need to be taken into consideration. The introduction of the 4.x basis layer represented a fairly significant change from 3x, for example, rating a 3. The jump from 4x to 6x is a bit riskier, rating a 4.

What about multiple changes? For example, if Global is partial to the low acquisition and management costs inherent to SQL Server 2000, they will also need to bring in a new server platform as well as a new OS! Such a decision compounds the risk of merely swapping out the database layer. Add to this scenario the decision to go with the newest gear available in virtual Storage Area Networks, and practically the entire solution stack represents serious deviations from Global's standard.

So, to get back to the math, if the most risk-averse strategy for Global is to go with a solution stack for APO as similar as possible to their current R/3 or BW standards, such a low-risk decision would rate a 4, which is simply the sum of the various risk factors, as detailed in the following list:

- Global's current exact server platform is no longer available. The updated platform featuring faster processors and more memory is very close, however. This rates a 1 in terms of risk.
- Global's current disk subsystem standard is still available. However, the standard disk drive size and form factor is different, and the firmware on both the disk drives and disk controllers has been updated as well. These minimal changes rate a 2.
- Global's OS standard is also still available, though the server platform requires an updated OS patch level. This rates a risk factor of 1—a bit more risky than the same exact OS/patch level combination.
- Global's database standard remains 8.1.7, and is not impacted by either the minor updates to the OS or the requirements of the release of APO to be implemented. Thus, its risk factor is zero.
- The particular release of APO to be implemented leverages the same basis layer as that employed by R/3, so again we have no impact when it comes to risk—another zero.

A risk factor of 4 represents a very low-risk way to go, in light of the fact that we could be looking potentially at a 50 (maximum of 10 for each of the five key solution stack layers). On the other hand, moving to a brand-new vendor's server and disk subsystem platforms, and introducing Windows 2000 with the same Oracle 8.1.7 release would give us a 21 ($5 + 10 + 6 + 0 + 0$), more than 500% riskier than staying with a current updated platform. An even riskier decision would be to bring in the new servers, disk subsystem, and OS, and top it off by supplanting Oracle with SQL Server 2000. Such a solution stack would represent a 725% delta in risk compared to the baseline ($5+10+6+8+0$, or 29 as compared to 4).

Other layers can be added to my master list of five, to create custom TCO Delta Analyses. For example, a Web/Intranet layer would need to be factored in for many mySAP.com components, or if a decision is being considered to deploy one of SAP's Web-based GUI interfaces. And an integration layer might need to be factored in if a requirement existed here, too.

Similarly, factors that help to mitigate the risks identified so far can be included in the analysis, too. Things as straightforward as obtaining training to support the new components, obtaining contractual access to expert consulting expertise in the event of an emergency, and so on can easily be factored in to reduce the risk unique to each component or layer.

To facilitate your own TCO risk analysis, I've included an Excel spreadsheet entitled "TCO Risk Factors" on the Planning CD. Before you begin to use this simple tool,

though, I need to conclude our risk discussions by asking that you don't read too much into the raw numbers in and of themselves. That is, whether a number is a 4 or a 21 or a 29 means little. What matters is the difference between the numbers, the deltas. And what *really* matters is how these deltas impact the overall TCO story—the end-to-end TCO big picture for a particular solution stack or approach.

Updating the RFI, RFP, or Internal Knowledge Repository

Remember, the goal of TCO analysis is to narrow your solution stack choices to a set of options that satisfy both your business units and your financial folks. When the analysis is complete, the information and knowledge gleaned from each TCO exercise usually makes its way into the RFI or RFP to be eventually published to the various solution stack vendors and partners.

If the RFI/RFP approach is not to be used, this knowledge must still be captured by the organization that wants to implement the mySAP solution. This is because it will be used to help you complete the various hardware and software sizing questionnaires you will run into later—I discuss this process in detail in Chapter 7. For our purposes here, though, it is important to document

- Assumptions, or why you believe or think in a certain way. This includes financial assumptions, people/skillset assumptions, process assumptions, and any assumptions regarding the potential make-up of the solution stack.
- Comprehensive TCO analyses that have been completed up to this point, which should include concrete results and findings (again, backed up by documented assumptions).

All of this data must find its way into a knowledge repository of sorts, and be treated as key data points as the project moves into the next phases of planning.

Documenting TCO Observations and Key Drivers

A number of methods exist for recording and managing your findings and observations thus far. I use the term *knowledge repository* generically. Some of the methods I have seen include

- A series of file shares, divided into various phases or functions. Although this method works, and is the most popular I've seen, it presents typical document management challenges solved by the next approach.
- A formal document management and collaboration application, like Microsoft's SharePoint Portal Server. In this way, data can be shared via a common project *portal*, facilitating meetings, information dissemination, version control of various project documents, and so on.
- A Web site that allows for document posting as well as read-only access.

Of course, other methods exist. I have even seen entire chunks of SAP projects managed via a single Excel spreadsheet with many tabs, each dedicated to a particular function or phase. At another customer site, much of the data relevant to sizing and configuring the technical piece of the solution was maintained in a database, and attachments and other information were maintained in a set of file shares. In the end, the easier it is to access, share, and manage the project data, the more smoothly the project will be positioned to run.

Tools and Techniques

Although excellent tools like Gartner's TCO Manager for Distributed Computing can be used to perform end-to-end holistic TCO analyses, I have provided a number of simple tools and related documents on the Planning CD.

- TCO Simple SAP Landscape Analysis.xls—This simple Excel spreadsheet helps you to identify and record where costs are incurred throughout the SAP landscape.
- TCO Solution Stack Considerations.xls—This document is used to compare and track the costs of various solution stack alternatives.
- TCO Lease versus Purchase Model.xls—This model is designed to help you analyze the total cost of ownership when it comes to leasing versus purchasing SAP infrastructure assets.
- TCO SAP Outsourcing Analysis.xls—This analysis is an excellent resource for estimating potential savings and trade-offs.
- Simple Project Staffing Costing Matrix.xls—Though not a TCO tool per se, such an approach facilitates tracking the people-costs of small SAP implementation projects easily.

Of course, each of the figures found in this chapter can also be found in electronic form on the Planning CD, in Microsoft PowerPoint format.

Summary

One of the most common schools of thought in Corporate IT today goes something like this: "If we buy and implement the infrastructure for a new mySAP.com component, and retain the assets for three to five years, we will save substantially in terms of overall SAP TSO support costs, help-desk costs, training costs, and minimized peer support as compared to outsourcing everything." Many others might disagree with this statement, backing up their vague claims with "It depends." What I hope I have accomplished in this chapter on total cost of ownership analysis is to provide you

with both an approach and the tools necessary to answer these kinds of questions specifically for your own SAP implementation or upgrade project.

Now, keeping in mind everything you have studied over the course of this chapter, you should be fully prepared to address the final and perhaps most critical characteristics of an SAP solution—high availability and disaster recovery. In doing so, I will wrap up the bulk of the “best practices SAP implementation planning” we have covered together over these first few chapters in the book, pushing forward to the next major milestone—solution sizing.

6

Identifying High Availability and Disaster Recovery Requirements

Introduction to Availability

Before you tackle sizing an SAP solution, you must understand the availability requirements of the system, or the amount of time that the system needs to be available to satisfy the needs of its users. In the world of SAP and other enterprise applications, this is referred to as *high availability*, or simply HA, to reflect the critical nature of the systems themselves. It's not enough to be available most of the time—the enterprise solution must be available nearly always, and the degree of availability depends on the requirements of the business.

This high-availability requirement manifests itself in a number of ways. The architecture of the solution, the solution components themselves, practices, processes, and much more all come into play. In the end, HA protects mission-critical business processes, safeguards the data supporting these business processes, and also provides flexibility with regard to maintenance windows. Ultimately, the goal of HA is simple—to increase the time that your mySAP.com solution component is up by decreasing both planned and unplanned downtime.

HA alone does not address the needs of disaster recoverability or disaster resilience, however. Tackling *disaster recovery* (DR) requires taking high availability to the next level. That is, where HA is usually concerned with addressing relatively small downtime outages that range from

IN THIS CHAPTER

- Introduction to Availability
- Addressing Availability Early in the SAP Project
- Determining Your Required Level of Disaster Tolerance
- Remedy Single Points of Failure in the Stack
- SAP Data Center Infrastructure SPOFs
- SAP Server SPOFs
- Disk Subsystem Single Points of Failure
- Single Points of Failure Inherent in the OS
- SPOFs and the Database Layer
- mySAP.com Single Points of Failure
- SAP General Availability and DR Best Practices
- A Final Look at the SAP System Landscape
- Tools and Techniques

seconds to perhaps many hours, DR focuses on how to handle downtime that lasts many hours to days or perhaps even weeks. Thus, whereas HA might simply represent short-term *failover solutions* or incorporate tactical practices and system characteristics designed to increase general availability, disaster recovery speaks directly to how a company continues to do business in the aftermath of a disaster. This is one reason why DR is also often referred to as business continuity, and DR solutions themselves as business continuity solutions or approaches.

Regardless of the label we apply to it, the goal of DR is clear—to protect an organization's production data and business processes from interruptions beyond an acceptable threshold. Our challenge, then, is to determine precisely where this "threshold" lies for a particular business organization—the intersection of long-term uptime cost/losses and budget constraints, as you see in Figure 6.1. Here, Organization A reveals little tolerance for downtime, given the huge losses it suffers after just a few days of downtime. Organization B, on the other hand, can tolerate a longer period of downtime before the costs, in their opinion, become unacceptable. When these production downtime tolerances are identified, our next challenge is to mitigate system failures such that the production environment never reaches that threshold.

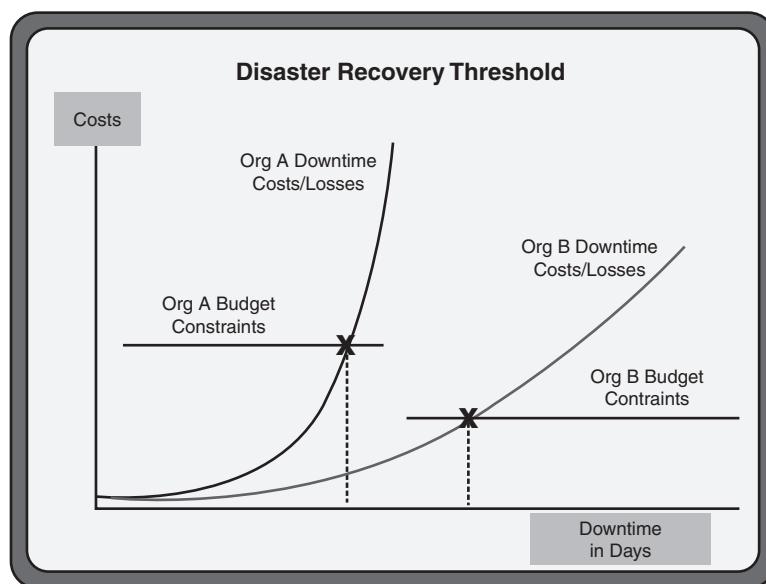


FIGURE 6.1 The disaster recovery threshold varies for different organizations. Here, Organization A has little tolerance for downtime, whereas Organization B can tolerate quite a bit more.

Throughout this chapter, we will together examine the many facets of HA and DR—how each can be addressed via the different layers of the SAP Solution Stack, when it makes sense to address these from a planning perspective, HA and DR best practices and approaches, advantages and disadvantages of these different approaches, and real-world solutions and issues. To keep things simple, I will use the general term “availability” when referring generically to either HA or DR. Before we get ahead of ourselves, though, let’s start by taking a quick look at the common reasons that companies suffer downtime.

Causes of Downtime

According to a Gartner Group study published in 2001, the top six reasons for enterprise systems *unplanned* downtime, in descending frequency or popularity, are as follows:

- Application failure
- Operator error
- Operating system failure
- Hardware failure
- Power outages
- Natural disasters

This might surprise some of you, but it’s pretty much in line with what I have seen when I perform SAP High Availability Reviews, a service that I developed, part of which entails collecting historical data as to why a particular SAP customer suffered from unplanned downtime.

For those interested, the most common cause of *planned* downtime should be no surprise, as it has not changed in years—system maintenance, the bulk of which is related to the time required to perform offline database backups.

With the common causes of downtime identified, let’s identify how high availability and disaster recovery are similar.

Similarities Between HA and DR

First, HA and DR both seek to reduce unplanned downtime to acceptable levels, and address planned downtime as well. Here are some more specific similarities:

- Both HA and DR require an exercise in Total Cost of Ownership (TCO) or Return on Investment (ROI), to determine the amount of unplanned downtime that is allowable before some kind of alternative action or solution (referred to as system “failover,” or simply failover) must be manually or automatically executed.

- HA and DR solutions add more complexity to the solution stack, both technology-wise and from a business support and IT support perspective.
- Both require in-depth training on the part of the SAP Technical Support Organization, to ensure that each HA/DR solution is understood, deployed, and managed correctly.
- Both HA and DR can be addressed or implemented at many different layers of the SAP Solution Stack.
- HA and DR are in essence insurance to the business—incremental budget money spent on the mySAP.com solution, which will hopefully never be needed, but represents money well spent “just in case.”
- Both HA and DR selections therefore tend to drive the architecture of the solution stack. In fact, they very much *restrict* the options of the stack, as only precise combinations of solutions and components work together to create a highly available system.

With regard to this last point, a particular disk subsystem vendor’s approach to HA might require a specific operating system or version of the underlying mySAP.com database. Or a certain level of disaster recovery might only be available from a hardware vendor’s particular line of disk subsystems. In either case, the need for a specific level of availability limits the choices you have in regard to the SAP Solution Stack—you in effect “cross off” many of the lower-level solution stack alternatives in favor of fewer, more highly available options.

How HA and DR Differ

Ultimately, both HA and DR beg the question “What is my business exposure when my SAP system is down?” Further, both high availability and disaster recovery seek to mitigate the risks and costs inherent to this downtime. Key differences exist, however:

- HA tends to be tactical in nature, whereas DR tends to protect the organization from strategic loss. Thus, HA focuses on mitigating component failure within the context of a layer of the solution stack (for example disk failures, server failures, network component failures, and so on). The focus of DR is on things like facilities and processes that come into play during the total failure of a data center, however.
- HA is typically implemented within the SAP system architecture or system landscape, whereas DR is nearly always architected “outside the box”—outside of the primary SAP data center facility, and certainly outside of specific internal server and disk subsystem high-availability options.

- HA usually involves SAP Solution Stack resources in one physical location, whereas Disaster Recovery nearly always entails a second data center site or approach to meeting the business/processing needs of an organization.
- Disaster Recovery tends to require very specific SAP Solution Stack components. That is, DR approaches tend to be based on a particular technology deployed at one or a few layers of the stack, whereas high-availability practices exist at every layer of the stack.
- DR recovery tends to be complex and time-consuming, often requiring a planned and time-consuming failback period; HA recovery is much simpler.
- DR solutions can be quite expensive to procure, not to mention implementation and support costs.

Another key difference exists—hopefully, the disaster recovery solution is rarely utilized, whereas high-availability approaches are exercised quite often. For example, nearly all SAP customers leverage their investments in HA (whatever that might entail) to more rapidly complete planned system maintenance, support change releases/change waves, and support regular activities like offline database backups and so on.

Perhaps the biggest difference between HA and DR is simply this—DR represents a huge medical insurance policy to the business, and HA a much smaller one. As with individual medical insurance policies, you hope to never really *need* to exercise your disaster recovery site and DR processes. But they are there if you need them. The trick is to buy the right “plan”—to purchase the proper DR solution required by the business. Why? Because the last thing you want to find out after a disaster hits your data center is that you signed up for a nickel and dime HMO when you really needed a class A policy.

Addressing Availability Early in the SAP Project

Because availability requirements drive the nature of the solution, these requirements need to be understood as soon as possible. Remember, HA and DR requirements limit your choices in regard to certain layers in the SAP Solutions Stack. If you make the mistake of purchasing key solution stack components *before* your availability needs are made clear by the business, you risk having to toss out substandard pieces of the stack before long. So before you go off and buy servers, database packages, and operating systems, work with the system’s intended end users and other project stakeholders to determine exactly the level of availability needed—what we often refer to as the “nines” of availability.

Determining HA Requirements—The “Nines”

High availability is often measured in terms of percentages that a particular system is “up” or available to be used productively by its end users. Most often, this percentage represents the number of minutes or hours in a year, less the amount of downtime realized by the system due to unforeseen issues—unplanned downtime. In other cases, some SAP customers add up both planned and unplanned downtime when they talk about high availability. Neither is necessarily a better approach than the other. Rather, it’s important that an organization is simply consistent when discussing and measuring their availability goals.

The nines, therefore, are simply percentages of uptime like 99%, or 99.9%, or 99.99% of a year, where a year is defined as follows:

- 8,760 hours (365 days times 24 hours in a day)
- 525,600 minutes (8,760 hours times 60 minutes in an hour)
- 31,536,000 seconds (525,600 times 60 seconds in a minute)

A solution that’s designed for five nines of availability, for example, will be architected to withstand all but the most horrific of disasters, suffering from less than six minutes of unplanned downtime throughout the course of an entire year. Refer to Figure 6.2 for a simple matrix reflecting the most common measurements of availability.

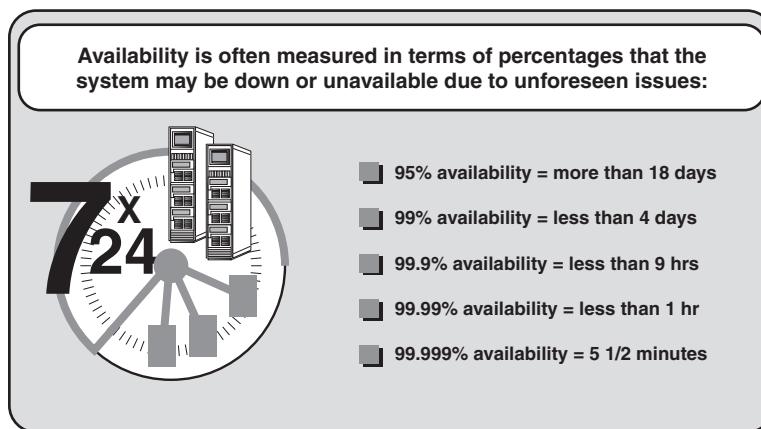


FIGURE 6.2 Note the “nines” of availability and what each equates to in terms of actual downtime observed by a system.

Throughout this book, it may be helpful to refer back to Figure 6.2 when the topic of four or five nines of availability comes up.

Availability Planning—Documenting Requirements and Key Drivers

At this point in our HA and DR discussions, you need to capture business requirements—the key business drivers that will push the SAP technical support organization, and the solution architect(s) specifically, toward designing a particular SAP solution stack. I recommend that the senior solution architect interview and work directly with the steering committee in this regard. The makeup of the steering committee should reflect stakeholders up and down the end user, IT, and executive management ranks:

- The chair and other senior executives/managers, typically responsible for the productivity of large pieces of the end-user community
- The key representative from every functional area, such as finance, HR, manufacturing, logistics, and so on
- A senior representative of the chief information officer, or the CIO himself
- The company-internal project manager
- The manager or director responsible for managing the enterprise computing systems upon which the company currently relies

I like to see the senior solution architect take on this responsibility himself, because it's ultimately too important to the success of the project not to get this right.

► For more information on the make-up of the SAP project steering committee, see "The Structure and Role of the Steering Committee," p. 47 in Chapter 2.

The solution architect needs to ask questions related to business processes—how critical each one is, typical and worse-case timelines, how disasters are currently addressed regarding the execution of these business processes, and so on. In the end, the SA must architect a system that delivers the highest level of availability needed by a particular subset of the end-user community (the rest of the end-user community gets to benefit from the HA and DR needs of the most critical processes). All of this information will eventually find its way into the project's knowledge repository, discussed earlier in the book and addressed in more detail toward the end of this chapter.

Determining Your Required Level of Disaster Tolerance

After the uptime requirements of each business group are understood, reviewed, rationalized, and finally documented, the SA can begin the task of reviewing these requirements with an eye toward cost. That is, a good SA will not just design a \$50 million solution capable of delivering five nines of availability simply because the

business told him that's what they need! Instead, he'll work with the business to help them understand the costs inherent in such a solution, as he also works toward better understanding high-availability and disaster recovery options that can be implemented up and down the solution stack.

DR and Return on Investment Calculations

Disaster recovery-related TCO and ROI exercises represent valuable methods of comparing various solution approaches. Such an approach boils down to math—each solution must be evaluated in terms of the following costs:

- Infrastructure costs associated with the floor space that will be required (and thus lost from other data center applications), as well as power, cabling, potentially increased network capacity, and more.
 - Acquisition/procurement costs of the actual hardware or software solution.
 - Other costs related to procuring the solution that might not be as obvious, like factoring in a specific version of a high-availability supporting operating system, database, or specific hardware component upgrades/delta necessary to realize the value the solution presents.
 - Costs related to implementing each solution. This often amounts to a combination of SAP Technical Support Organization, or TSO training, plus HA/DR solution-provider consulting hours, and any costs specific to knowledge transfer enabling the customer to implement the solution themselves in the future.
 - Support and other steady-state management costs inherent in each HA/DR approach. Some solutions require special enterprise management bolt-ons (that of course add to the cost of the enterprise management utility); other solutions benefit from specific versions of enterprise management and other supporting software.
 - A value should be assigned to each HA/DR approach, such that a “cost” can be assigned related to the general complexity of each solution. When all other things are close to equal, the less complex solution is always more appealing.
 - Finally, weighing each solution in terms of its unique cost/benefit ratio is advantageous. One solution, for example, might provide 99.95% availability, whereas another might provide 99.9%—both meet a “three nines of availability” requirement, but if all other costs are close to equal, it will probably make sense to go with the solution supporting the higher level of availability.
- To learn more about the costs of downtime related to disasters, as well as DR site TCO considerations, see “Disaster Recovery Requirements That Drive TCO,” p. 130 in Chapter 5.

In some cases, “doing the math” clearly supports one alternative over another. More often, though, quite a bit of work needs to go into understanding different vendors’ approaches to solving availability challenges.

Calculating the Cost of Downtime

Another key area that the SA must understand is related to the average cost of downtime, both planned and unplanned. As the graphs in Figure 6.3 show, planned downtime is inherently less expensive to an organization than unplanned downtime. Nonetheless, the numbers start out pretty big, and grow quickly as an organization incurs more and more downtime. Consider the cost of planned downtime at the end of the millennium, for example. At more than \$35K per average outage per week, and something like \$10K/hour, even the “typical” small ERP organization lost a small mint over the course of the year. Manufacturing organizations lost even more—an hour of downtime alone racked up \$28K. And yet even that was a paltry sum compared to banking and similar financial institutions, who might lose \$2.5 million per hour of downtime, or brokerages houses, who might lose \$6.5 million per hour of downtime. Today, although I have no new figures to share, it seems reasonable to assume that these numbers are probably much higher—an incremental dollar in revenue is certainly harder to capture now than it was a few years ago, and an organization’s customers are even more likely to shop around, thereby weighting the cost of lost customers and lower customer satisfaction more heavily than I’ve ever before experienced.

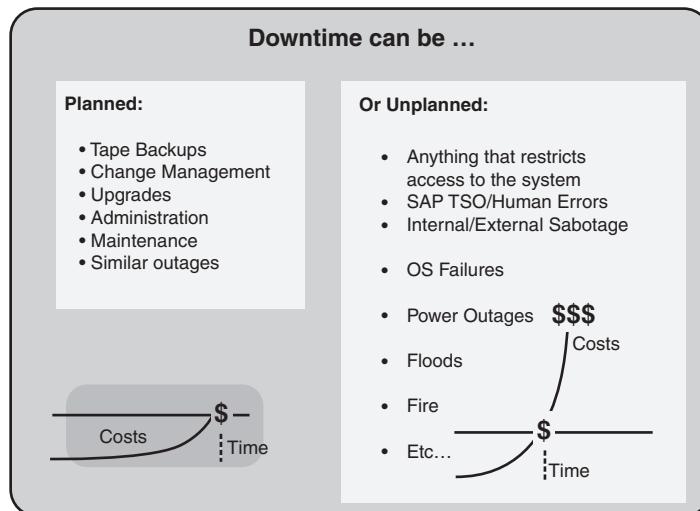


FIGURE 6.3 Regardless of whether downtime is planned or unplanned, it is expensive. But the costs of unplanned downtime can grow rapidly, and therefore these costs need to be well understood.

With dollars like these at stake, the cost of both planned and unplanned downtime needs to be well understood by all of the members of the SAP project's steering Committee. In particular, the SA needs to understand how availability impacts costs to the company, so that a solution design begins to emerge that addresses these needs. These costs can manifest themselves as lost revenue streams, spoiled/unfit inventory, lost end-user productivity, decreased customer satisfaction, decreased vendor or distributor faith, less-optimized supply chains, increased compensation, damaged reputation, and so on—in each case, for each specific mySAP solution, the costs need to be characterized, understood, and documented.

In the next section, I take a look at how the solution architect can begin educating himself about potential HA/DR solutions, where the goal is simple—to begin eliminating as many non-applicable or substandard approaches as makes sense, so that you have an idea as to what sizing information to share with your hardware and software partners going forward.

- ▶ Solution design and other “sizing” considerations are covered in depth in the next chapter. See “Overview—The Sizing and Blueprinting Process,” p. 221 in Chapter 7.

Educating the Solution Architect on HA/DR Options

When the solution architect has his arms around the “nines” and therefore understands both basic budget constraints as well as uptime needs, he can then begin researching in earnest the various HA and DR approaches and solutions at his disposal. At this point, I assume that the specific hardware and software vendors upon which the SAP Solution Stack will be formed have not been determined. However, the list has probably been narrowed down to some extent as we worked through technical staffing considerations in Chapter 4, TCO considerations in Chapter 5, and now have begun to understand basic availability drivers.

Based on my own experience designing and supporting SAP solutions that have sat on top of numerous hardware platforms, across five very different operating systems, and leveraging five very different relational database systems, I drafted the following “Top Ten” list for educating oneself on various SAP high-availability and disaster recovery offerings:

- Vendor-supplied general high-availability papers/best-practices documents for SAP.
- Vendor-specific documentation for HA solution offerings for SAP, like the installation PDF for HP's MC/ServiceGuard for SAP Extension.

- Reading and research via SAP technical journals, the World Wide Web, and other such resources—SAP AG-provided HA and DR documents are a good example.
- Vendor-supplied SAP questionnaires, which generally detail a vendor's offerings in support of highly available SAP solutions.
- SAP and partner-sponsored SAP-specific technical conferences, user group meetings, seminars, and so on, where material can be collected and relationships with key SAP and partner consulting/technology organizations can be forged.
- Leveraging a “circle of support” that can be constructed by building relationships with other SAP customers who have similar HA requirements, or belong to a specific user group or industry vertical, and so on.
- Formal training or less formal workshops where official training curriculum is available, and a portion of the training focuses on hands-on implementation.
- Collecting and actually walking through vendor-supplied recipes and checklists for highly available SAP installations (such as cluster-specific documents, or installing 9iRAC for SAP, and so on).
- Leveraging the hands-on experience of SAP, and SAP Global Partners, when it comes to designing and actually implementing HA/DR solutions.
- Working in a technical sandbox environment with demo or other such working copies of HA software, or the applicable hardware components.

By taking advantage of resources such as these, and gaining whatever limited hands-on or training exposure he can get, the solution architect will be well-positioned to further rule out solutions that fail to meet the organization's needs.

Remedying Single Points of Failure in the Stack

We can now turn our attention to the actual problem areas inherent in our particular SAP Solution Stack, and ultimately to the mySAP.com solution being deployed. Single points of failure (SPOFs) potentially exist throughout the SAP Solution Stack and the SAP system landscape in general. We will walk through each layer in the stack next, grouped into six key areas as shown in Figure 6.4. We also look at both general and layer-specific high-availability and disaster recovery offerings as well, noting advantages and disadvantages of each as we move through the simplified stack.

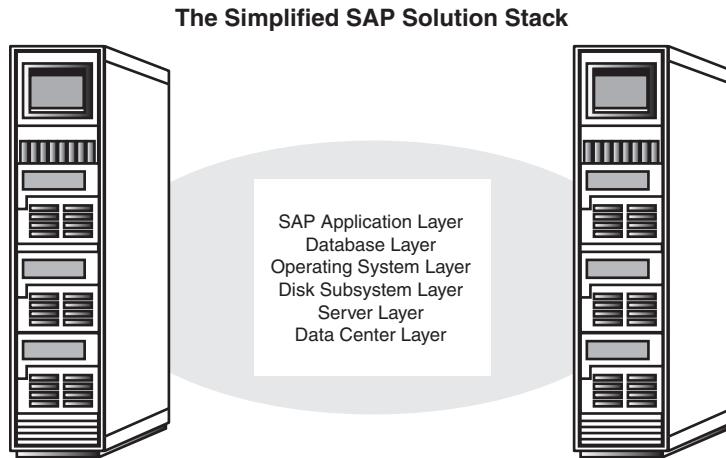


FIGURE 6.4 To simplify things, the complexity of the SAP Solution Stack is reduced to six key areas in this chapter.

SAP Data Center Infrastructure SPOFs

Single points of failure abound in the data center layer more than most. Fortunately, this is one of the easiest layers to address from a high-availability perspective. Consider the following:

- Data center power must be redundant, from the actual power sources or power grids (which can be safeguarded through the use of a generator), to dual-feed breaker panels, to dual UPSes, to dual power distribution units, to dual power cables feeding dual power supplies in every critical hardware component.
- Data center cooling represents an everyday single point of failure. I've seen more than a few data centers that cannot withstand the loss of one of their air handlers/air conditioners without the heat rising to the point of forcing equipment power-down.
- Network infrastructure, like power, also benefits from redundancy. This is important from the client network all the way back to the data center. Client network routers, dial-up access devices, VPN connections, and other access points must all be redundant. Similarly, any switches, bridges, hubs, or other network devices also need to be protected through redundancy. Any application that leverages a particular subnet in an attempt to integrate with your mySAP.com solution also becomes a SPOF. Finally, the network cables and individual network cards that facilitate communication to a particular server must be redundant as well.

- Data center rack placement can inadvertently become a single point of failure, too—just ask my customer who placed both of their SAP cluster nodes in a single rack, only to have that rack tip over during a maintenance window and completely lose power. In another case, I heard a story of another shop that had positioned their racks too closely together front-to-rear, such that serviceability to Production SAP was impacted when another system (non-SAP in this case) was also undergoing maintenance.

Power Considerations

With regard to power, remember that every critical server, disk subsystem, network component, air handler, and so on should have access to redundant power. For maximum availability, therefore, every component in the chain of power should be redundant, as you see in Figure 6.5.

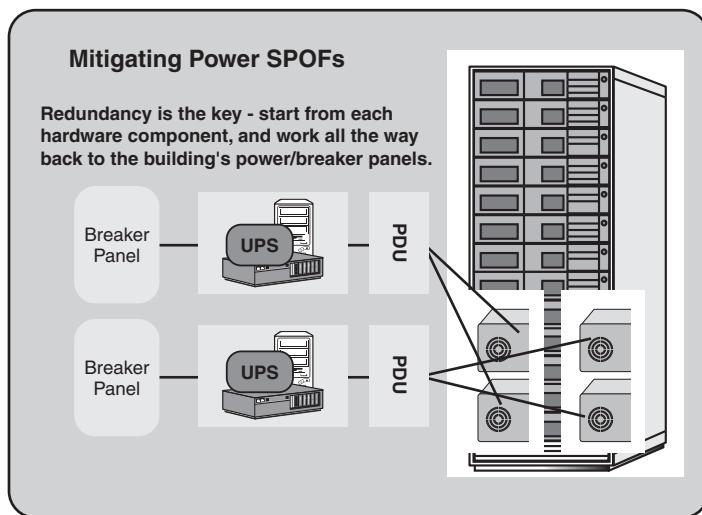


FIGURE 6.5 Redundant power starts at each hardware component that supports redundant power supplies, and works its way back to the ultimate power source.

Additionally, I like to see power cables color-coded. For example, at one of my customer sites, the primary power is supplied through black cables, and the redundant power is supplied through gray cables—this reduces human error later on, as it is very clear to everyone which components are protected, and which power source is being drawn upon during maintenance.

There are also some very important tools or utilities available that should play a part in monitoring power. Most UPS vendors offer tools that snap in to popular SNMP-based enterprise management tools, for example.

Network Infrastructure

When it comes to network infrastructure, redundancy is key—if an SAP end user cannot gain access to the system because a critical network link is down, the system is in effect “down,” after all. Other important areas that need to be considered include the following:

- Color-coding makes a lot of sense here, too. One client site of mine uses a public network segment for client-to-application server traffic, and another segment for back-end application server-to-DB server traffic. The first segment uses solid green cable for primary connections, and striped green/white cables for redundant links. Similarly, the back-end network consists of solid blue and striped blue/white cabling. This level of standardization makes troubleshooting simpler, and reduces human errors going forward.
- Certain software packages or HA solutions require the use of virtual IP addresses, or *relocatable* IP addresses. This may require special software or network drivers that support teaming or pairing network cards into a single virtual network card.
- Any hardware-based network load-balancing gear must be redundant, too.
- Some software-based load-balancing solutions do nothing for high availability. For example, Microsoft’s Network Load Balancing (NLB) does not detect application failures.
- Similarly, some failover cluster approaches and other high-availability solutions cannot detect network failures. Thus, these failures need to be detected in another manner, for example, through enterprise management software or hardware-specific monitoring utilities.
- Dual-port network cards (NICs with two ports on one physical card occupying a single PCI slot, for example) present an opportunity to create a SPOF where none might otherwise exist. Consider the case of one of my SAP customers who implemented a pair of dual-port NICs, but ran both ports from NIC “A” to their client network, and both ports from NIC “B” to their back-end network. In doing this, each NIC became a single point of failure. A better approach would have been to take one set of ports from *each* NIC, and run this to the public network, and then run the two remaining ports from each NIC to the back-end network.

- Disaster recovery sites probably need to be protected in a similar manner. Plus, links to the DR site from the public network also need to be protected. I recommend two links managed by two different service providers—in this way, even provider-specific issues do not become a single point of failure.

With redundancy in place as shown in Figure 6.6, even multiple failures throughout an SAP enterprise will not take the system out of service.

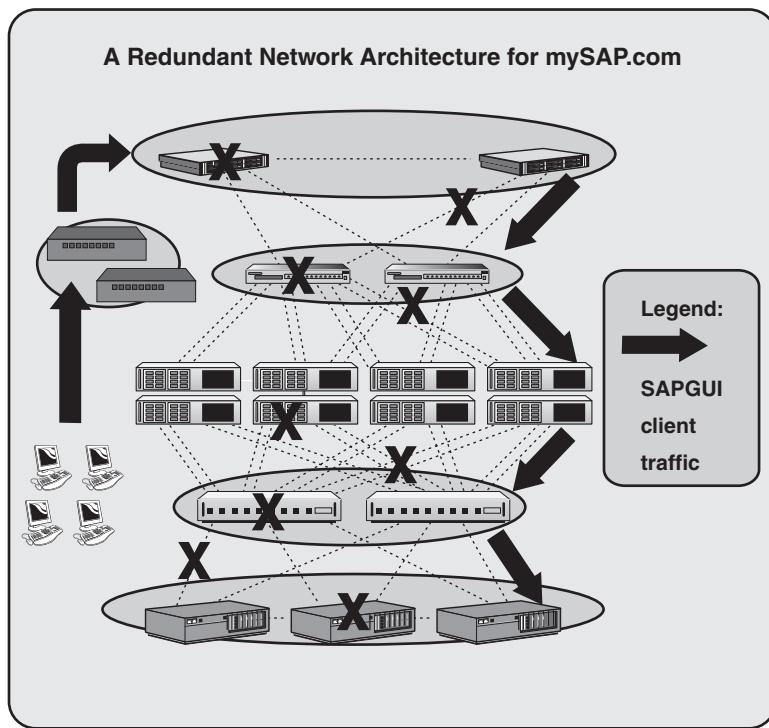


FIGURE 6.6 Network-layer redundancy allows for a variety of discrete and multiple failures, while still affording connectivity to a mySAP solution.

Finally, with regard to network components, it's important to proactively monitor these critical SAP solution points as much as any other component whose failure is capable of essentially shutting down your SAP system.

Rack Infrastructure in the Real World

Perhaps surprisingly, the rack infrastructure used to physically house hardware represents a big HA concern. Not only is it important to physically install these racks

consistent with the vendors' requirements, but it's also important to step back and take a holistic view of what is actually housed in the rack and how this gear is accessed. For example:

- Single points of failure are created when a specific type of computing resource is only installed in a single rack. Therefore, I always recommend that my customers shy away from creating "centralized" racks housing all network components, or all production SAP application servers, or both nodes of a production cluster, or all SAP Internet Transaction Servers, and so on. In this way, loss of power or network services to an individual rack does not automatically (in effect) bring down an entire production system.
- Housing heavy components high in a rack may cause the entire rack to come crashing down when the heavy component is pulled out for service. This, and the fact that it's just plain dangerous and contrary to all vendor warnings, should warrant repositioning heavy gear on the bottom of the rack.
- One of my customers mounted their TFT rack-mounted retractable monitors on the top of a row of 7-foot racks. In and of itself, this was not a terrible thing. However, the inconvenience of accessing the monitors eventually impacted production uptime at this small shop. How? My SAP Basis/Operations colleague did not spend the time needed to proactively monitor his SAP R/3 system from a hardware and OS perspective. Eventually, a memory leak (due to a poorly written management agent) caused one of the production servers to crash.
- Keyboard, video/monitor, and mouse (KVM) connections need to be considered. Like the problems described regarding "centralized" rack configurations, KVM can represent a SPOF if both nodes in a cluster, for example, share the same KVM components. I also like the idea of maintaining two different methods of accessing server resources, like virtual connection methods (PC Anywhere, PC-Duo, Windows Terminal Services, KVM over IP, or HP's Remote Insight Boards) in conjunction with physical KVM connections.
- Lack of sound cable management can also compromise high availability. That is, cables that are improperly mounted to servers such that pulling out a server wrenches out its power cables will only increase unplanned downtime. Similarly, stuffing all of your power, network, SAN, and other cables into an already crowded subfloor not only complicates cable management, but also makes troubleshooting more complex. In fact, in the case of one of my customers, too many piled-up cables actually blocked cool airflow to some of their server racks, resulting in automatic hardware-initiated power-down. Today, ceiling-mounted cable trays for network/SAN cables, combined with subfloor-based power cables, help ensure that they never succumb to this problem again.

- Racks stuffed with gear from varying vendors may improperly vent or exhaust hot air. This is especially true in cases where some of the gear draws air from the front and exhausts hot air out the back, while other gear draws cool air from the subfloor and exhausts it out the top, and other gear pulls from the tops and exhausts into the subfloor!

As one of my SAP clients recently told me, BTUs are the true enemy of the data center today—the high densities and smaller and smaller form factors that we all benefit from only exacerbate the cooling needs of already-crowded data centers. Thus, it is as important as ever to plan well, and plan ahead, not only for data center growth but also for data center cooling requirements.

The Ultimate SPOF—The SAP Data Center

What will happen to your business if your entire data center falls off the face of the earth? Answering this question consumes many SAP DR specialists' time, and keeps a lot of people awake at night. Certainly, it's expensive to duplicate an entire data center. In my experience, I see more customers duplicate core business processing systems instead. Other key concerns include the following points:

- DR sites are expensive to build.
- They are also expensive to maintain.
- Testing a DR site is expensive, too.
- Further, testing seldom reflects the reality of true disasters without a tremendous amount of time invested in developing and working through disaster scenarios.
- It can be difficult to sell a management team on the idea of spending a pile of budget money on a resource that will hopefully never be "used."

I therefore like to identify the hidden benefits of a DR site outside of the most obvious benefit—protection of your core business (which should be enough to satisfy anyone). These include leveraging the DR site for end-user reporting requirements, using the site for testing major infrastructure changes, and using it for pre-upgrade testing/staging. Staging your pre-production systems at your DR site with the strategy of using them in the event of a primary data center failure can also ease financial pressures associated with building a DR site. Finally, using your DR site as the definitive testing ground in your change control or change management processes, prior to deploying changes to production, represents a huge benefit as well.

Barring selling management on any of the benefits just discussed, another perhaps even better approach can be taken. Rather than spending a fortune on idle

computing resources, a company might consider outsourcing their DR needs on a “computing on demand” basis instead, where computing resources are paid for as needed. A monthly “access” or “availability” fee is typical, of course. In the long run, though, the savings can be huge. Large SAP outsourcers are doing more and more of this type of DR service, including HP, Cap Gemini Ernst & Young (CGEY), and others.

SAP Server SPOFs

Server-based single points of failure have been recognized and addressed for many years now. Today, what used to be considered “features” or “differentiators” between server platforms are no more than commodity traits. As you see in Figure 6.7, things like ECC (error code correcting) RAM, redundant/hot-pluggable power supplies, hot-pluggable hard disks, redundant/hot-pluggable fans, hot-pluggable PCI cards, and so on can be found in everything from mainframe class platforms to practically entry-level servers.

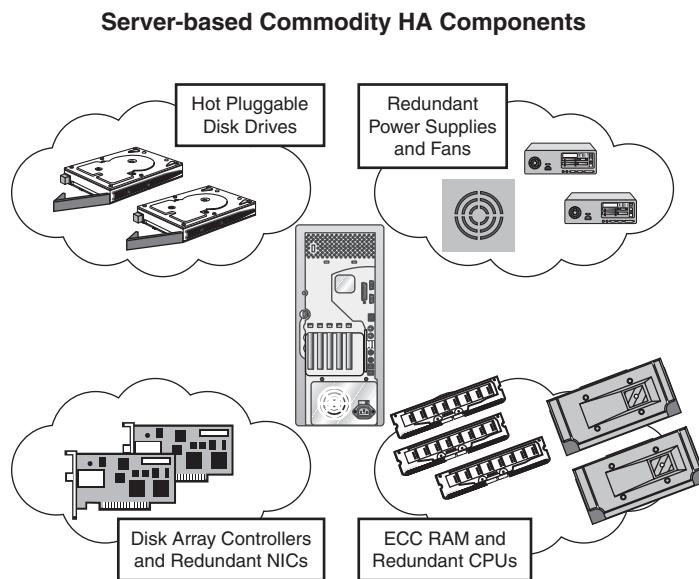


FIGURE 6.7 High-availability features like those identified here are practically a “given” in server platforms today.

Server manufacturers continue to strive to address HA in their new platforms. Certainly, some of the newest features, like RAID-protected hot-pluggable RAM and hot-pluggable processors, will also become commodities in the near future.

Clustering

After a server manufacturer has addressed high availability "inside the box," it becomes a bit more challenging to provide even higher levels of uptime. That is, single points of failure may still exist even in "highly available" servers, but it becomes more and more expensive to address these last SPOFs through hardware redundancy. For example, a server's primary system board or bootable disk controller could be made redundant—indeed, systems exist today that can be purchased with this level of redundancy. But the typical benefit realized through this level of redundancy is simply the minimization of downtime, not transparent failover. Consider the following—if a server's primary disk controller fails, the server's connection to its operating system is severed. Eventually, the server will lock up or freeze. Most server manufacturers address this problem by monitoring the activity of a server, and simply initiating an automatic reboot. Thus, the benefit of a redundant controller is only realized when the server is rebooted (in this case, rebooted successfully to the second controller)—no time is wasted rifling through onsite parts bins locating a new one and replacing the failed one.

Rather than investing engineering dollars in redundant system boards and other components that realize little ROI, many server manufacturers have instead adopted a preference for promoting server clusters. In this way, the entire server is theoretically protected from internal failures, and the long-term cost to the consumer (in terms of procurement, management, and service/support costs) is potentially less. Before we take a closer look at various clustering approaches and vendor-specific offerings, though, let's step back a bit and start from the beginning.

What exactly *is* a cluster? In so many words, a *cluster* is simply an interconnection of servers and storage supported by a software solution that in effect creates a virtual server-based solution. In doing so, services associated with enterprise applications like mySAP solutions, or the components that make up an SAP solution (like the central instance and database system) can seamlessly move between clustered servers, or *nodes*. And end users who connect to these systems can relatively easily be shifted back and forth between available nodes, as illustrated in Figure 6.8.

Thus, for our purposes here, clusters increase high availability by removing the entire physical server as a single point of failure. And taken to the extreme, clustering also addresses more than high availability. Clustering can be leveraged to provide highly available geographically dispersed solutions, where not only servers are removed as SPOFs but so is storage (in the case of cluster implementations that support writing to two different disk subsystems, which is addressed later). These types of cluster solutions both take care of disaster recovery concerns and meet basic availability requirements.

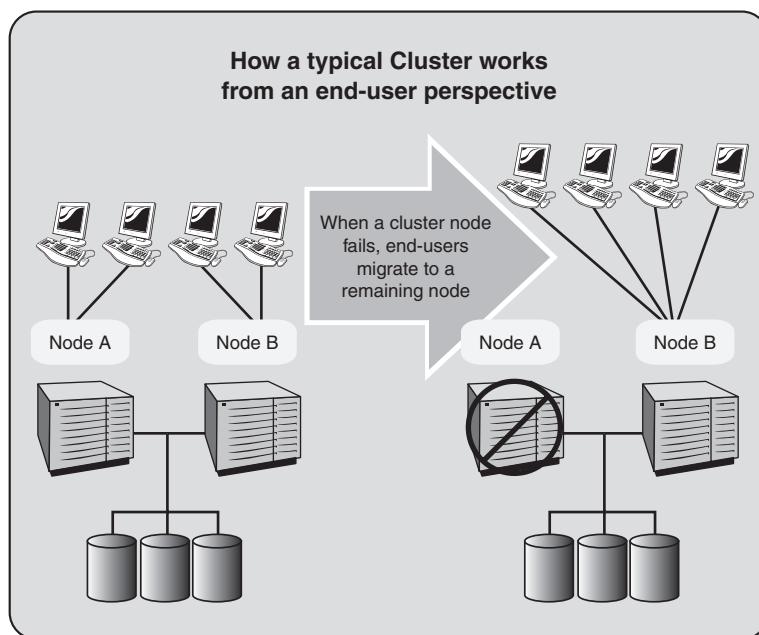


FIGURE 6.8 To the end users connected to this cluster, it appears as though they have connected to a single system. They are usually unaware as to which physical server in a cluster they are connected.

Clustering techniques are familiar and commonplace today. Every server platform supported by SAP can be clustered (simply because every operating system supported by SAP also supports clustering). And every hardware and enterprise storage vendor supports clustering, too. Refer to the next few sections in this chapter for details and discussions of specific clustering solutions for SAP.

Disk Subsystem Single Points of Failure

For years, high availability and disaster recovery have been focused on the data that resides on disk subsystems. Disk subsystems, due to the physically moving disk drives themselves, represent by far the most common failure point in computing systems. In fact, it is because of this shortcoming that RAID (redundant array of inexpensive/independent disks) came to be.

Disk Drives and RAID

Different levels of RAID exist, each with its own advantages and disadvantages. Some of the most common implementations include:

- RAID 1, or mirroring. Here, a physical drive “mirrors” a second drive. Thus, for every two drives paid for and installed, only one is effectively “used” for data—for this reason, it is the most expensive of all RAID options. However, the penalty realized in terms of usable space proves to be a great benefit when it comes to the number of disk drives that may be lost while still providing access to the data. In all, half of the disk drives in a RAID 1 array may fail. Thus, RAID 1 provides the highest level of availability of all RAID options.
- RAID 5, or striping data and parity. Traditionally with RAID 5, the equivalent of one drive (in a set of drives) is lost to maintain parity data, such that any single drive in the array may fail and yet the data remains intact and accessible. Thus, a set of eight 36GB drives in a RAID 5 configuration does not equal 8 times 36, or 288GB. Instead, it is 252GB, as expressed in the following formula:

$$(\# \text{ drives} * \text{size of each drive}) - (1 * \text{size of each drive}) = \text{total usable space for data}$$

In other words, a RAID 5 solution provides the equivalent data space of $(n-1)$ drives. If a second drive in the array fails, though, you lose data. Exceptions exist to this in some of the newer virtual SAN array technology, though, where the equivalent of multiple drives are “lost” to maintain parity information, thereby allowing multiple drive failures within a single array.

RAID 5 is typically utilized for data volumes when the cost of additional disk drives, disk controllers, and so on is as much a consideration as raw I/O performance. As such, RAID 5 represents a good balance between cost and overall performance.

- RAID 1+0, or 0+1, or 10, or striping/mirroring. Different vendors implement this differently (hence the different labels given to this RAID option), but RAID 1+0 is essentially mirroring a large group of drives and then striping data across the drives *without parity*. Thus, the availability features of a RAID 1 solution are realized, with the added benefit of outstanding performance as multiple disk heads are accessed concurrently. Unlike RAID 5, parity is not required because the data is protected by virtue of the one-for-one disk mirroring.
- Other RAID levels exist as well, including RAID 3/5, RAID 4, and more. Refer to your hardware vendor’s RAID controller documentation for descriptions of these various implementations. Generally speaking, though, RAID 1, 5, and 1+0 tend to be used almost exclusively today.

NOTE

For any enterprise environment, refrain from implementing OS-based RAID for anything but your operating system drives. Sure, it works and it's supported, but it robs the system of CPU resources at the expense of overall mySAP application or database performance. That is, operating system-based RAID adds a layer of overhead that directly impacts the performance of everything running on top of the OS. This hit on performance may be nearly negligible in some cases, but all too noticeable in others. Protecting your data and logs in these environments is therefore best left to hardware-based array controllers dedicated to the task.

Interestingly, RAID 1 evolved in a few different directions. In the case of RAID 1+0, it evolved to provide greater performance than can be realized by a RAID 1 solution. At the same time, disk subsystem vendors identified yet another permutation that would provide even greater availability and support new functionality as well—*cloning*, or *triple-mirroring*.

Cloning and Triple-Mirroring

The simplest way to explain triple-mirroring is to think of a RAID 1 mirror, which consists of two drives, and to then add a third drive to maintain yet another copy of the data housed on the two other drives. Thus, a single data drive requires two additional drives to support triple-mirroring, as you see in Figure 6.9. HP refers to this as cloning; other vendors label this with their respective trademarked names as well.

Cloning is a wonderful way to provide near-disaster recoverability while also presenting some other compelling capabilities. Disks may be “frozen” in time, for instance, accomplished by directing a disk controller to quit synchronizing multiple disks. In this way, a disk in a clone set can be used to keep a point-in-time backup of a database—instead of spending hours restoring from tape or performing disk copies, a physical as-is “snapshot” of the system can be created. Later, if this snapshot is needed, it may simply be presented by the disk controller back to the database server, and logs may even be applied to it (if they have been maintained somewhere where they are accessible) so that the database can roll forward to a more current point in time.

Cloning provides other benefits, too. For example, many of my customers use this third mirror to run their nightly and weekly tape backups. Doing tape backups this way can eliminate the performance hit from the actual production system (as opposed to performing online backups), as well as shrink the downtime required to perform offline backups. This last benefit is most often leveraged—instead of having to keep a database system down for hours so that all data files can be dumped to tape, cloning reduces downtime to the few minutes it takes to quiesce a database and run a script that “breaks off” the third mirror of every LUN or volume that houses a database data file.

Note that with Cloning, only 33% of all drives contain unique data!

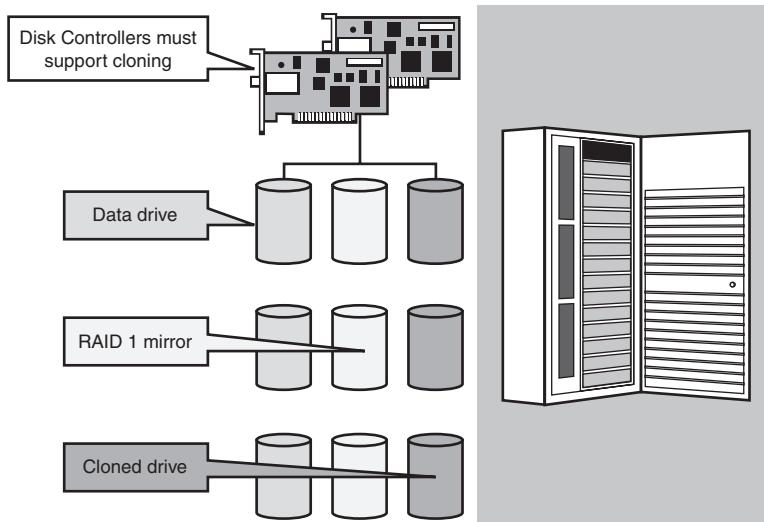


FIGURE 6.9 Although cloning represents an excellent HA approach, it's "expensive" in terms of the percentage of usable disk space yielded.

Finally, triple-mirroring also allows for the clone set to be presented to yet another SAP server, and then accessed by end users in read-only mode. This gives end users a system upon which to execute long-running SAP queries or reports, freeing the real production system from these intense resource-consuming tasks.

All of these benefits can be realized simply by adding a third mirror to mirror sets. Of course, the storage system must support this special implementation of RAID, and the data being cloned cannot typically exist in any format other than a RAID 1 or RAID 1+0 array.

Other permutations of triple mirroring exist, too. Another one of my largest SAP/Oracle customers actually maintains three mirrors, for example. We refer to this as quadruple-cloning, for lack of a better descriptor. At midnight every evening, my customer breaks off one clone set, and uses it to support both DR and to act as a source system for refreshing other SAP systems within their SAP R/3 system landscape. Then, at noon the next day, they resynchronize this clone set with production, and break off the *fourth* mirror to act as a point-in-time snapshot (primarily for DR purposes). By rotating every 12 hours between breaking off and resyncing new clone sets, this customer is well positioned to address a host of HA and DR issues that might crop up and otherwise consume many hours of system restore time.

Beyond the physical disk drives that we have covered here, other common points of failure exist throughout disk subsystems. These are addressed next.

Disk Controllers

Not only are the RAID options important for disk subsystem availability—the disk controller itself is equally, if not more, critical. It is the disk controller that supports the ability to create highly available RAID arrays or clone sets in the first place. And the disk controller may often be paired, too, such that a disk controller failure can be withstood.

Many disk controllers also support *data caching*, where frequently or last-accessed data is retained in onboard controller-based RAM, such that subsequent accesses to the same data are performed in microseconds rather than milliseconds. The best controllers provide battery-backed cache, and configurable read/write caches, so as to assist you in both safeguarding and tuning your SAP server platforms. Be wary of using array controllers with *write* cache that is not battery-backed, however. Such controllers run the risk of losing or corrupting your data in the event of a power failure. That is, any writes still sitting in the write cache that have actually *not* been posted to the physical disk drives represent data that will be lost when the server/disk controller loses power, if not backed up by a battery on the disk controller. At one time, this was a frequent cause of corrupt and inconsistent databases. Today, more and more array controllers—even fairly inexpensive versions—support battery-backed write cache.

Unlike write caches, a controller's read cache does not need to be protected from power outages. Why? Because if the server or controller loses the data sitting in a read cache, the controller will simply reread the data from physical disk again if it's ever needed. Read caches only read data that already exists, so no changes are possible.

Disk Subsystem Infrastructure

Drive shelves, individual drive cages, power bars supplying power to shelves/cages, and all of the gear used to connect the disk subsystem to the server also represent potential single points of failure. All of these are addressed by redundancy to one extent or the other. Protecting against the failure of an entire drive shelf (normally a horizontally positioned piece of gear housing 7 to 14 disk drives) is also accomplished through configuration best practices. For example, drives that are mirrored to one another (mirror pairs) should never reside in the same drive shelf—failure of the shelf would take down the entire data volume, mirror and all. Similarly, a RAID 5 array capable of surviving a single drive failure must not consist of more than one physical drive *per shelf*, as you see in Figure 6.10. In the real world, though, this best practice is often ignored—customers implementing RAID 5 solutions are usually

doing so because they are more interested in reducing cost than completely maximizing availability.

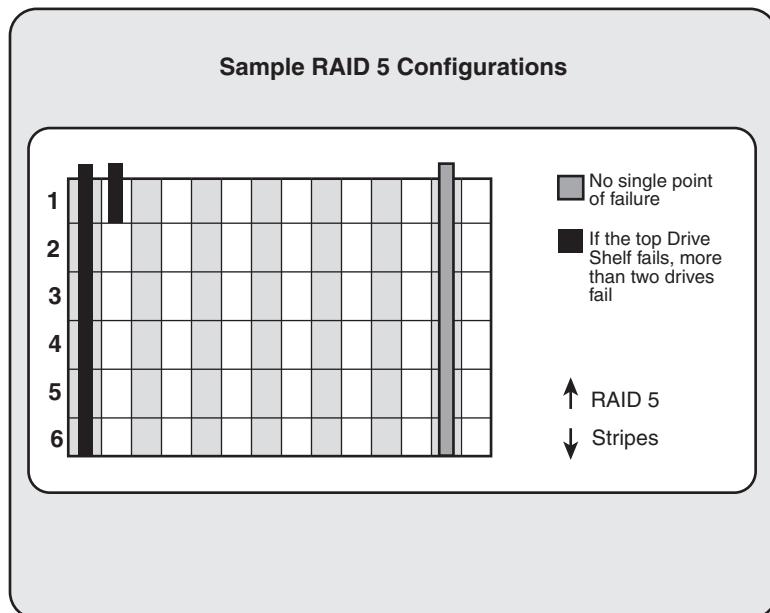


FIGURE 6.10 Traditional RAID 5 configurations can only afford the loss of a single drive per array, thus limiting high-availability solutions in terms of data capacity. Fortunately, newer virtual-SAN-based products allow for multiple drive failures.

As hardware manufacturers addressed these single points of failure over the years, what started out as high-availability offerings eventually supported true disaster recoverability, as you will see next.

PMD—Physically Moving Drives

Perhaps the most basic of disaster recovery approaches I've ever seen involved the following process:

- The customer configured all of their production database volumes for RAID 1 sets, where one drive mirrored another drive. In total, their production database data volumes resided across 16 drives (8 drives mirrored to 8 drives).
- Every week, they ran through a five-minute process that I helped draft for them long ago, culminating in the physical removal of the eight drives that act as mirrors to the eight primary drives.

- An operator verified that each drive was labeled as to the position that the drive held in one of the two disk subsystem drive cages (position 0 through 7, cage 0 or 1).
- Meanwhile, that same day a three-week-old set of drives was brought back in from the offsite storage facility and inserted into the now-empty drive cages. The drives were re-synced to the existing drives within a matter of hours, and afterward the system was again protected by RAID 1.
- The drives removed earlier were then packed and shipped offsite.
- In the event that the point-in-time snapshot was ever required (for example, upon tape drive or tape media failure when attempting a restore), they obtained the drives from offsite and reinserted them into the system. The QA system could be used in this role as well if the production server failed and corrupted the database in the process of failing.

This simple approach to DR is cheap, simple, and effective. No, it's not fully automated, and yes, it can take some time to get the disk drives back from the offsite facility. But it works for this small SAP shop, and gives them a comfort level unmatched by basic offsite tape shipping (which they still do as well).

Campus and Metropolitan Clusters

Shortcomings to approaches like PMD spawned more robust and more automated methods of maintaining offsite disk-based copies of a database. Solutions like that shown in Figure 6.11 evolved, where physical or OS-based disk sectors were copied and synchronized between sites. In this way, regardless of the database or application layers of a solution, a highly disaster-resilient solution could be employed.

A good example of this kind of solution still in wide use today is HP's Campus and Metropolitan Clusters, where a primary data center and its disk subsystem are essentially "mirrored" to a secondary data center. Such an arrangement may entail mirroring only disk subsystems, or mirroring both servers and disk subsystems. The former does little for true disaster recoverability, so the latter is a more common DR approach.

In the case of a Campus Cluster, the secondary data center is actually located at the same "site." This might be within the same building, or perhaps a few buildings away in the case of a true campus environment. Such a cluster is built by leveraging the ability of software to write to two distinct disk subsystems, much as hardware-based RAID 1 writes to two different physical disks. The only difference here is that the physical disks are located in different disk cabinets, housed in different data centers, and a software layer like MirrorDisk/UX actually handles the data replication between the two points. To make the physical connection possible, fibre channel

links are used for the data connections, and the cluster connection (or “heartbeat”) operates across a dedicated network link backed up by the IP-based public local area network used by all servers and clients. In case one of the heartbeats is lost between the sites, a cluster *lock disk* at each site acts as a “tie breaker,” dictating which site will own the cluster resources. All nodes must therefore be connected to both cluster lock disks. Not doing so (or losing both heartbeat connections) risks a condition referred to as *split-brain syndrome*, where each cluster thinks the other is down, and therefore tries to start up the services and packages related to both cluster sites. The results can be quite ugly, as you can imagine. To avoid split-brain syndrome, ensure that your heartbeat connections are routed over completely separate network infrastructures, and avoid creating cluster-disk-related single points of failure. For example, it’s important to ensure that each lock disk is maintained in its own disk subsystem; otherwise, a single point of failure results, and the cluster’s tie-breaking ability falls apart.

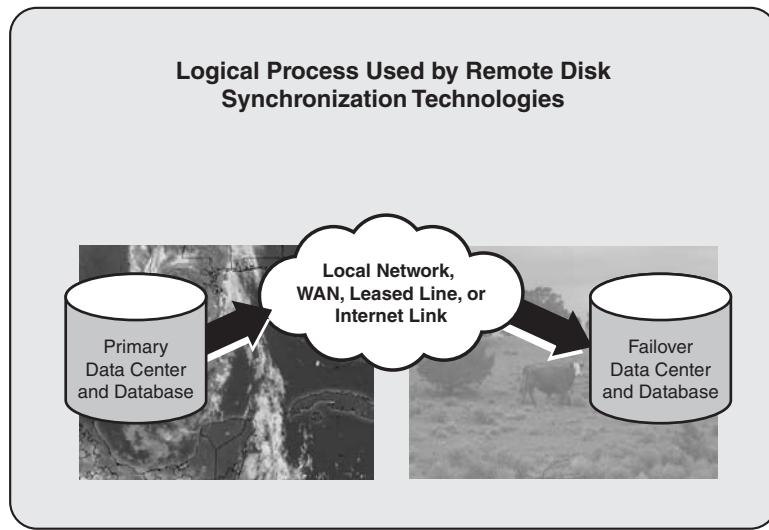


FIGURE 6.11 Remote disk synchronization technologies like this are in wide use today, as they are typically database- and application-independent.

A Metropolitan Cluster, or MetroCluster, takes clustering to the next level by allowing for the secondary data center to reside farther away, up to 100 kilometers from the primary. The replication technology must change, however, due to the potential distances involved. This is because the standard maximum fibre channel distances are exceeded, and therefore native SCSI/RAID operations or operations managed via MirrorDisk/UX are not possible. Replication to the secondary data center requires the features of highly capable disk subsystems like EMC storage systems supporting the Symmetrix Remote Data Facility (SRDF), or HP’s SureStore XP Array with Continuous

Access. To get around the distance issue, ESCON links are used, and the remote database copy is maintained in read-only mode during normal operation while each write operation is performed synchronously on disks in both locations (it is this write operation that effectively limits the distance between the data centers).

Should a failure occur at the primary data center, the secondary data center's disk subsystem is set to read/write mode and thereby put into "production." No cluster or quorum disks are used to manage and track the resources at each data center site. Instead, cluster arbitration servers located at a third data center site, connected to the public network, help determine which site should be running and presenting itself as the production system. In doing so, the arbitration servers play a role to avoid split-brain syndrome, much like the role played by the Campus Cluster's dual cluster locks. Otherwise, it would be unclear to each set of nodes (site) as to which site should host the productive SAP instance. That is, without a third-party arbitrator, each site would otherwise determine that *it* should host production, as both sites would believe the other was down due to lack of network connectivity. Because of the multiple network links between all three sites, this arbitration server approach to avoiding split-brain syndrome is superior to the approach used by Campus Clusters.

HP Continental Cluster

Continental clusters build upon the greater distances allowed with a MetroCluster, to the point where the primary and secondary data centers can reside on different continents! Each data center maintains its own quorum, and any IP-based WAN connection suffices for data replication. Four key potential issues must be addressed prior to deploying a Continental Cluster, however:

- It's unlikely that the intercluster connection will be a dedicated line, due to the tremendous costs potentially involved. Thus, the time lag for data replication needs to be determined and addressed (or budget money earmarked for this regular expense).
- Given the data replication issues, a WAN-capable Transaction Processing Monitor (TPM), a database replication tool, or some other similar utility is required to ensure that transactions are applied to the database in the correct order.
- Operational issues are complicated in the event of a failover, given that failback can be difficult—Continental Clusters initially supported failover in one direction only, in fact (now, bidirectional failover is straightforward). Regardless, the SAP Technical Support Organization needs to be clear on how to eventually restore operations back to the primary data center.
- Failover is not automatic; rather, the cluster itself detects a problem and then notifies the system administrator, who must verify that the cluster has indeed failed before issuing a command to recover the cluster at the other site.

In the most common implementation, asynchronous replication is used for Continental Clusters given the large distances involved in data replication. For performance reasons, I recommend that only *changed data* (not all data) be sent across the line, to be applied to the secondary database. Therefore, when first implemented, the secondary site must be set up manually in a data-consistent state with the primary site, such that changes are tracked and committed as they occur, one for one.

HP/Compaq Data Replication Manager (DRM)

Like MetroCluster and Continental Clusters, HP's DRM solution is also hardware-based—it requires HP's HSG80-based StorageWorks disk subsystems installed with version x.P firmware actually coded to support DRM. Sometimes referred to simply as a "stretch-cluster," DRM implementations for SAP require that transactions be committed synchronously between the primary and secondary data center sites. Therefore, distance limitations of less than 100 kilometers are required (though theoretically, any distance is acceptable as long as the resulting lag in end-user response time is also acceptable). By forcing synchronous communications, the data that is written to the cache of the primary database system is written to the secondary (target) system simultaneously. To ensure SAP data integrity, the primary site waits for an "I/O completion" message from the target site before the I/O is considered successful.

CAUTION

Asynchronous data transfer is an option for DRM. However, given the critical role that a DR site plays in ensuring that a given mySAP system remains available even after a disaster, the risk related to never receiving the last few data packets sent by the primary system prior to failure outweighs the performance benefits inherent to asynchronous transfers—even a single missing data packet would put the DR database into an inconsistent or unknown state, making it useless for DR.

Unlike similar solutions in the marketplace today, the HP Data Replication Manager solution supports robust failover in both directions, and therefore allows each data center to effectively "back up" the other. Plus, given its huge support base of different operating systems—Tru64 UNIX, HP-UX, OpenVMS, Microsoft Windows 2000/NT, Sun Solaris, and IBM AIX—DRM is well-positioned for organizations running diverse server environments, as long as all of them leverage HSG80-based StorageWorks disk subsystems (soon to include HSV-based solutions, too).

Single Points of Failure Inherent in the OS

When it comes to the operating system running on a server, HA often equates to

- An ability to software-mirror critical volumes or partitions, for example, through HP's MirrorDisk/UX or Windows Disk Administrator (though the latter is not normally seen much nor recommended for production systems)
- A highly available file system that supports redundant file allocation tables or similar structures
- An ability to apply changes or patches "on the fly" or in another manner that minimizes downtime
- Protecting the NFS server, in the case of UNIX-based solutions, where the NFS server represents a single point of failure. Therefore, the UNIX OS must support a high-availability option like clustering the NFS server.

As to the last point, clustering support in and of itself is a key feature of an enterprise-class operating system (which is described in detail later in this chapter).

Another interesting file-system consideration is whether an OS supports *journaling*. A journaling file system is a fault-resilient file system that updates a serial log on disk immediately after a change to the disk, thereby better ensuring metadata and ultimately data integrity. It also allows unsaved data to be recovered, making it an essential feature, in my opinion, of OSes used for mySAP.com applications. Many OSes support journaling. Windows NT/2000 supports it to a great extent, which is why it's unlikely that you will lose an entire disk volume due to corruption, for example, after the system crashes unexpectedly (like after a power outage). With any operating system, though, all of the data yet to be written to disk prior to the crash will still be lost unless it's protected by an array controller with battery-backed write cache.

On the other hand, the default Linux system (ext2fs) does not journal at all. That means a system crash can more easily corrupt an entire disk volume. I expect that this will be resolved soon in one manner or another, however, given the open-source availability of Silicon Graphic's XFS journaling file system.

Beyond the HA features built into a file system itself, restoring a file system quickly and without error is left to backup/restore solutions.

Tape Shipping and Disk-to-Disk Copies

Tape shipping—using backup tapes to rebuild a database system or other file system—is the oldest approach to high availability known. Tape shipping is limited, though, in that it's normally the slowest method of rebuilding, and it only helps to create a system that is very much out of date the minute the system is restored and "available" again. For example, a system that was backed up last night at midnight

and fails today at 4:00 p.m. will, even after the tape backup is completely restored, represent the system as it looked at midnight. Any changes made after midnight will not be captured, and will therefore need to somehow be applied to the system after the fact.

To address the amount of time a “backup” requires to be restored, many companies employ the use of disk-to-disk copies. Here, the image of a database or other file-system-level disk partition is simply copied to another partition for safekeeping. Of course, in Windows-based systems, necessary application-specific registry entries are not captured. But in the event that the data is needed again, the entire data file or contents of a disk partition may be quickly accessed or even copied to another location.

Although this speeds up the tape-restore issue, it does nothing for the point-in-time issue already described—our system still looks the same as it did at midnight, for example. For this reason, many database and hardware vendors came up with approaches to HA that minimize the delta between what a system looks like in real time, and how current a system can be created through restoring data or applying changes.

Microsoft Cluster Service—MSCS

Microsoft has supported clustering since late 1996 (it was introduced with NT 4.0 Enterprise Edition). In the SAP enterprise space, though, I didn’t see it really take off until 2001, after clustering with Windows 2000 Advanced Server and Data Center Editions had stabilized in the marketplace. In the last two years, my colleagues and I have implemented quite a few production MSCS clusters running under both SQL Server and Oracle databases. And because this type of clustering is dependent upon the SAP basis release version, rather than something SAP component-specific, there is virtually no difference in clustering the DB and CI of an R/3 system compared to SAP BW, APO, EBP, and other mySAP offerings.

MSCS works under the concepts of resource groups and virtual IP addresses. A resource, like the SAP CI, is defined in a group and given an IP address. Similarly, all database resources are defined in a second group and assigned an IP address. A third IP address is assigned to the cluster itself, and each host in a cluster receives its own IP address as well, for five total.

MSCS Issues in the Real World

In my experience, MSCS cluster issues generally involve the following points—as you can see, network issues are prevalent, and all of these impact high availability:

- If the network card binding order on one or both of the cluster nodes is incorrect, name resolution will probably fail.

- Problems with hosts files on one of the cluster nodes will cause major name resolution problems, too. Most often, servers are listed with incorrect IP addresses or aliases.
- Additional security parameters at the NT/W2K level need to be set, enabling an SAP database to be added to its cluster group.
- Oracle FailSafe (OFS) configuration issues are pretty common (Oracle databases only, of course).
- Setting the client network (under the Cluster Configuration tab in the Cluster Administrator utility) for Client Access only, and not Internal & Client Access causes the heartbeat connection to become a SPOF—if a heartbeat NIC fails in a node, internal cluster communication becomes impossible.
- Adding an additional third NIC in each node (to segregate public from back-end database traffic, whereas still providing a heartbeat NIC) tends to complicate network connectivity, especially where static routes are concerned.
- With regard to the heartbeat connection between the two cluster nodes, using a crossover cable instead of a hub or switch and standard network cables can cause issues. Refer to Microsoft's Knowledge Base (KB) article Q242430 for the process to follow when a crossover cable is used—if not followed, the heartbeat NIC may not be detected during the cluster service installation.
- Incorrectly setting up the shared disk storage, such that both nodes are not able to “see” the storage system, or are unable to access resources that are supposed to be shared, makes cluster installation impossible.
- Yes, one of my customers actually tried clustering with plain old Windows 2000 Server—you need W2K Advanced Server or W2K Data Center to support clustering.

For additional information on installing and troubleshooting MSCS implementations for SAP on HP's ProLiant platform, see “SAP MSCS clustering SQL2000 or Ora81x.pdf” on the Planning CD.

HP MC/ServiceGuard and Similar UNIX Cluster Approaches

HP has been clustering under the HP-UX operating system version 10.20 and higher since SAP R/3 basis release 3.0D, supporting both Oracle and Informix. Today, MC/ServiceGuard (MC/SQ) supports HP-UX 11.0 and 11i, and an SAP-specific bolt-on product called MC/SQ Extension for SAP automated much of the work involved with setting up, managing, and operating an MC/SQ cluster in an SAP environment.

MC/SG works by monitoring a heartbeat connection between cluster nodes, similar to most other clustering approaches. MC/SG is more flexible than many other clustering solutions, though, allowing a number of different configurations:

- One-package configurations, where both the DB and CI are defined in a single package (similar to a resource group found in a Microsoft cluster). This single package can move between one node and another, thereby creating something like an active/passive cluster, where one node does all the work while the other remains available “just in case.”
- Two-package configurations, where the DB exists in one package, and normally executes on one node, and the CI exists in a second package and executes on the second node. Such a configuration should be quite familiar to those folks who have implemented a Microsoft cluster for SAP, as the approaches are nearly identical.
- Multiple packages can exist, in effect allowing support for two or more clusters.
- In all cases, other SAP instances can run on one of the nodes while production runs on the other. It’s quite common to see a test or development system run together with production, for example. Normally, if the production node fails, the instances running on the second node are stopped so that all of the server’s resources can be granted to the production system as it begins to come up on the remaining node.

For different MC/SG configurations, refer to the “SPOFs and the Database Layer” section found later in this chapter.

HP/Compaq Tru64 Cluster File System

The approach that Digital Equipment Corporation took to clustering back in the early 1990s is still one of the most compelling high-availability offerings for SAP in the market today. And it’s one of the simplest. With most cluster solutions, each node in the cluster maintains its own boot partition and related driver stack. When a new node is introduced to the cluster, or a failed cluster node is replaced by a new one, this new server must be loaded with the same image as its peers—all of the hardware-specific drivers and various software components and so on must be duplicated, consuming valuable time and resources.

The *cluster file system* (CFS) pioneered by DEC, however, allows for all cluster nodes to share the same file system. Thus, where other cluster solutions take hours to add or replace a node, CFS requires a single command-line entry to do the same thing. This substantially reduces planned downtime requirements, and makes adding a last-minute cluster node a breeze.

Because of this differentiating feature, among others, Oracle Corporation selected the Tru64 operating system as its development platform of choice for 9i Real Application Clusters (the HA successor to its *Oracle Parallel Server* [OPS] product). RAC is discussed later in this chapter.

Linux Clusters

SAP earnestly began supporting high-availability clustering for Linux platforms with the *SAP Recovery Kit for Linux*, introduced in beta form in September 2002. Based on the LifeKeeper Recovery Kit for SAP R/3, SuSe Linux SLES 7, and Oracle 8.1.7, this SteelEye offering supports the following:

- Multiple configurations, including hosting the CI and DB on the same node, on different nodes, and separated into distinct failover clusters (some time after the initial release, by clustering the DB with a standby server and the CI with another standby server, for four servers total). In this way, active/passive and active/active cluster configurations are both supported, and heterogeneous systems are supported as well.
- Multiple and redundant heartbeat media types (instead of just IP, as is so common). For example, by using both an Ethernet connection and a serial port connection, the failure of one of these devices on one node will not cause failover to the other node. This approach not only eliminates a common single point of failure with most cluster solutions, but it also reduces unplanned and planned downtime.
- Configurable heartbeat parameters also provide another way to minimize downtime—in the case of robust and redundant heartbeat links, for example, fewer “I’m Alive” broadcasts should be necessary to trigger a failover, when both heartbeats fail to respond to the other node’s “I’m Alive” pings.
- SCSI, Fibre-Channel, and Network Attached Storage (NAS) shared-disk offerings.

SteelEye expects to port its HA solution to other versions of Linux, and Windows-based operating systems, soon. Support for additional RDBMS vendors is on the way as well. To read more about SteelEye and their solutions, refer to SteelEye Technology’s Web site, or read through their white paper entitled “Implementing Fault Resilient Protection for SAP R3 in a Linux Environment” on the Planning CD.

SPOFs and the Database Layer

To many, providing high availability for SAP often equates to protecting the database. Of course, as you have seen already, there is a lot more to HA than simply safeguarding the structure that houses all of your SAP enterprise data. But the database

historically represents a key single point of failure in SAP installations. Consider the following:

- Without the database, neither new nor old transactions can be created or referenced. Thus, all of the surrounding SAP infrastructure is worthless when the database becomes unavailable.
- Until recently, the database for an SAP instance could not be split into pieces across multiple servers. Of course, Oracle's 9iRAC now solves this age-old SPOF. But the lack of such a solution drove everyone—hardware, OS, and database vendors alike—to create high-availability approaches and offerings designed to protect either the database itself, or the data residing in the database.

Some of these approaches and offerings for protecting the database are discussed next.

Standby Database and Log Replication Solutions

At a hardware level, data is replicated between sites or disk subsystems disk block by disk block. OS-based solutions follow a similar approach, synchronizing the contents of a disk volume block by block. It might be surprising, then, that database-layer solutions replicate the data resident within a database in a different manner. In most cases, a database vendor provides high availability in one of two ways:

- By replicating data one transaction at a time, applying changes to a standby database via incremental database log updates
- Or, by creating a cluster such that the database and all of its associated services may fail over to another server node if the first node fails

Before I cover the various database-layer clustering approaches later in this section, let's investigate log-based replication schemes. An entire database can be replicated to another site, close by or far away, by using a method provided by all enterprise database vendors today—log shipping. This approach is sometimes known as “Standby Database” or “Hot Site Backup,” too, depending on the database vendor involved and the manner in which the data is being replicated. In all the following cases that I describe, the functionality necessary to create these standby or replicated databases is inherent to the database product. In other words, no special SAP server configuration or application-level coding is required to take advantage of these HA and DR solutions.

Oracle Standby Database Offerings

Probably the most commonly implemented HA solution for Oracle databases is the standby database. A standby database allows you to keep a nearly synchronized copy

of your SAP database at another location. Using products like Oracle9i's Data Guard, the less-capable Oracle8i Data Guard, or the older Oracle8i Automated Standby Database, the general process is quite simple:

- The standby server is set up, installed, and configured on a server at a remote location.
- A recent full database backup is restored to the standby server.
- The standby server is configured as an Oracle Standby Server, and therefore put in perpetual recovery mode.
- As the primary system creates redo logs and archives them, these archived redo logs are systematically copied to the Standby Server (perhaps every 15 or 30 minutes or so).
- After a period of time, the archived redo logs are applied to the Standby Server, such that the Standby Server is kept perhaps four or eight hours “behind” the primary.

By maintaining a delta of 4–8 hours between the primary and Standby Server, it's possible to recover from user errors and other data manipulation issues that would otherwise be applied to the Standby Server so quickly that no one would have a chance to realize that an error occurred and still have time to “back out” of it.

With this type of HA/DR solution, if the primary server fails, the secondary can be put into production rapidly—only the remaining archived redo logs that have not yet been applied to the Standby Server need to be applied. Worse case, if the Standby Server never received the last archived redo log, the Standby Server will only be 15 or 30 minutes behind the primary—this is normally quite acceptable, requiring the end users to log in to the new system and re-enter only 15 or 30 minutes worth of data and transactions.

HP Oracle Disaster Recovery (ODR)

Like Oracle's Standby Database offerings, HP's ODR also works by copying and applying archived logs to a secondary database site. ODR is automatically implemented using Perl scripts, the Oracle Server Manager, and OS-specific utilities, and is supported on both Windows-based and Tru64 operating system platforms. I like this solution because it's fast to implement, inexpensive, and benefits from essentially being just another log-shipping approach to HA/DR—failover time is quick, and the amount of data lost between failovers can be quite small.

Oracle Advanced Replication

Available since version 7.3, Oracle Advanced Replication allows updates to a primary database to be replicated to many additional databases. This replication occurs automatically, allowing for complete data integrity and transactional consistency. And the really nice thing is that each database, even the replicated copies, can be accessed by end users all day long. My favorite implementation of this HA solution allows for the disaster recovery site (or any secondary site housing a replica of the primary database) to be used for reporting, thus offloading the primary SAP database from this processor- and disk-intensive daily activity.

With some of the more common transaction and log-shipping approaches behind us, let's look next at how clustering is employed at a database level.

Clustering Microsoft SQL Server

When it comes to providing high availability for SQL Server, I tend to see both log shipping and MSCS clustering used most often, with the former approach being most popular. Of course other clustering software packages exist. But unless you want to be a lonely fish in a large sea (that is, the only company around for miles clustering your SAP instance with some unknown or unpopular HA solution), it makes much more sense to go with log shipping or MSCS clustering (where hundreds of other companies and consulting organizations can be leveraged for their lessons learned and other experiences).

Today, clustering SQL Server is simple. SQL Server 2000 is cluster-aware and automatically recognizes when it is being installed on a cluster. It requires that its executables be loaded on each node in a cluster, similar to Oracle's approach to clustering on the Windows platform. In this way, SQL Server itself can be easily patched and updated via rolling upgrades. In the past this was not possible—the SQL Server 7.0 executables were maintained out on the shared disk subsystem, for example. Patching or upgrading SQL Server 7.0 therefore meant incurring downtime, as the shared executables only existed in one place.

Oracle 8i Clustering and Failsafe

Like SQL Server 2000, Oracle 8i (8.1.7 is the final release of 8i) is also clustered on Windows platforms by using Microsoft's Cluster Service. In addition, though, Oracle also requires the addition of another bolt-on product called Oracle Failsafe (OFS), when clustering on Windows NT or 2000. OFS works by polling the Oracle database resource DLL to ensure that it has not failed. If an "Is Alive" message is not returned by the database after three consecutive database polling attempts, a failure status message is forwarded to the cluster resource monitor.

Clustering Oracle on top of various UNIX platforms, though, is even simpler. In these cases, scripts are used to control both failover and resource management, and a “failsafe” overlay product is not required.

9iRAC—Real Application Clusters

Oracle's 9iRAC is the biggest thing to happen to SAP high availability in years. 9iRAC once and for all eliminates the database as a single point of failure, and in doing so also provides tremendous scalability and solid near-linear performance as additional nodes are added to the cluster. Other compelling benefits or features of 9iRAC (as compared to most other cluster solutions) include the following:

- Transparent client failover capabilities, in that applications and users are automatically and transparently reconnected to another surviving cluster node, while their queries continue uninterrupted.
- Ability to add incremental cluster nodes online (I have installed up to six nodes in a single cluster, and more are possible). In this way, 9iRAC not only reduces downtime, but it allows SAP to “scale out” through inexpensive commodity servers, rather than forcing “scale-up,” which eventually drives platform and other expensive solution stack changes.
- Ability to add incremental disk resources without repartitioning existing drives.
- Moving from a 9i database to 9iRAC is seamless, requiring no database unload/DB load.
- Support for all of the popular operating system platforms.

Oracle 9iRAC is so effective because of the way it approaches utilizing each database node's data cache. In this method, which is called *cache fusion*, all local node cache is made available to all nodes in the cluster to satisfy database requests from any node. When an end user's transaction requests a data block, the system determines whether the request can be satisfied first by the cache residing in the locally attached database node. If the data cannot be found there, the request is sent over a high-speed low-latency link—the *interconnect*, as depicted in Figure 6.12—to each node's remote caches. If the data still cannot be found, an actual disk read cannot be avoided and therefore a physical disk read is performed—the slowest operation in any enterprise system, and the reason why high-speed disk caches were developed in the first place.

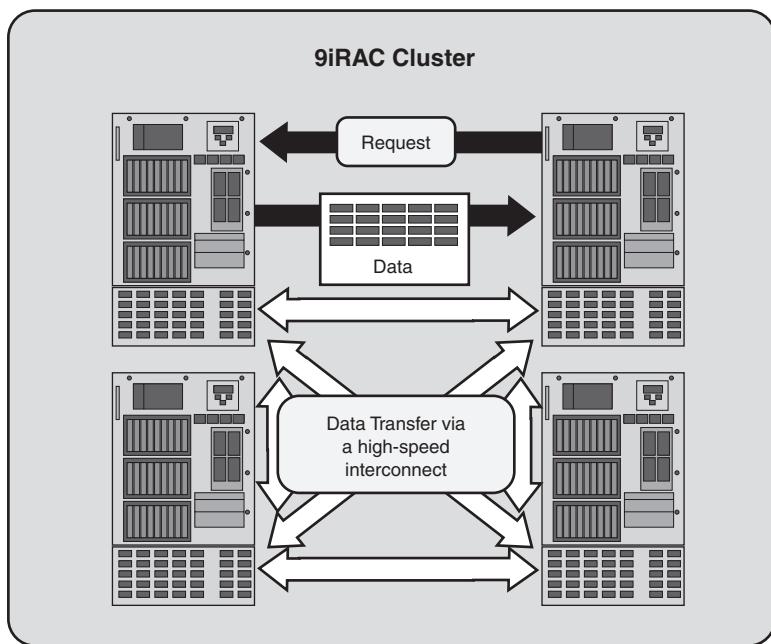


FIGURE 6.12 A 9iRAC cluster takes advantage of cache fusion and a high-speed interconnect to create not only a highly available solution, but a tremendously scalable one as well.

Thus, Oracle 9i cache fusion reduces disk I/O activity and synchronization, thereby speeding up database operations. None of this has anything to do with high availability, of course, but it sets the stage for where new single points of failure need to be mitigated. Specifically, the highly efficient inter-node message-passing mechanism, which handles the data block transfers between nodes, represents a key SPOF in two-node clusters, as shown in Figure 6.13—this SPOF is eliminated with the addition of a third node.

With such tremendous and obvious benefits, why isn't 9iRAC storming the world of enterprise applications like SAP? The product is still being piloted by SAP customers as I write this, so that's certainly one reason. But the number of pilots seems small compared to the potential benefits. I can only guess that price continues to play a pivotal role—the uplift for Real Application Clusters is something like 50% on top of Oracle's already less-than-modest pricing. As Oracle's Real Application Clusters start to prove themselves in the marketplace, though, I fully expect 9iRAC to be adopted throughout much of the Oracle installed base of database customers seeking the highest levels of availability and scalability.

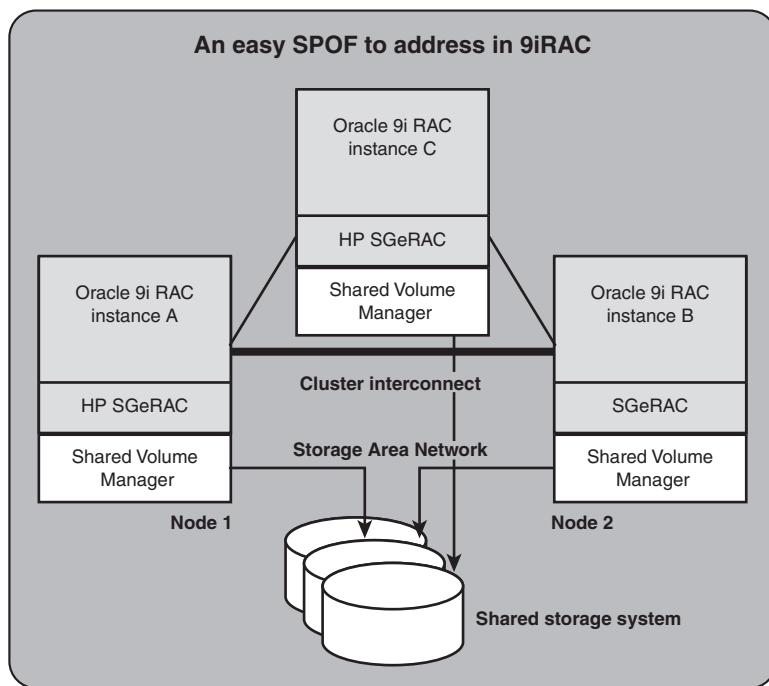


FIGURE 6.13 Although 9iRAC is both fast and scalable, a two-node cluster's interconnect (the red line between nodes) still represents a single point of failure. Simply adding an additional node or nodes resolves this issue by creating redundant interconnects (green links).

mySAP.com Single Points of Failure

SAP AG tells us that a “disaster is a situation in which critical components in the SAP environment become unavailable so that service cannot be resumed in a short period,” such as a few hours or days or so. Further, the two most critical SAP System components are: first, the database, and second, the SAP instance that runs the enqueue and message services—the central instance (CI).

In previous sections of this chapter, I described a number of methods and approaches for protecting the database. The topic of this section, on the other hand, is how to protect the CI and other SAP-specific single points of failure from both interruptions in service and full-fledged disasters.

Generally speaking, the CI can enjoy a high level of availability by having a standby system available (at a remote site), waiting to be started up in the event of a disaster. When a failover occurs to this standby system, all other application servers that

might exist as part of that system have to be restarted, though, or reconfigured to reconnect automatically. The same is true of a cluster node—all application servers must reconnect to the CI after the CI fails (relocates itself) from one cluster node to another. This fact underscores my point earlier in this chapter that it is as important to *avoid* failing services to another system as anything else. We accomplish this goal by piecing together a server with built-in redundancy, hot-pluggable components, support for proactive systems monitoring, and so on, in the hope that most failures will be addressed by this redundancy and therefore not initiate a failover.

Fortunately, the time required to reconnect to the CI is minimal, in the neighborhood of tens of seconds. And through the auto reconnect SAP profile parameter, this process can be easily automated and therefore almost appear as a seamless reconnect to the system's end-user community—most users will never even notice a CI failover!

Clustering mySAP.com Components in General

Any number of clustering technologies can be used to “cluster” an enterprise application. Even applications that are not *cluster-aware*, or written in such a manner as to record and track state information needed by a quorum or other cluster-tracking mechanism, can still be clustered. The drawback is just that no state information is maintained. The non-cluster-aware clustered applications can only be automatically stopped and started, as they know nothing about the status of a particular process or package, or the state of end users connected to the system, and so on. But they can still be made part of a package or group, and therefore have other dependencies defined for them (for example, a database service must be started before an application can be started). In this way, great value can still be obtained by clustering non-cluster-aware applications.

When it comes to clustering a mySAP.com component, we are fortunate in that SAP has indeed been cluster-aware since R/3 3.1i. The manner in which SAP supports clustering differs between various operating systems, however. Consider the following:

- UNIX-based operating systems typically support clustering a database and a central instance as either a single package (where both the DB and CI must run *together* on node A or node B), or individual packages (where each solution component can move independently between nodes).
- Further, UNIX-based operating systems typically support clustering a production instance running on node A with a non-production instance (that is, QA or Test) running on another node in a cluster. The idea here is that if the production node fails, the second node will stop the Test or QA services, and start up the production services instead.

- Application servers are not clustered—availability is achieved through redundancy and SAP logon load balancing instead, as these approaches are less expensive and less complicated.
- Windows-based clusters are much more rigid in terms of what is supported by SAP AG, generally speaking, than their UNIX counterparts. Only active/active clustering is supported, and then only between the Database and Central Instance, where each is defined in a separate group, and each can execute on either node in the cluster.
- Therefore, SAP AG does not support Windows-based clusters hosting production on one node with, say, Test on the other.

My point here is simply that the CI can be clustered in an active/passive manner (clustered with a standby server) regardless of operating system environment. I have customers doing just that, running heterogeneous systems where the CI and DB reside on different hardware/OS platforms and therefore cannot participate in a single shared cluster. However, each individual resource—the DB, or the CI—can be clustered such that a standby server is sitting idly by, awaiting the failure of its associated DB or CI partner.

More often, though, when high availability is required for the CI, a product like HP's Somersault or SAP's new Replicated Enqueue offering is brought in.

HP Somersault

When the enqueue process fails, all transactions executing at that point in time are aborted. HP Somersault was the first product of its kind, able to minimize downtime due to failure of the SAP Enqueue server/service, whereas other HA approaches like MC/ServiceGuard clustering (with the ServiceGuard extension for SAP) could only protect against DB and Message Server failures. Used together, a highly available SAP system was made possible. It makes perfect sense then why SAP only supported HP Somersault when used in conjunction with MC/SG and other solutions that addressed the remaining three SPOFs outside of the CI (the DB, Message Server, and NFS).

And Somersault was unique in its day, too, in that it was transparent, replacing the standard enqueue process. Simple in its approach, Somersault was launched by MC/SG as it started the SAP central instance executables. It's no wonder that SAP AG finally introduced a competing product, the Replicated Enqueue. Meanwhile, HP recently announced that it will be transitioning its own mySAP customers to SAP's offering, too, covered next.

SAP Replicated Enqueue

Nearly identical to Somersault in terms of functionality, SAP's Replicated Enqueue Server prevents data loss by copying data to a redundant replication server. As shown in Figure 6.14, if the primary enqueue server fails, the secondary server takes over the role of the enqueue service, maintaining status information on SAP object locks. Some key SAP Replicated Enqueue product differentiators over Somersault are as follows:

- It is not platform- or vendor-dependent.
- It does not change the executables; thus, both HA and non-HA implementations are identical in this regard, simplifying support and change management.
- It's fast, both from a failover perspective and in regard to communications between the CI and other SAP elements.

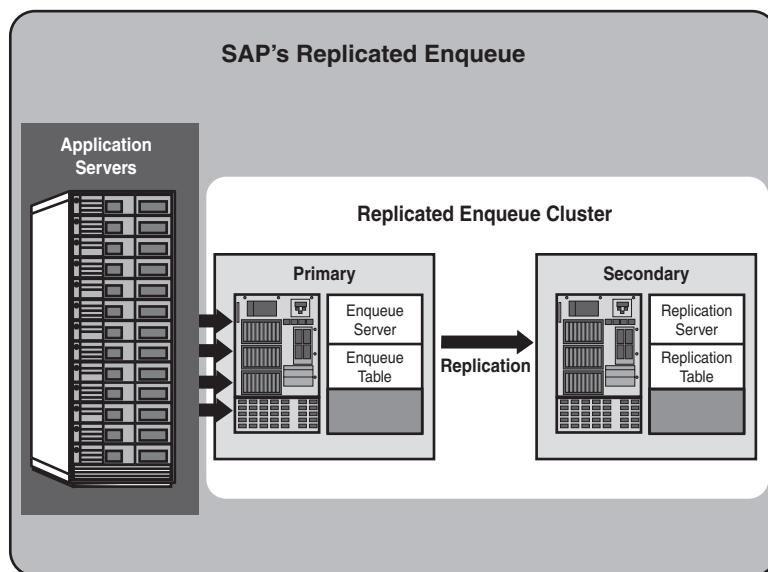


FIGURE 6.14 SAP's Replicated Enqueue removes the Central Instance's Enqueue process as a single point of failure, and does so in a hardware- and OS-independent manner.

SAP's Replicated Enqueue takes advantage of a multithreaded architecture to allow parallel I/O to enqueue clients via TCP/IP-driven Asynchronous replication. Transactional integrity is maintained in that enqueue responses are delayed until the replication is finished. The failover process is as follows:

1. The failure is detected and the original enqueue server is stopped.
2. The virtual enqueue host is switched to the physical host acting as the replicated enqueue server (the “new” enqueue server).
3. The new enqueue server reads its replica copy from shared memory.
4. The enqueue process is actually started on the new enqueue server.
5. The replication server itself is stopped and restarted on any new host available, in an effort to again protect the new enqueue server from failure.

With many of the general SAP-layer single points of failure covered, let’s turn our attention to single points of failure relevant to individual mySAP.com components next.

Specific mySAP.com Components SPOFs

Each mySAP.com component offers unique single points of failure that need to be evaluated and then either eliminated or mitigated in each production system.

Remember, all mySAP.com components are subject to the DB, CI, Enqueue (and in the case of UNIX-based systems, NFS) single points of failure already discussed. But the following list identifies the SPOFs with which each component is faced, and offers suggestions for improving high availability where possible:

- APO—both liveCache and the APO Optimizer(s) represent single points of failure. With APO 3.0A, fortunately, it is possible to create a very robust active/active APO Optimizer Cluster using MSCS. For each Optimizer, two RFC destinations are configured, one pointing to Node A and one pointing to Node B. Therefore, both nodes can be used during production, and in the event that one Optimizer fails, the transaction is simply rolled back and re-executed on the remaining node. Additionally, liveCache 7.4 supports more than simple active/passive clustering. Finally, high availability can also be augmented through APO-specific monitoring tools like HP’s SAPLC.mon utility, which is used with MC/ServiceGuard to monitor whether APO liveCache is up and available.
- BW, specifically the server(s) that actually reside in other SAP landscapes (like R/3), loaded with BW extractors used to pull transactional data and then populate BW with this data. Thus, it’s necessary to ensure that the BW extractor server is not a single point of failure. A functional SPOF exists, too, if the BW production server is designed with only a single server supporting data mining or other similar activity. This is easily resolved, though, with a little forethought.

- CRM, in the form of the Internet Pricing Configurator and other special function servers or links to external systems and components. If any of these are not duplicated functionally, for example, through an active/passive cluster, they represent SPOFs. Also, the R/3 Adapters used by CRM pose the same challenges as those noted for the BW Extractors.
- EBP, specifically the Requisite Catalog Server and all connectivity points and software integration products, like JRUN (or other virtual java machines). An active/passive cluster is an easy way to resolve this. A more compelling approach involves using the staging catalog server or another catalog typically earmarked for testing new products/procurement processes or providing access to vendors to make routine product line changes. Thus, an active/active cluster can be maintained (active in the sense that each node is performing useful work, though not in the sense that they are both able to share the load between them and thus improve performance).
- EP, including typical access points “in front” of it (such as any hardware-based load-balancing gear front-ending Enterprise Portal). Other SPOFs are the LDAP Directory server, TRex, Portal Content Directory (including IIS files), and the Unification and Lock servers, all of which can be mitigated today by the use of active/passive MSCS clusters. Note that for active/active clusters, both MSCS and load balancing must be implemented. And unfortunately, if you are using the Sun One (iPlanet) Directory Server, a fully protected active/active implementation is simply not possible. The use of other directory services may allow you to circumvent this shortcoming, however.
- ITS (until the advent of version 4.6D) suffered from the same shortcoming that plagued it since its inception—the WGATE could only maintain state information with a single AGATE, so HA equated primarily to redundancy through availability. My favorite configuration involved four very inexpensive WGATE servers, consisting of two clusters of two servers each, with each WGATE cluster communicating with a single AGATE (for six servers total). In this way, multiple WGATES and even one of the AGATE servers could fail before end-user accessibility is impacted. Today, these issues are resolved both with ITS version 4.6D and the advent of Web Application Server, as ITS implementations slowly give way to WebAS-based solutions.
- KM, specifically the Content Server. Similar approaches discussed for protecting the EBP Catalog Server can be employed here, or an active/passive cluster can be used.
- PLM, specifically the Variant Configuration Server, not to mention links to all other mySAP.com components needed to make PLM work, including R/3, BW, APO, CRM, and KM.

- Workplace (WP), or actually accessibility to Workplace. At one of my customer accounts, we failed to really think about Workplace high availability, even as we busily constructed a highly fault-tolerant Production R/3 cluster. Because WP represented the core access vehicle to R/3, we missed the boat by a mile—today, WP is also clustered at this account, and the classic SAPGUI represents a second accessibility option in the event that WP (with its Web-browser-based front end) fails completely.

Certainly, other products and components exist or are soon to be released. R/3 Enterprise, for example, further reduces downtime by virtue of support for system switch upgrades—upgrades to the R/3 core are reduced to simply applying service packages, and functionality updates are now handled by new SAP R/3 Extensions. The point here, though, is to do your homework and carefully study any mySAP.com component to be deployed. With the material provided in this chapter underpinning knowledge you gather on your own, you should feel pretty competent identifying single points of failure with the help of your SAP Solution Stack partners.

Functional and Application-Layer SPOFs

Some of the worst SPOFs that I have run into are “self-created.” In other words, single points of failure were created where none naturally had to exist. The following list provides some examples, and hopefully some insight, into how to avoid these:

- All SAP user-specific work processes must be distributed, such that a single server does not run all of one type of work process and therefore become a SPOF. This includes Dialog, Background, Update, and Spool work processes. In other words, shy away from creating a single batch server, a dedicated spool server, or a lone update server, unless the ramifications and trade-offs are clearly understood.
- Logon load balancing should be configured such that no functional area is designated to a single server (unless, of course, this is determined to be acceptable). Why? Because if this server fails, the functional area is in effect unavailable. Workarounds abound, fortunately—the most common is to allow users to log in to a different logon group when their primary group is unavailable.
- Any software loaded on a single server automatically creates a single point of failure should that server crash or otherwise become unavailable. Thus, for the highest levels of availability, BW Extractors, or Vertex accessibility, or similar integration/connectivity points should be configured on more than one server.

In the next section, I build on everything we have covered thus far and investigate HA and DR outside the neat borders of the SAP Solution Stack.

SAP General Availability and DR Best Practices

It's no secret that lots and lots of "best practices" exist for high availability and disaster recovery. We have just poured through many of these in detail. But I believe it makes sense to compile a top-ten list here, so to speak, before we move into more DR-specific considerations.

- At the lowest level, or where it makes the most financial sense when everything else is equal, practice "Availability Through Redundancy."
- Clustering—where possible, cluster resources to maximize availability or in some other manner provide redundancy of critical solution components. Simple active/passive clustering should be considered as a baseline HA approach, to which other approaches are compared in terms of the degree of availability provided, complexity, cost, and long-term supportability. And don't forget to create a similar cluster in your testing or sandbox environments, for failover testing and training!
- Always perform at least a cursory TCO analysis to roughly determine the point where downtime exceeds the cost of a high-availability or disaster recovery solution, based on the level of availability requested by the business. This therefore requires knowledge of the number of "nines of availability" that your business truly requires.
- Work methodically through the SAP Solution Stack, identifying single points of failure at each layer. When you have identified them, refer back to the cost of downtime to determine whether it makes financial sense to eliminate or simply reduce/mitigate the SPOF.
- Step back and look at the mySAP.com solution holistically, to identify less-obvious technology-related, process-related, or people-related single points of failure—SPOFs that exist "outside" of the SAP Solution Stack.
- Manage user access to data, security rights, profiles, the ability to launch batch jobs, the ability to tie up multiple dialog processes for long periods of time, and so on. In this way, overt "denial of service" is addressed as well as honest user-originated mistakes and other acts, all of which can easily impact availability.
- Ensure that the solution finally implemented is documented "as is," noting all deviations from a standard installation or implementation. I have always been a big fan of documentation, as it serves so many roles—operational processes rely on repeatable processes, disaster recovery plans dictate a rebuildable SAP solution, and training materials evolve from documentation as well, for example.

- Meticulously plan and test all changes to your SAP environment, being careful not to ignore your change control processes even in emergency situations. Improperly tested or executed *changes* made to a system tend to impact availability as much as anything else, in my experience.
- Ensure that all operational procedures are documented and understood—publish HA and DR policies and business continuity plans via a formal Communications Plan. Further, create and maintain a *Disaster Recovery Crash Kit* at your DR site, covered in more detail later in this chapter.
- Test your HA and DR plans on a regular basis, ensuring not only that the entire team understands and is trained in HA/DR processes, but that the team is capable of handling a disaster even when key human resources are missing. One of my favorite ways of testing a DR plan is by “killing off” the senior SAP Basis administrator or DBA position, and watching how the team backfills and otherwise responds to meeting the needs normally addressed by that key position.

In the end, human error and poor attention to managing change probably constitute 60% of what I describe as “avoidable” downtime—that’s the unplanned downtime that should never really happen in the first place, which is a lot different from the unavoidable downtime incurred so rarely from actual solution stack failures and honest-to-goodness disasters.

In the next few pages, I detail some of these top-ten best practices, and the challenges inherent in nailing them down.

SPOFs Beyond the SAP Solution Stack

In addition to technology-specific single points of failure like those you have already read about at the hardware, OS, database, and mySAP.com layers of the SAP Solution Stack, SPOFs regarding people and processes exist as well. Consider the following:

- All key personnel positions require backups and documented backup plans. This should include earmarking and training internal folks, as well as establishing contractual relationships (for example, executing “consulting on demand” agreements) with third-party resources.
- A solution’s client strategy could inadvertently create a single point of failure, especially if multiple languages are involved. This needs to be addressed in the system landscape.
- All processes related to change management, deployment, management, and operations represent potential single points of failure. Two methods of performing each process should be clearly identified and documented in each case.

- Tools and approaches used to monitor a system or execute certain processes should also be duplicated. In the best case, manual processes should be documented and tested regularly, ready to be used when automated approaches fail.
- Access to documentation must be addressed, especially with regard to disasters. I like the idea of maintaining documentation on a company intranet site or public password-protected Internet site, along with maintaining a printed copy both at the primary data center and at the DR site.

Other single points of failure can be in a sense “created” as well. To combat this, my suggestion is to have your entire DR plan reviewed by a “new” pair of eyes every year. Doing so eliminates oversights caused by people who might be too familiar with the system and therefore prone to assume things that would otherwise warrant further discussion or clarification.

The Disaster Recovery Organization

When it comes to people-related SPOFs, few are as important as those discussed here. Four roles in the *Disaster Recovery Organization* (DRO), a subset of your SAP Technical Support Organization, are key. The number of folks performing these roles may vary from site to site, ranging from a single individual in a small SAP shop to perhaps four or many more people in larger implementations. SAP AG defines the following key DRO roles:

- Recovery Manager, responsible for managing the entire actual disaster recovery process from a technical perspective. All DR activities and issues are coordinated through this critical single point of contact.
- Communication Liaison, also typically a single point of contact within the DR, tasked with handling all communication between the SAP TSO and upper management, and the SAP TSO and other technology-focused support organizations. By keeping the management team apprised of the recovery status, and handling all phone calls into the DRO, this critical role helps ensure that the technical disaster recovery process can continue uninterrupted.
- Technical Recovery Team, from very few to perhaps a half dozen or more SAP Solution Stack specialists. These folks perform the actual technical DR process, seeking to both manage changes to the recovery plan and coordinate the actual technical recovery as they see fit.
- Review/Certification Manager—When the system is backed up, this individual coordinates and plans all post-recovery testing and certification with end users, to ensure that no data loss was realized or other failures inadvertently initiated.

Organizing the DRO around these four roles is a challenge in and of itself, too. Successful disaster recovery is more than simply an exercise in technology; it is an organizational challenge, and thus requires advance planning. People need to understand their roles, the responsibilities they have during a disaster, and what level of commitment is needed both during real disasters and in support of DR tests.

Decision makers need to be identified early on, too. In other words, who will actually make the call to fail over to a DR site? Who declares the disaster, or decides that an event is not worthy of DR actions? In most cases, the recovery manager plays the key role here. But other folks, normally executive-level, may play a role too, and certainly systems administrators will be involved by virtue of their support roles—this all needs to be ironed out well in advance.

Finally, basic human resource management needs to be considered. This might consist of addressing the need to identify and obtain backfill personnel for people unable to fulfill their DR roles during an actual disaster. In turn, internal staffing processes and even external support contracts may need to be revised or put in place.

The Disaster Recovery Crash Kit

With people considerations behind us, we can now investigate the other resources and considerations inherent in successfully recovering from a disaster. The *DR Crash Kit* is your fundamental DR resource, containing everything needed to recover your mySAP.com production environment in the event of a disaster. It is maintained at the DR site itself, and consists of the following:

- Software necessary to *rebuild* the production system—All operating system files, any hardware-specific drivers, the database/SAP installation kit, and all solution stack patches, support packages, service packs, current OS and SAP kernels in use, SAP profiles, and so on. Additionally, this includes any other software packages absolutely required by your production system, like tax packages or other bolt-ons, that can be added to your mySAP.com environment over its lifetime.
- Software necessary to *access* the production system—any utilities; SAP licenses; the SAPGUI installation CD; other user interface packages (like Internet Explorer or legacy UIs); the saplogon.ini, tpparam, and saprouttab files; software used to access the servers and other hardware components (like Terminal Services, VT100 terminal emulation software, and so on); the Internet Transaction Server installation kit and a copy of all up-to-date HTML load templates; and so forth.
- Administrative and management-related software necessary to ensure *operational integrity* when your production system is running at the DR site, including backup/restore and related utility programs, UPS control/management programs, hardware monitors and other systems monitors, and so on.

- Detailed *current state* information as to how the system needs to be laid out (like file system details), or set up (like hardware specifics related to cabling, physical disk layouts, PCI card slot locations, and so on). I recommend the use of a detailed hardware and software “matrix,” which I have used extensively in the past to document just this kind of detailed configuration data. This matrix is discussed in more detail in Chapter 14 (and a sample matrix may be found on the Planning CD, too).
- Step-by-step documentation necessary to install the productive system, all access methods, and all operations management utilities/applications. Additional documentation explaining how to restore the database and potentially other critical data (control files, and so on) to a specific point in time is necessary as well. I like maintaining both electronic and printed media of all documentation in the crash kit. And I highly recommend creating and maintaining a “Master DR Installation Checklist,” which breaks down all installation tasks that must be accomplished, and in what order, as well as who is responsible for performing these tasks.
- Operational documentation, like the regularly used daily and weekly checklists, information on how to use your specific enterprise management products, and so on.
- Administrative items, like hardware/software service agreements (with contact phone numbers and email addresses for subject matter experts), any special instructions as to how to recall tapes from offsite data storage or acquire replacement gear, escalation procedures, a comprehensive company-wide and partner-wide phone list, and more.
- An inventory list, so that anyone can quickly tell whether a piece of software or a documented procedure can be found in the DR Crash Kit, or determine that something is missing.

I cannot stress enough how important good documentation becomes during an emergency or disaster. Your goal should be to remove as much as possible of the guesswork related to rapidly reinstalling or failing over a production system under adverse conditions *before* a disaster ever occurs. This means staying on top of changes as they are made to your production system over time, so that the DR Crash Kit truly reflects what is needed to restore the production system “as is” as it evolves. For example, when changes are made to the SAP Solution Stack, all installation, support, maintenance, and other operational procedures need to be updated immediately.

Of course, these updated procedures are worthless without the updated requisite software—ensure that the outdated items in the crash kit are updated immediately with these new versions. And finally, review the entire contents of your crash kit on a

regular basis (like every quarter). Use the kit's inventory list as a basis not only for checking that every crash kit component is present, but to track changes to each component—a good inventory list will reflect the date that any particular item was last updated, by whom, and why.

All of this invaluable software, procedures, and so on must not only be maintained at the DR site, but it must also be easily available or accessible. As I mentioned before, placing documentation out on an intranet or secured Internet site makes a lot of sense. But I also recommend that the physical components of the DR Crash Kit—things like software CDs, driver diskettes, restore tapes, and so on—be preserved at a second site too. This second site might simply be another company site, or even a commercial offsite data storage facility. Or you might make it a policy to keep everything “online” and dump all data to a file share or Web site. The point is this—during the middle of a disaster, you want to be able to fall back to another set of media if your primary media fails. We all know that tapes go bad, CDs get scratched, and things sometimes get lost. Protect yourself in this regard, and the extra day or two of effort beforehand may pay big dividends later on.

With regard to the contact information that must be maintained in the DR Crash Kit, I find the following contacts most useful:

- All members of the SAP TSO, SAP DRO, and especially the SAP Recovery Team
- Facilities personnel, like the people responsible for power, network infrastructure, access to the data center and DR sites, and so on
- Any IT infrastructure folks not already covered, like perhaps IT security
- Key users from each core functional area or mySAP.com component, to verify after a failover that the DR site is functioning as expected
- Key consulting resources, like those from SAP, your hardware vendor, disk subsystem vendor, OS vendor, systems integrators, and so on
- SAP Hotline
- Offsite data storage/tape storage personnel
- Physical security, should physical access to the UPS room or into the data center (or the building housing the data center or DR site itself!) present a challenge
- Anyone with whom you have service agreements or contracts, who is tasked with providing or potentially providing support during an emergency

Additionally, access to the Internet (for miscellaneous online SAP Solution Stack vendor support, or to facilitate Internet-based troubleshooting or access to email) is highly recommended, and often required anyway.

Testing the Disaster Recovery Process

Because recovering from a disaster is so complex, and involves many dependencies on technology, people, and processes, your mySAP.com disaster recovery process must be integrated with your company's general disaster planning. This process includes how to obtain telephone, network, product deliveries, mail, and other basic services in the event of a disaster, and so on.

With all of the technology, people, and process complexities surrounding recovery from a disaster, all of this must occasionally be tested together as well. Why? Because unless you actually test your recovery process, you will never be assured that your particular SAP system can truly be recovered.

A good test, then, involves recovering from a simulated disaster to verify not only that you can recover the system, but that you can also use it productively. And a really comprehensive test includes testing people and process SPOFs, not just the failover process to a DR site. That is, a true disaster recovery plan forces an organization to actually use every component of their DR Crash Kit and perform every task outlined in the disaster recovery plan to determine if

- The DR Crash Kit is appropriately stocked.
- The disaster recovery procedures actually work.
- Any documented steps need further clarification, in the event that a “non-expert” is called in to perform that particular process or procedure.
- Any changes exist that have not been documented. Changes include current-state as well as DR site-specific solution stack details.
- All infrastructure services are available and work as expected, like DNS or SMTP servers or similar network services.
- The hardware housed at the DR site, and the supporting network and other connectivity infrastructure, provide acceptable performance at the end of the day, so to speak.

Actually working through the disaster recovery process provides valuable timeline information to everyone, so that, for example, the SAP Disaster Recovery Organization understands how long the recovery process actually takes, and the end-user community understands this as well.

Just how often should a comprehensive DR test be conducted, though? Most experts agree that an annual test is appropriate. Again, though, a rushed-through, ill-planned DR test will do little to assure a company that it can survive a disaster. Not only is a good DR test comprehensive, it is also unique—no two tests should be identical. Further, it includes the actual SAP TSO and other personnel who will be responsible for failing over the SAP environment to the DR site in a real disaster, though

with one important caveat—at least *one* staff member needs to be “killed off” or otherwise made unavailable during the DR test. In this way, the test proves the ability of the rest of the team to pick up the slack and perform a successful recovery, representing one of the worst implications of a real-world disaster.

The DR test also needs to include any third parties that are critical to the recovery. Anyone with whom a Service Level Agreement or other contractual obligation is in place must be present to “prove” that they can truly provide the DR services needed in the event of a real disaster. If this level of service is not included in the contract, it should be.

Finally, the DR test needs to cover how an organization communicates status updates to all of its stakeholders—end users, customers, senior management, the SAP Technical Support Organization, and so on. How will these key stakeholders be kept in the loop? What kind of additional communication responsibilities need to be doled out, and to whom? Answering questions like these before a disaster makes it easier to focus on recovering from a disaster during the real thing.

Availability Pitfalls and Mishaps in the Real World

To help you not underestimate the importance of high availability and disaster recovery, I think it makes sense to take a quick look at a few cases where SAP enterprise customers failed to give *general availability* the attention I have given it here.

- End-to-end critical points of potential failure—This Tru64/Oracle customer spent much time working on Database and SAP application-layer-specific single points of failure, but missed basic infrastructure SPOFs. A bad network switch and a less-than-optimally configured NIC (set to auto-sense, rather than hard-coded for 100mbit Ethernet) caused intermittent failures for literally months. Had they practiced “availability through redundancy” and followed NIC configuration best practices, they could easily have eliminated most of their resulting downtime.
- Compressing change management—in an effort to “get a change in fast,” this customer did what many of us have done before, and compressed the promote-to-production process. Instead of keeping a change in their technical sandbox for a few weeks, and then promoting it to development, test, training, and finally production (which normally consumed another six weeks), they pushed the change through in less than a week. I was happy to hear that they at least went through the process, instead of ignoring it altogether. In the end, though, an issue that only manifested itself over a period of time usually greater than a week caused them substantial and recurring unplanned downtime.
- Poor documentation—I provided one customer of mine a slew of methods for maintaining their SAP disaster recovery processes and procedures. I covered the

pros and cons of using Web sites, file shares, SAP Enterprise Portal, Oracle Portal, Microsoft SharePoint Portal Server, and even a simple Excel Workbook approach. In the end, they never even made a decision. And when they suffered an unplanned outage last summer, not only did they have little consistent formal documentation to help them, but the fact that it was spread out all over the place without the benefit of centralization and version control only exacerbated an already tense issue.

- Change management shortcomings—There are so many “keys” to sound change management that I can only recommend reading Chapter 13 in its entirety. But the real gotchas probably boil down to ignoring change management, poorly documenting and therefore poorly implementing change processes, and compressing the promote-to-production process. With regard to this first point, my colleague and I received a call from a frantic outsourcing partner at 6:30 a.m. one morning. They were in the middle of doing an upgrade, but never actually tested the firmware updates associated with updating their disk subsystem to new technology. Four hours later, they were back in business, but they blew their SLA for unplanned quarterly downtime by a wide margin that day.
- RAID 5 limitations—Here’s an example of a problem that could easily have been avoided. My really small Oracle/SAP customer had some employee turnover, and a year after I designed and helped to install their production system, they had a new SAP operator/junior basis guy on staff. He was pretty good with Oracle and SAP but apparently had no real understanding of hardware, including RAID 5 limitations. Anyway, one day he noticed a disk drive glowing amber instead of the usual green color, and he soon determined that he lost a drive on his database. Rather than replacing it, he apparently sat around for weeks looking at the failed drive, comfortable in his ignorant assumption that he could lose plenty more drives before he really needed to worry. The whole time, a couple of replacement drives (hot-pluggable ones at that!) were sitting in a cabinet in his same small data center a few feet away. Anyway, he learned about the limitation the hard way—he lost another drive. There are a lot of “worse practices” illustrated in this example, as I’m sure you’ve already surmised. Unsurprisingly, though still sadly, he did not make it through a round of layoffs later that year.
- More employee turnover, and documentation—Another customer laid off two operators, and no specific person was ever held accountable again for maintaining SAP operations/monitoring and related “how-to” process documentation. As it turned out, no one actually monitored SAP on third shift for months, and a problem eventually cropped up. But with no knowledge of how to identify the issue, much less how to troubleshoot and resolve it, half of the SAP Technical Support Organization was awakened that night.

- No training—In a similar case, a company's HA failover mechanism for SAP R/3 worked nicely, but the people who were formally responsible for failing the system *back* to the original site were reorganized and new to the team, and had neither been trained nor apprised of where to obtain the system documentation. Think about it—no one still on the team had ever actually gone through the procedure for moving the production system back to the original site and pointing end users to it again. They got lucky and were able to contact one of their former colleagues who had moved into a new role in the company but still remembered the process (and had the documentation, too). Fortunately, their “lessons learned” paid off afterwards, before a real disaster occurred.
- No testing performed—Actually, I am really referring to stress-testing in this case. This customer opted to incur the risk associated with foregoing stress testing, which had included a provision for testing basic failover capabilities while a load was on the system (note that the savings was \$45–60K in this case, pretty standard for a 15–20 business process OLTP transaction-based stress test with 400 virtual users). After they went live, they found the usual ABAP issues early on, and discovered some other easily resolved performance problems related to the number of background processes deployed. More importantly, though, SPOFs existed in that they deployed only a single server to run all background work processes. The SAP EarlyWatch service had caught this, and the customer had fixed it weeks before Go-Live. But an enterprising junior Basis Administrator took it upon himself to change it back without understanding the consequences. A stress test would have caught this sooner, both from a performance perspective and in the end-to-end system review that I perform as a prerequisite to testing. And everyone would have understood the consequences.
- Single point of people failure—Only one person in the entire technical organization at this company, the DBA, knew how to set up and configure the disk subsystem for SAP. When he left the company, though, this fact was not fully understood until the system suffered a failure, and many people tried unsuccessfully to access the password-protected system. When that failed to work, they blew away the system and actually reconfigured the disk subsystem, incorrectly. Lots of people were called that weekend. Bottom line, “availability through redundancy” does not apply only to hardware- or process-related single points of failure. It also applies to people.
- Communication—The SAP IT staff went to a lot of trouble to script a really nice failover/fail-back routine for their production SAP cluster, and they even set up an alternate access method for getting into SAP R/3. But they failed to share *how* with end users! The lack of communication rose to the surface as they ran into a variety of issues during their quarterly mini-DR test, including

issues with using the SAP WebGUI versus the classic SAPGUI. In failover, they could no longer connect to their ITS WGATE cluster front-ending SAP Workplace, and had to fall back to directly logging into the R/3 Central Instance using the SAPGUI.

- Classic example—A very large customer of mine decided *after they had already sized for and purchased their SAP storage solution* that they needed a SAP DR system installed 500 miles away from the primary site. And they needed to meet some really aggressive failover times. Of course, by this time in the project, the customer was already live on core R/3 and BW functionality, having implemented Compaq ProLiant Servers running Windows NT and SQL Server. The DR solution they actually needed was only supported at that time in a UNIX/Oracle environment, though, leveraging a specific model of disk subsystem running a specific version of disk controller firmware. Big oops... that's what happens when DR is at the back of the book, or in this case at the back of the project plan.

It's a shame that practically all of this downtime I just described was actually avoidable. My real-world stories have another thing in common, too—each of these customers would benefit from a regular high-availability review, not to mention more consistent or vigorous DR/business continuity testing.

A Final Look at the SAP System Landscape

With everything I have covered thus far in all of the chapters up to this point, you should be feeling comfortable as to how your potential SAP system landscape design is evolving and shaping up. Your specific availability requirements have narrowed down your disaster recovery and high-availability alternatives to a precious few “best-fit” approaches, thereby driving server, disk subsystem, operating system, and perhaps even database choices. This final section closes Chapter 6 with how to use this information going forward, as we prepare for Chapter 7 and sizing.

Preparing for the Next Big Step—Sizing

Soon, the project solution architect will begin working directly with various hardware and software vendor organizations adept at designing and deploying mySAP solutions. The SA and participants from each potential solution stack partner will work together to further refine availability, performance, scalability, manageability, and other solution characteristics. I suggest that the solution architect step back and gather supporting data to answer the following questions related to availability:

- How many “nines of availability” do you think the system requires? What does the business say?

- What is the maximum amount of time you can be “down” before you must fail over to a DR site? Failing and then bringing up production on a DR site takes time, and ultimately requires failing back again to the primary site later.
- How big are your backup and restore windows, in hours?
- How “extended” does your mySAP.com component need to be? That is, will you require highly available front-end (ITS, Web Application Server, load balancing, other) or back-end solutions (communications with or between other enterprise products)?
- What are your server and disk subsystem hardware platform preferences, and why?
- What is your operating system preference, and why?
- Similarly, what is your database preference, and why?
- What high availability and disaster recovery approaches are you considering, and why?
- Do you have any special high-availability requirements that need to be considered (specific integration or other “touch” points, and so on)?

The solution architect is bound to discuss these same points over and over again with each vendor, and answer similar HA-related questions on each vendor’s version of the SAP sizing questionnaire. And these are just the HA questions! It’s no wonder, then, that larger project implementations lean toward issuing a *Request for Information* (RFI), rather than going through the sizing process with 10 different vendors vying for the same business, asking the same questions. The RFI is the topic of our next section.

Completing the Knowledge Repository or RFI

Regardless of whether an SAP project is issuing an RFI to a bunch of SAP Solution Stack prospects, or simply going through the standard sizing questionnaire process, all of the information collected during the activities described in this chapter needs to be captured and documented in the knowledge repository. Remember, the knowledge repository is simply our documentation vehicle where assumptions, constraints, and so on are all documented.

For those SAP projects leveraging the RFI, the knowledge repository serves another purpose—it allows you to put together a well-conceived RFI. And to the point here, it allows you to nail down and publish any pre-determined “hard” availability requirements, and share this information consistently with all of the prospective vendors. Further, as new information comes to light, or questions are posed by these

prospective vendors, the knowledge repository will naturally lend itself to collecting and adding this incremental data to your growing list of constraints, assumptions, requirements, needs, and so on.

Revising Your SAP Implementation Budget Again

In the end, maintaining the “running history” of HA and DR requirements versus constraints published in the knowledge repository will also help to justify (or at least explain!) the changes that you will need to make to your SAP implementation budget. I fully expect my own SAP clients to review and update their budgets significantly when it comes to planning for and implementing availability solutions. As I’ve said before, it’s not uncommon to see the initial sticker shock of a five-nines solution change the perceived availability requirements otherwise demanded by the end-user community. But I have seen it work the other way around, too, where the business drives greater availability requirements than had been originally envisioned. And in both cases, budget numbers change pretty dramatically.

Even at this stage in the project, where an official vendor-provided sizing is still a chapter away, so to speak, a good solution architect should still be able to assemble the costs of an HA/DR solution that approximates what will eventually be implemented. If you have a feel for what you need—if you were able to answer most of the questions posed in the previous section that relate to sizing and availability—it’s pretty simple to cobble together the costs of individual solution stack components, and come up with a dollar range. I suggest you then start working with the upper end of that range, and if necessary begin whatever process you need to begin, to make changes to the budget. If you have been conservative, you should require no changes at this point—but I still recommend that you don’t relinquish any budget money until you are well past the entire sizing process. On to Chapter 7!

Tools and Techniques

Quite a number of supporting documents can be found on the Planning CD, not all of which are listed here. But I have tried to include pertinent white papers and other published documents that relate to many of the HA and DR areas we covered in this chapter. For example, information regarding installing and troubleshooting MSCS implementations for SAP on HP’s ProLiant platform can be found in the “MSCS Clustering for SAP on SQL2000 or ORA81x” PDF. And details regarding SteelEye Technology’s HA offering for SAP/Linux solutions, different ways to implement Oracle HA solutions, and much more can be found on the CD.

All of the figures found in this chapter are also included on the Planning CD, in Microsoft PowerPoint format. I suggest that you leverage these where you can as you assemble your own HA and DR presentations to deliver to your colleagues, management team, RFI team, and so on.

Summary

This chapter addressed high availability and its more serious companion, disaster recovery. HA and DR were compared and contrasted, and basic tasks like identifying the cost of downtime and documenting HA drivers were covered. You discovered that DR seeks to provide production-level coverage in the absence of the production system landscape, and that this could mean many things, depending on the configuration of the system. The SAP Solution Stack model was then used to break down general availability into core offerings and approaches, based on each individual layer of the stack. As you learned, each layer featured quite a few high-availability options, but only a few layers really provided robust DR capabilities.

After walking through the solution stack, we stepped back and identified end-to-end HA and DR challenges, and next spent a considerable amount of time really fleshing out what disaster recovery planning means in the real world. I then concluded the chapter by revisiting the topics of sizing, the role of the knowledge repository, and budget considerations, all of which are covered in vast detail in the next chapter.

7

Sizing: Engaging the SAP Solution Stack Vendors

Overview—The Sizing and Blueprinting Process

If you have been diligent in following the processes outlined up to this point, all of the information collected with regard to your SAP project's requirements, assumptions, constraints, and so on should be documented in your Knowledge Repository. The Knowledge Repository serves as your plan-of-record, and feeds either creating your Request for Information (RFI), or capturing your solution requirements via a hardware vendor's Sizing Questionnaire (sometimes called a *survey*). From this information, any number of SAP technology partners can fashion a custom solution for you, a process collectively known as *sizing*. From a big-picture perspective, the sizing process is nothing more than a workflow of sorts, as illustrated in Figure 7.1.

Sizing is all about translating business requirements into technical requirements, and then into a solution consisting of various technology components, which in turn are configured or "sized" for a particular set of customer-unique requirements. To ensure an apples-to-apples result at the end of the process, it's advisable to include the folks at SAP AG in this process (as illustrated in Figure 7.1); otherwise, different hardware vendors may interpret the data you provide to them differently. In the real world, though, you should know that many solutions are sized without the benefit of SAP AG's direct input and expertise, as many of SAP's technology partners are quite adept at addressing the entire sizing process themselves. In any

IN THIS CHAPTER

- Overview—The Sizing and Blueprinting Process
- General Sizing Best Practices and Approaches
- Preparing for the SAP Sizing Process
- SAP Hardware, OS, and Database Sizing
- Sizing Considerations for mySAP Components
- Selecting Your SAP Solution Stack Partners
- Executing the SAP Infrastructure Planning Sessions
- Tools and Techniques

case, throughout this chapter, I will walk you through the sizing process to help you understand what is occurring at each phase, how you can be better prepared for the different challenges along the way, and what you can expect after the process is over.

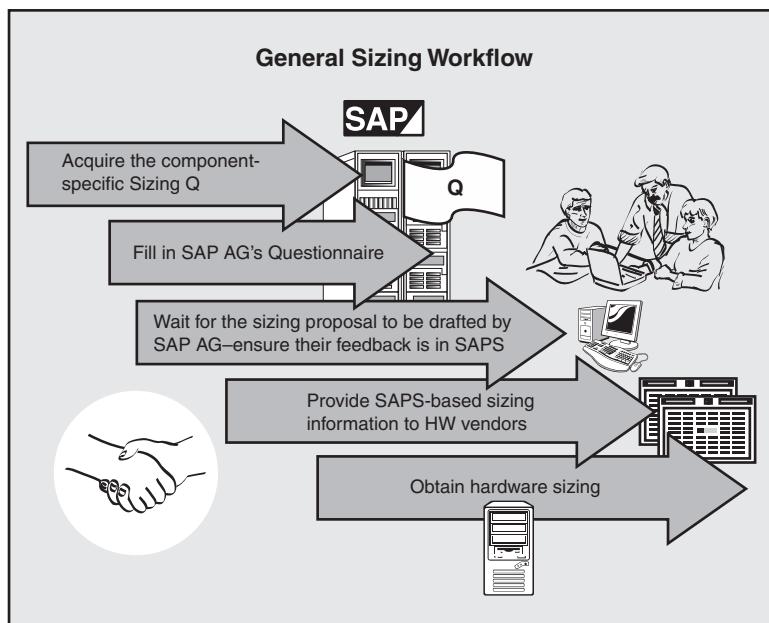


FIGURE 7.1 The sizing process, regardless of specific mySAP.com component, follows a similar workflow.

Sharing Your Vision—The RFI and Sizing Questionnaires

The importance of sizing cannot be overstated. Crafting a system too robustly only wastes money. On the other hand, missing your sizing target directly impacts end-user performance and in its own way also wastes money: Consider thousands of expensive users standing by an incremental second or two for each transaction they execute on the SAP system, and you'll begin to understand how the money accumulates over time. This is why it is so important to nail down what you really require of an SAP system.

Your RFI or completed Sizing Questionnaire helps hardware and software SAP technology partners quantify the level of performance, availability, scalability, and so on that you require, and then map that directly into hardware and software solutions that support your SAP project. We have covered the kind of information that goes into an RFI back in Chapter 3. Sizing Questionnaires seek to gather much of the

same information, though generally focused at helping a technology partner fashion a hardware/software solution:

- Number and type of end users, or simply users, who will eventually log in to the system and use it in support of their day-to-day job requirements.
- Level of activity of these end users, including *think times*, or the time spent in between executing transactions on the system, and other quantifiable work rate or interaction rate characteristics.
- Different functional areas or modules of each mySAP component to be sized.
- Customizing, interface details, and other system requirements unique to your future SAP environment.

Sizing Questionnaires differ from RFIs in both their scope and level of completeness. That is, an RFI often seeks to understand where a vendor can help a company with their SAP project *beyond* sizing and designing an SAP Solution Stack. Sizing Questionnaires, as alluded to earlier, tend to be more focused on the hardware stack or perhaps a piece of the software stack.

Obtaining Vendor's SAP Sizing Questionnaires

Every hardware or software company serious about their relationship with SAP AG maintains an SAP Competency Center (or Competence Center), which is simply an organization tasked with understanding the evolving mySAP product lines coming out of Walldorf, and how SAP business processes can be mapped into a hardware or software solution design. Competency Centers go by many names—Solution Centers, Enterprise Solutions Groups, Centers of Excellence, and so on—but serve the same purpose, to create SAP solution designs based on data gleaned from RFIs, Sizing Questionnaires, and surveys. To help you build bridges to the SAP Competency Centers found within SAP's largest hardware technology partners in North America, I have assembled the following alphabetical list:

- Bull hosts Web access to its SAP Competency Center at <http://www.bull.com/sapsizing>, and offers two business-day turnaround on budgetary sizings.
- Fujitsu-Siemens' SAP Competency Center is at <http://www.fujitsu-siemens.com/sap/quicksizing.html>.
- HP's SAP Solutions Center can be reached via the Web at <http://www.hp.com/go/sap/sizing>. HP offers access to both online and traditional methods of SAP solution sizing, including phone, email, and onsite customer engagements. They offer rapid turnaround on budgetary sizings as well.

- IBM's SAP sizing Web site can be accessed at <http://www6.software.ibm.com/reg/ibmsap/ibmsap-i>, after you register with their site.
- Sun can be contacted for SAP sizing assistance at <http://www.sun.com/software/solutions/third-party/sap/competency/sizing.html>. They prefer that you contact the local Sun sales organization with your sizing figures in hand, and work through their *iSizing for SAP* sizing process, which commences by filling out an online form.
- Unisys's link from SAP's Quick Sizer, as of the writing of this book, is no longer valid. I suggest trying http://www.unisys.com/about_unisys/partners/alphabetical_listing/sap.htm, and from here obtain contact information for one of Unisys's four SAP Competency Centers.

Microsoft and Oracle offer Internet-based access to information regarding their SAP Competency Centers as well. The best place to start in my experience, though, is with SAP's own online sizing resource, the *SAP Quick Sizer*. Although this tool cannot help you size an actual hardware configuration, it is the perfect tool for helping you understand basic hardware needs as well as facilitating an apples-to-apples comparison between different technology partners' approaches to sizing. And links from the Quick Sizer can point you to even more SAP hardware partners than those I pointed out previously, accessible through a simple link from the SAP Quick Sizer's main page as seen in Figure 7.2.

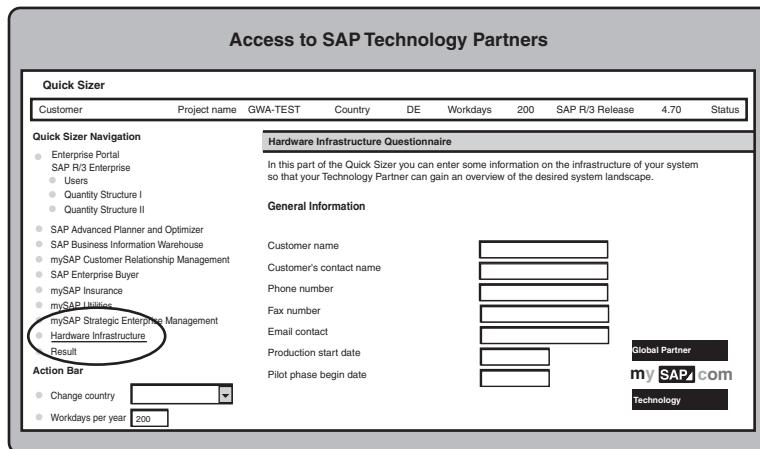


FIGURE 7.2 By using the Hardware Infrastructure link on the SAP Quick Sizer's main page, online access to a large number of SAP hardware technology partners is facilitated.

Fostering Apples-to-Apples Sizings

The key to ensuring that all hardware and software vendors start off on an equal footing lies in the data you provide to them. This data is often provided through an RFI or questionnaire. However, at the end of the day this data can be quite raw or incomplete, and therefore subject to interpretation. Consider the following example—if a potential SAP customer shared with my colleague the fact that they expected 1,000 active users on their SAP R/3 system, but were not clear as to *how* active these users might be, what kind of wait times in between transactions to expect, or which mySAP functional areas to assume, and so on, my colleague could only take a stab at designing a system for them. He would leverage his own experiences and assumptions, document these in the sizing that he delivered to them, and wish them the best of luck.

On the other hand, if I were asked to do the same thing, I would discuss the various types of users with my customer, ask for clarification, and in some way gently “push” them toward giving me better numbers. Like my colleague, I would document my assumptions, craft a solution sizing, and publish it. I would also document my “newly” uncovered sizing data as well, which in effect would give me an advantage over the sizing attempts of my previous colleague—I would better understand my customer’s needs, and would probably create a sizing more reflective of their true requirements.

But what if my customer shared all of this data, and the two sizings now in their hands, with yet another colleague of mine? Given that more time would have probably passed, and my customer’s solution requirements could be even better understood now, my new colleague might be able to pull even more data from the customer, or perhaps bring to light constraints or assumptions true today, but not in the recent past. But his assumptions as to the activity levels or work rates of different end users might differ. All of this iterative and new activity would equate to yet a third solution sizing, very different from the previous sizing exercises.

This scenario I just described illustrates exactly why it is so important to understand as much as you can about your system’s and your end user’s requirements. In this scenario, a lot of time was wasted, and undoubtedly some credibility was lost on the part of the hardware vendor. Imagine if multiple hardware vendors had been engaged, each with their own biases, assumptions, and experience with different computing platforms! The sizings would have looked nothing alike, and the customer would be left frustrated rather than in a position to move to the next step.

One good thing uncovered from this illustration, however, was the fact that the sizing process is very much an iterative process. In other words, even in the best of worlds, new requirements come to light as time passes, or current requirements are better understood. As you see in Figure 7.3, it is up to the customer and SAP technology partners to work through these changes and other variables as efficiently as possible.

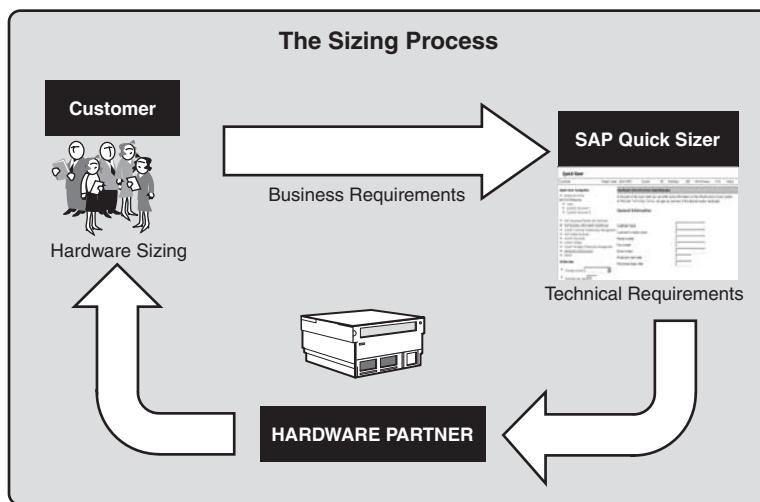


FIGURE 7.3 Sizing an SAP solution is an iterative process, even in the best of worlds.

At this point, SAP's Quick Sizer comes into play and really proves its value. To remove a lot of the variables from SAP solution sizing, and create a near-perfect foundation for apples-to-apples comparisons, I suggest running all of your pertinent solution data through the Quick Sizer. Available to anyone with an SAP Service ID, this online Web-based tool allows you to input user data, mySAP.com components to be sized, transaction loads, and much more. With this information, the Quick Sizer can then calculate general CPU, memory, and disk resource requirements.

At the conclusion of the SAP Quick Sizer process, you are also provided with an idea as to the horsepower required of a hardware solution to host what you just described as your system requirements. This horsepower is measured in something that SAP refers to simply as SAPS, or the *SAP Application Benchmark Performance Standard*. Note that 100 SAPS are equivalent to 2,000 fully processed order line items per hour, or 6,000 dialog steps (screen changes), or 2,400 SAP transactions. Herein lies the great equalizer—by sharing the number of SAPS required of your solution with all of your potential hardware and other SAP technology partners, you have in effect eliminated much of the error-prone sizing processes that serve as a solution's foundation but are subject to broad interpretation. In other words, communicating your mySAP solution requirements in terms of SAPS helps each vendor size a solution for you that is consistent in its requirements. SAPS and other sizing-specific terminology is discussed in more detail next.

Sizing Terminology

Certain terms and vernacular are inherent to the SAP sizing process. Many of these are covered here, though others are covered in context as they are introduced throughout this chapter.

Perhaps the most misunderstood word with regard to SAP sizing is *user*. The term user is a very abstract notion, unless qualified as follows:

- A *named user* is a user with an account in a specific system; named users have User IDs in the system.
- *Logged-on users*, sometimes called *active users*, are a subset of named users. In my experience, the number of logged-on users might equate to only 20–50% of all named users.
- *Concurrent users* are a subset of logged-on or active users, working simultaneously or concurrently in the system. I like to think of this as the number of users who press the Enter key within the same 30 second time period—they are all actively processing or otherwise executing transactions.
- The number of active or open *sessions* consists of the number of SAPGUI or other interface windows open. Many users actually run two or three sessions, and thus look in many respects like 2–3 logged-on users. This can become an important factor in sizing, as you can easily run transactions within different sessions and therefore place quite a bit of load on a system through only a single named user.

Other factors further complicate definitions of the term user. For example, Enterprise Portal or SAP Workplace might be used to facilitate single sign-on to your enterprise mySAP solution, routing users to the proper mySAP component based on user-based roles and transactions to be executed. And mobile, pervasive, and WAP (wireless access protocol) users also generate load in the form of connections to your SAP system. Internet users in general create a load different in weighting than SAPGUI users, too. All of this makes it difficult to quantify the characteristics of a particular set of users.

Thus, it's useful to define a user in terms of *think time*, or the idle time that passes by after a user executes one transaction, but before he executes a second transaction, for example. SAP provides us with three categories that represent typical user activity patterns, measured in terms of the number of processed transactions, or more specifically the number of *dialog steps*, each user executes. A dialog step is roughly equivalent to pressing the Enter key (or equivalent key) on the keyboard, resulting in a screen change:

- A *low user* or occasional user processes on average 10 dialog steps an hour or one every six minutes. Depending on the number of dialog steps in a transaction, this equates to something like one to two transactions per hour. Most users fall into this category in my experience.
- A *medium user* processes an average of 120 dialog steps an hour or one every 30 seconds, executing about 10–15 transactions per hour. This includes accountants, clerks, and so on. Medium users represent the next most popular category of users, behind low users.
- A *high user* (or “power user”) processes an average of 360 dialog steps an hour or one every 10 seconds (something like a full transaction each minute). High-productivity users, like those associated with data entry and telephone sales roles, constitute high users.

Other ways to measure throughput exist as well. As we saw earlier, SAPS are used by SAP’s QuickSizer tool (and incidentally has been part of the SAP sizing process for many years now) to describe the load placed on a system. Hardware vendors sometimes publish numbers like those seen in Figure 7.4 to help customers and other SAP consulting organizations understand how well their hardware platforms scale in terms of SAPS. The fictional matrix shown in Figure 7.4 illustrates the relationship between the number of SAPS supported by a particular platform versus the number and speed of the CPUs supported by the platform.

Sample Hardware Platforms—SAPS Supported														
Sample Server Platforms	CPU Type	SAPS per CPU	Max CPUs	Scaling Factor	SAPS for a system with 1-10 CPUs									
					1	2	3	4	5	6	7	8	9	10
Server 100	6/933	446	2	0.92	892	1,713	0	0	0	0	0	0	0	0
Server 200plus	6/1000	750	2	0.95	1,499	2,924	0	0	0	0	0	0	0	0
Server 400	5/933	379	4	0.92	758	1,455	2,096	2,686	0	0	0	0	0	0
	6/1000	446	4	0.92	892	1,713	2,468	3,163	0	0	0	0	0	0
	5/1200	505	4	0.92	1,414	2,716	2,795	3,582	0	0	0	0	0	0
Server 1200	5/933	379	4	0.92	758	1,455	2,096	2,686	0	0	0	0	0	0
	6/1000	446	4	0.92	892	1,713	2,468	3,163	0	0	0	0	0	0
	5/1200	505	4	0.92	1,617	3,104	4,472	5,731	0	0	0	0	0	0
Server 6300plus	6/1000	750	6	0.95	1,499	2,924	4,277	5,563	6,784	7,944	0	0	0	0
Server 8000	6/1200	505	8	0.91	1,010	1,930	2,766	3,528	4,221	4,851	5,425	5,947	0	0
	7/2000	750	8	0.91	2,999	5,728	6,211	10,471	12,527	14,399	16,102	17,651	0	0
Server 10000plus	7/2000	505	10	0.94	3,031	5,880	8,558	11,076	13,442	15,667	17,758	19,724	21,571	23,308
	7/3000	750	10	0.94	4,498	8,727	12,701	16,437	19,949	23,250	26,354	29,271	32,013	34,590

FIGURE 7.4 Hardware vendors sometimes publish SAP sizing data like this fictional matrix.

NOTE

It's interesting to note that as SAP's products have matured and grown, the need for computing platforms that support a greater number of SAPS has also grown. For example, an R/3 3.1I system a few years ago might have required a platform capable of supporting only 300 SAPS to in turn support 200 active Sales and Distribution end users. With the new features and functionality available on R/3 Enterprise, however, the same 200 users might require a hardware platform capable of supporting 500 SAPS.

For particular implementations of mySAP solutions, it might also be helpful to characterize a system's load in terms of the number of SD, FI, or PP dialog steps—each of these is a functional area with R/3. With this kind of granular data, even the least savvy of SAP technology partners can cobble together something approximating a pretty good SAP solution sizing.

Specifically, use of *SD Dialog Steps* processed per hour is perhaps the most common measurement of throughput outside of SAPS. SAP AG publishes its benchmarks in reference to the total number of *SAP Benchmark Users* and related SD Dialog Steps executed by these benchmark users. Because of this, most of the SAP hardware providers publish their own SAP benchmarks and sizing data in a similar fashion.

CAUTION

People need to take care when discussing SAP Benchmark Users, as there is not a one-to-one correlation between benchmark users and the users found on a typical mySAP system. For starters, because the think time of an SAP benchmark user is 10 seconds, there is actually a three-to-one ratio of benchmark users to actual users. In other words, if a published benchmark indicates that 9,000 benchmark users achieved a certain level of throughput, rest assured that three times this number, or 27,000 medium benchmark users, could be supported by the same system. At the same time, though, consider the fact that you don't want a real productive system to be running at 99% utilization (like a benchmark system), nor will you be executing the same small set of benchmark processes over and over again (and therefore pulling all data from cache rather than truly exercising your disk subsystem). Additionally, the real-world ramifications of background processing and customized code change this relationship to something more like 5-10 benchmark users to one real user. In reality, then, an SAP benchmark is nothing more than a tool you can use to compare one hardware vendor's platform to another's, not a vehicle used for extrapolating real-world performance.

Given the known relationships between SD and other functional areas, any SAP technology partner can easily move between using different throughput measurements and rates, based on what kinds of numbers or sizing formats a customer wants. I suggest sticking with SAPS whenever possible, nonetheless.

There is one last unit of measurement that is worth mentioning, though, known both as *Concurrent SD Users* and *Normalized SD Users* (I've heard this called "Normalized Active Users," too). Regardless of the name, the idea is to convert the users of different functional areas into SD, taking into account the "weight" of that particular functional area. In doing so, a single measurement stick is created. Thus, when an R/3 customer provides a list of users running a mix of MM, SD, FI, PP, and other functional areas, one of the first things that my sizing colleagues and I often do is to convert these figures into normalized SD users. From there, we can easily convert to other numbers and formats, like SAPS. In case you ever need to do the same kind of conversion, I have included a simple Excel-based tool on the Planning CD.

General Sizing Best Practices and Approaches

Even in the best of cases, where much is known about a future solution's peak transaction workload or typical end user's work habits, SAP solution sizing still remains an iterative process, as much art as science. Understanding SAP's architecture can pay big dividends, therefore, when it comes to the sizing process. Of course, to gain more than a cursory understanding of the internal architecture employed by mySAP components requires considerable training, coupled with a number of years of experience. But a basic understanding of the core concepts behind the operation of an SAP system will help smooth out some of the bumps during the sizing process. And this knowledge should help you in maintaining an apples-to-apples sizing comparison between different hardware and other SAP technology partners.

Each mySAP component can be architected to take advantage of something called a *three-tiered client/server architecture*. Many years ago, SAP realized the advantages of separating the application's logic from the database. The three technology layers that came of this—database, application, and front-end client—came to be known as a three-tiered architecture. In breaking each layer out this way, each could be *scaled*, or grown, which at the time was a very different approach from the monolithic mainframe solutions of the day, where growth meant tossing out your current mainframe and lugging in a bigger one.

SAP also architected the three layers such that they could reside on a single physical machine, or could be combined in different ways. The result was a very flexible and—based on the number of SAP deployments—a very successful architecture. Today, every layer, even the database server, which handles all database transactions, can be scaled through products like Oracle's 9i Real Application Clusters.

If we think of the database as the *first* layer in the three-tiered client/server architecture, the application component, called the Application Server, is the *second* layer. It is very common to see anywhere from 2 or 3 to perhaps 10 or 12 application servers

in a single system. In this way, the system's processing power is easily increased as a system's utilization or requirement to host a greater number of users increases and therefore more horsepower is necessary. The runtime element of mySAP components is referred to as the *kernel*, which spawns a number of SAP *work processes*, each serving different functions—work processes are created specifically to support online users, background or batch processes, printing, database updates, and so on.

Another unique element of the application layer is the *Central Instance*, which takes care of handling database locks, interapplication server messaging, and other core housekeeping activities; without the Central Instance, there is no SAP installation. Oftentimes, the Central Instance (CI) actually runs with one of the Application Servers dedicated to servicing end-user transactions. For the most robust configurations, though, SAP allows us to actually relocate the CI to its own physical server. This is so that the very processor-intensive application servers can be granted the resources of an entire server, instead of being forced to share CPU and memory with the CI. In other words, a separate CI server helps ensure the system can respond instantly to the next request without waiting for resources (primarily CPU) to be freed from some non-CI related usage.

The third layer in the three-tiered client/server architecture is the presentation layer, which simply means front-end clients. It is with these front-end clients (desktops, laptops, wireless handheld devices, and so on), that users connect to the SAP system using a Web browser or SAP's own user interface, the SAPGUI. In many cases nowadays, an additional tier, the Internet or a company's intranet, exists, too. This tier actually resides between the application and client tiers, and in effect extends both the application logic and the network of mySAP solutions. Thus, a four-tier solution is born in these cases.

Although my focus thus far has been on three-tier environments, remember that SAP's architecture is flexible and can easily be adopted to support two tiers as well. Simply put, if the database server and application server execute on the same physical server, you have a two-tier system. In this kind of environment, end users connect directly to the central instance, whereas in a three-tier environment, end users connect to a specific application server or pre-established group of servers called a *logon group* (though this connection is still intelligently handled by the CI). And whether you have architected a two-, three-, or four-tiered SAP system, all communication between the different tiers takes place over TCP/IP (with some exceptions in two-tier systems that leverage process-to-process communications, which are outside the scope of this introduction to SAP architecture).

Now, armed with a better understanding of what SAP architecture entails, let's move into the next section where different sizing methodologies are put into practice to architect specific mySAP solutions.

Understanding Different Sizing Methodologies

In addition to a full-fledged top-to-bottom solution stack approach to SAP sizing, a number of other sizing methodologies and approaches are often undertaken by different SAP technology partners. The key to any valid sizing approach is to understand the workload being performed, so that a hardware configuration with the proper number and speed of CPUs, RAM, and disk drives can be assembled. Some sizing approaches are faster than others, though, at the expense of sizing precision. For example, many hardware vendors provide *Budgetary Sizings* based solely on the expected number of active users to be supported by a particular mySAP solution. In this way, a ballpark dollar figure can be gleaned early in an SAP project without requiring all the time and trouble of answering a comprehensive sizing questionnaire.

As I mentioned earlier, SAP AG provides its own rendition of a budgetary sizing by means of its QuickSizer, an online tool most often leveraged for its ability to perform rapid user-based mySAP sizings. Available at <http://service.sap.com/quicksizing>, the QuickSizer can also model mySAP solutions even more accurately through an analysis of transactions and resulting outputs, in the form of customer-provided quantity and structure-related data. For example, business requirements that can be described in terms of the number of expected financial documents, receipts, postings, average line items in a typical order, and so on to be processed or created annually will more accurately help an SAP sizing expert craft a hardware solution than will an online user-based sizing approach.

This brings us to *Transaction-Based Sizing*. As the name implies, this approach seeks to characterize and understand the nature of end-to-end functional transactions being executed as part of a particular mySAP component. In addition to the quantities and structures already mentioned, peak processing hours and peak throughput loads are also factored in, just as they should be in user-based sizing exercises. Great care needs to be taken to avoid underestimating the number of transactions to be performed by a particular system, though—it's easy to shortchange the sizing exercise. In my own experience, I therefore try to do the following:

- Understand what the new SAP system replaces, which can help me understand potentially how many users in various functional areas might be using SAP in the future. Again, though, great care must be taken not to confuse the limited capabilities of a legacy system with a new SAP solution. The SAP solution will generate many more transactions per user, due to its greater capabilities and ties back into other functional areas.
- Define the peak transaction processing requirements, not just what the system will typically be doing day-to-day. In other words, it's important to discover what a particular customer's month-end or quarter-end close looks like from a transaction load perspective, and whether any seasonal peaks exceed even this load. Don't forget to include both online and batch transactions.

- Explicitly state assumptions. If a customer does not understand his batch job requirements, or is unclear as to reporting requirements, I will take a cut at this based on my own experience, and document my assumptions. In this way, if the customer later learns what his exact requirements are, it is a simple matter to refine the sizing document (and therefore avoid accidentally doubling or tripling a load that had been previously extrapolated but not clearly identified).
- Determine transaction types and weighting. Not all transactions are equally “heavy” in the eyes of SAP. Financial transactions, for example, may only consist of four dialog steps whereas SD transactions are five to six times heavier. Thus, not only do I determine the types of transactions that will occur on a system, but I also seek to convert or normalize all transactions into a similar genre (like SD, if I’m working with R/3), by weighting many light transactions into heavier SD transactions, and so on.

Other sizing approaches are quite common, too. A *Delta Sizing* approach, for instance, is quite useful for customers already live on a particular SAP product. The customer’s SAP Basis or adept SAP Operations team can be easily directed through SAP’s Computer Center Management System to identify the load observed real-time and historically in terms of the number of dialog steps processed, so that planned changes to the system (like adding an incremental number of users or transactions) can be intelligently extrapolated.

The final and most demanding sizing process that I am aware of is called something akin to “customer-specific sizing benchmarks” or “customer performance testing” or “proof-of-concept tests.” Regardless of the label, these sizing exercises take much of the assumption and to some extent guesswork out of sizing, replacing them instead with hard facts as to the load that a particular SAP Solution Stack is capable of bearing. Although a Proof-of-Concept, or *POC*, can be time-consuming (not to mention expensive), the resulting peace of mind is compelling. POCs share much in common with stress tests and load tests, which are executed prior to Go-Live to ensure that a production system is indeed capable of meeting performance metrics and other service-level agreements made between the IT department and an enterprise solution’s customers, its end users. For example:

- A POC and a stress test are both focused on testing the performance and scalability of an SAP product.
- In both cases, testing is usually initiated from single-user tests, and then scaled to a larger and larger number of front-end clients that eventually represent what a customer will expect in the real world.
- To perform a POC or stress test, either the actual pre-production system or a system configured identically to it must be installed, configured, and tuned.

- Real-world data, and plenty of it, is required. In the early stages of sizing, this is often the biggest factor in pulling off a successful POC, as good data is hard to come by, much less lots of good data.
- Access to onsite sizing and POC professionals can be a challenge, depending on the solution stack you wish to test.
- The overall expense can seem prohibitive, but as with an insurance policy, you likely prefer to spend a little today to save a lot down the road.

► To learn more about SAP stress testing, see “Introduction—Preparing for Production Stress Testing,” p. 569 in Chapter 16.

Because of these factors, in the end I’ve seen more user-based SAP solution sizing than anything else. Of course, transaction-based sizing and to a lesser extent, proof-of-concept testing will always be popular when risk is the highest. This is especially true for large or otherwise complex mySAP architectures, where transaction-based sizing should represent a minimum requirement of sorts, so as to both accurately and conservatively size your SAP project. And other sizing approaches exist, too, that target specific unknowns. For example, “characterization testing” seeks to test a specific function like batch processing or reporting, to learn how much horsepower needs to be available to meet the required minimum window of time to complete the batch processing or reporting. In those cases where SAP allows a process to be broken down into parallel processes and executed concurrently (called *parallelization*), such characterization is particularly important.

Sizing Tools, Practices, and Assumptions

SAP’s Quick Sizer, along with all hardware vendors’ SAP sizing tools, must make assumptions regarding what you seek in a solution. One of the most important assumptions that you need to verify with your hardware vendor involves how your specific SAP workload is distributed among the servers in your solution. Each vendor and their SAP sizing tools makes assumptions like these:

- The load borne by a system architected for three tiers is often split 33/67 or 25/75, between the database server and the Central Instance combined with all application servers, respectively. Verify these numbers are consistent if you are working with multiple hardware vendors.
- Batch and user/online loads may be distributed to dedicated servers, or shared. Verify how the tool addresses this, if at all.
- When clustering, each node in a cluster can be configured to perform work while running in “normal” non-failover mode. Verify what both the normal and failover workloads look like for each cluster node.

A sizing tool must also make assumptions as to the specific version of a database release, operating system version, and even mySAP release! My advice is to verify with your hardware or software vendor that any specific SAP Solution Stack components you require are indeed addressed by their toolsets and approaches. It makes little sense, for example, for a hardware vendor to use its SAP/UNIX sizer for a specifically requested Windows solution. The same goes for specific versions of databases and mySAP components—each version has different processing, memory, and often even disk requirements. Using a tool incapable of addressing your specific solution stack makes the output derived from that tool suspect.

Similarly, attention needs to be given to the methodology employed to determine how large your SAP database will be in two or three years, and what the growth chart will look like over time. Different database sizing approaches have evolved over the years; verify that your prospective vendors are using the same method, or in some other way guide them toward agreeing on a number that makes sense to everyone. I like to size an SAP database for three years' growth, for example.

Another important assumption has to do with system utilization numbers. Remember earlier in this chapter when I cautioned against misinterpreting the performance obtained by SAP Benchmark Users? When a benchmark is run, the system being tested is usually stressed up to the point where the average response time observed by each user is just under two seconds. Doing so typically pushes CPU utilization to a maximum of 99 or 100%. In the real world, though, when sizing SAP solutions, hardware vendors need to make assumptions as to what kind of utilization thresholds you are comfortable with. These vary depending upon the vendor, but typically resemble the following de facto standards:

- Servers are sized such that the average CPU utilization over time is 65% or so. In other words, the system might spike to 100%, or sit nearly idle occasionally, but generally will hover around the 65% mark.
- Of this 65% utilization, fully half is dedicated to user-based dialog processing, and the other half to a combination of batch processing, printing, interface processing, and reporting.
- The remaining 33% worth of “capacity” remains available to provide capacity to initiate new work with minimal delay which, in turn, results in predictably good response times. This extra capacity also helps to address unforeseen or unplanned future workloads.

Be careful that each of these assumptions is clearly documented in the sizing documents you receive from each hardware vendor. Differences in assumptions can make an enormous difference in the solution proposed by one vendor over another, for instance. I know of one hardware vendor in particular who in the past has deviated from these standards for the express purpose of making their hardware solutions

seem more robust than the competition's: By sizing for the full 100% capacity of a server, their solutions therefore appeared to require less RAM and CPU processing power. This helped them undercut other hardware vendor's proposals when in fact their less-than-customer-focused tactics only left their clients with premature performance problems that eventually had to be addressed before Go-Live.

Best Practices Regarding System Landscape Design

Although the architecture of your SAP system landscape was discussed in Chapter 3, the topic needs to be revisited quickly here. In short, to ensure apples-to-apples sizings, I recommend that you plainly direct each potential hardware vendor to size for identical system landscapes. In other words, do not leave this up to their discretion (unless your goal is to simply see at a high level what kind of unique solution each vendor can craft to solve your business problem). For example, you may want to be explicit about how each vendor should address high availability. It is better to indicate "include clusters for HA and SQL Server log shipping for DR" rather than only stating a 99.9% uptime requirement and allowing each vendor to determine how to address this themselves.

And be clear as to which SAP system landscape components you want to see included in your sizing. A four-system landscape can be interpreted in many different ways—one vendor might make the fourth system training, another configures a technical sandbox, and a third vendor gives you a staging system. The same approach is true for database sizing—clearly indicate where you wish to host copies of your full production size database, and where smaller development or sandbox databases are appropriate.

With regard to the system landscape, you also must be clear about whether a fourth tier is required, and what exactly that entails. And you need to cover landscape deployment options, like using *instance stacking* to install multiple systems on one physical server. Stacking is quite common in the Unix world of SAP (and to a much lesser extent, Windows), where development, test, and training instances might all reside on one very capable server rather than separate servers. Finally, you should help push each vendor toward a consistent standard for sizing the various systems within the system landscape. Specify, for example, where minimal server, disk subsystem, and other hardware components should be employed. Your Test system should be able to support a specific number of users, or a specific percentage of the load to be eventually borne by production. Similarly, your development system should be configured robustly enough to keep your development team from walking off the job—give your hardware vendors a specific target, like the ability to support 20 high-activity developers, and this will help you to continue to support an apples-to-apples sizing comparison.

Preparing for the SAP Sizing Process

After you have obtained an SAP hardware or software partner's Sizing Questionnaire, the real work of completing the questionnaire begins. Do not be tempted to shortcut this process! The questions in each questionnaire are important, and need to be answered. These answers will in turn feed a vendor's specific custom SAP sizing tools, methodologies, and models. Combined with each vendor's judgment and experience, they will then be able to determine the specific configuration needed for your system should you choose to run it on their platform.

I like to have the project's SAP solution architect take ownership of completing all questionnaires. The SA is also well-equipped to actually answer most of the questions, so this approach makes sense, with the following exceptions:

- Disk sizing questions that derive their questions from functional or transaction-based requirements can actually be better addressed by experienced end users (super users), especially when they are paired with experienced SAP basis or sizing experts.
- The SAP project manager should provide an SAP resource trained and experienced in sizing. SAP often refers to this skillset as *capacity planning*. Regardless of the label, this role is critical from a quality assurance point of view.
- A member of each hardware and software vendor's technical sales staff should also be involved, responsible in a liaison-like manner for ensuring that the customer-vendor SAP sizing process does not stall. Because of the iterative nature of good sizing, the process is especially susceptible to being derailed over time; direct involvement of each vendor helps avoid this.

After the questionnaire, RFI, and/or output data from the SAP QuickSizer is internally reviewed, refined, and shared with your prospective vendors, it's a good idea to host a one- or two-hour conference call with everyone. I have seen calls hosted by a customer where all potential vendors were on the call, and at the other extreme I have played a part in one-on-one calls, too. Either way is effective, but for the sake of apples-to-apples comparisons, I prefer that a single conference call be used to host everyone—this way, everyone hears the same things. I also suggest that the senior solution architect as well as both the client and SAP project managers host the call, as then most any question can be answered "real-time."

The Pre-Sizing Conference Call in the Real World

Through this initial pre-sizing conference call, before any real hardware sizing begins, you can accomplish a lot. First of all, you can help set the stage for your particular mySAP project by sharing your vision, your business drivers, and your timelines. You also have an opportunity to outline your priorities, discuss your

biases, identify the skillsets and experience you already have in-house that will prove relevant to managing and supporting SAP, and so on. All of this will serve to greatly improve your chances of obtaining SAP solution sizings that are done as close to “right” as possible, the first time.

Remember, of course, that the sizing process is still iterative even in the best of worlds. However, the conference call approach just discussed will minimize the number of unnecessary and time-consuming SAP solution resizings that you would otherwise be forced to review.

Given all of the sizing conference calls and onsite meetings that I have attended personally, I thought it would make sense and be in your best interests to share the kinds of questions and data that I most often appreciated—my solution sizing colleagues with whom you will work will certainly appreciate much of the same:

- As I mentioned previously, take this time to set the stage for your particular mySAP project in terms of Solution Vision, SAP components to be implemented, business drivers, project timelines, key milestones, and other project-specific data.
- Determine whether this new SAP component to be sized is being integrated with an existing SAP landscape, replacing a current system, or simply refreshing a current system. If vendors have little direction in this regard, then the resulting sizings will probably be so different as to be incomparable.
- Review the numbers you have shared through your RFI, Sizing Questionnaires, or QuickSizer output, and make sure that everyone understands your definition of what the term “user” means to you.
- Clearly identify your priorities in regard to the following—total cost of ownership, performance, availability, scalability, and manageability. Giving your hardware vendors this kind of insight helps to avoid huge surprises when you finally receive their version of what you need in terms of an SAP solution sizing.
- Discuss your system architecture and other technology biases, including whether you prefer a central system approach to a more distributed architecture, whether you are biased for or against a particular operating system or database product, and so on. You should also share what hardware, OS, and database platforms your current enterprise software solutions depend upon.
- Along these same lines, be sure to share the skillsets and experience you already have in-house with regard to various hardware, OS, and database platforms, and how likely you are to change or accept something new.

- Discuss your views and thoughts with regard to risk: Are you willing to try something brand new, or reasonably new, if the potential rewards are great? Do you tend to favor a conservative approach to solution sizing, or something perhaps more innovative or leading-edge?
- Identify which SAP system landscape components must absolutely be sized (like development, test, and production), and which might be “in the air” (like a business sandbox, technical sandbox, training system, and so on). Include whether you prefer the same platforms and components across these systems (a high level of standardization), or whether initial budget constraints require a different approach. The more details the better.
- Similarly, address the sizing of each database for each system landscape component.
- If scalability was not previously covered in detail, indicate something as to the scalability that the system must provide. For example, does your company intend to grow or shrink significantly through acquisitions, mergers, or divestitures over the next 12–36 months?
- In terms of accessibility and front-end desktop or laptop requirements, will Internet access be required? What about new desktops or other client devices?
- If certain vendors have already been selected, I think it makes sense to share this information as well. For example, as a hardware vendor with excellent relationships with a number of the “Big 4,” I like to hear about whether an SAP consulting organization has already been selected to do the functional work of implementation. Similarly, it’s helpful to understand which systems integrators, if any, have already been selected, or whether a certain database or operating system vendor has already been given the nod.
- You also need to discuss any plans for stress-testing, and any thoughts you might have on proof-of-concept exercises. Every now and then, at little to no cost, you can leverage an SAP technology partner to actually “prove” that his solution operates as advertised. This is more often the case with new SAP Solution Stack alternatives that a particular technology partner is anxious to “get into production” for the sake of selling more of the same solutions down the road. In example, I’ve been involved with POCs related to proving the scalability of new hardware platforms and different database products.
- Open up the call at this point for questions. Usually, questions involve clarifying why or how you completed the questions in various SAP sizing questionnaires, or clarification of data provided in your RFI.
- I believe that it makes sense to end the call with a synopsis of everything that has been discussed, and a specific date by which you expect all solution sizings

in your inbox, along with reiterating what you perceive to be the greatest risk in your SAP project plan. This latter point serves to remind all of the participants on the call how important your SAP project is to your business, and therefore should underscore the importance that each vendor places in his solution sizing efforts. Follow up these words with a written one to two page document shared via email.

With a solid sizing foundation produced, you can now let your prospective hardware and software partners get to work crafting solutions that not only meet your business requirements, but hopefully also reflect each partner's unique hardware capabilities and competitive advantages.

Building Your Sizing Evaluation Team

With sizings in process, it's time to get started assembling an official Sizing Evaluation Team (SET). Up to this point, the solution architect and some of his senior team members have played a role in working to develop, document, and otherwise publish your system requirements. Your Knowledge Repository has served its purpose, too, and will continue to grow in value as the solution sizings start rolling in. Additional SAP Technical Support Organization (SAP TSO) team members need to be pulled in at this point, however, to build a virtual team or subteam that includes the following staff:

- The senior solution architect heads this team. Other solution architects may be needed to help evaluate solution sizings and work with different technology partners, however. This is because a large SAP project covering four or five mySAP components, shared with four or five different SAP technology partners, can add up to a whole lot of reading for a single SA to do.
- Technology SMEs (Subject Matter Experts) in each solution stack layer must be represented on the team, including data center, hardware, network, OS, database, and of course mySAP specialists. This often includes many preselected technology partners (SAP and perhaps hardware and database experts). Even so, do your best to avoid obviously biased team members; an open mind is important at this stage in the project. Together, the senior solution architect, any other solution architects, and the combined SMEs make up the core of the SET team.
- Current systems administrators, systems management, and computer operations organizations are represented. This is more for their benefit than the core SET team, but helps to build excellent relationship bridges as well.
- Business staff (from key functional areas) must be represented, to include them in the sizing review process and therefore develop critical buy-in. This avoids "I told you so" or "I didn't know" later in the project, if something goes wrong.

- A documentation specialist (DS) will update your Knowledge Repository and play a role in developing or customizing evaluation templates or similar approaches, if, for example, the number of technology partners makes this necessary.

The goal of the SET team is simple: to review each solution sizing, and evaluate the responses in terms of each vendor's ability to meet your needs. The team needs to keep the various project managers up to speed about this review process. None of this is trivial, though, as you see in the next section.

The Sizing Review Process

With each vendor's sizing attempts in front of us, it is now time to review each solution document in terms of the following:

- Type of sizing—Is the sizing little more than a parts list, or a full-fledged solution architecture document, or something in between?
- Completeness—Does the sizing take into account the entire mySAP system landscape discussed in the pre-sizing conference call, or identified in the questionnaire?
- Reflection of the Sizing Questionnaire or RFI data provided—Does the sizing document clearly restate where it obtained its assumptions, boundary conditions, system requirements, biases, and so on?
- Accurate load factors—Are the correct named, online, or concurrent user counts reflected in the sizing, or is the proper transaction load noted? What about reporting and batch loads, or the potential impact due to expected heavy customizing?
- HA/DR—Is there a section in the sizing that addresses high availability and/or disaster recovery, as requested? Further, is the approach hardware-, OS-, or database-specific?
- Core infrastructure—Is there a section in the sizing that addresses core infrastructure requirements, including network, rack, special servers, systems management options, and more?
- Disk details—Does the sizing go into detail with regard to the disk subsystem configuration?
- Backup/Restore—Does the sizing address technology used to actually back up the various database structures and other file partitions that support your mySAP solution as architected by the vendor?

This sizing review process is by no means exhaustive; thoroughly evaluating each sizing response is discussed shortly, but only after we work through the next few sections in this chapter. Rather, the goal at this stage in the sizing evaluation process is simply to weed out or at minimum identify any solution stack vendors that simply failed to address your requirements.

The Documentation Specialist (DS) has some work to do, then. An enterprising DS will not only insert all sizings and supporting data into the Knowledge Repository, but at the direction of the SA he will also begin identifying where and how solutions differ from one another. To do this effectively, a matrix is useful (the “TCO Solution Stack Considerations” spreadsheet discussed in Chapter 5 and found on the Planning CD can provide a useful starting point). For example, all assumptions, boundary conditions, and other constraints that are documented by each solution vendor need to be recorded between the individual sizings. Basic hardware configurations need to be verified as well, like the number and speed of CPUs, RAM, and disk drives in each server, the number of servers in each SAP system, the size and scope of each SAP system landscape, and so on. Further, missing components need to be noted, like the absence of Requisite BugsEye servers in mySAP SRM solutions, or overlooking the need for ITS servers. All of these factors and more influence the configuration, and therefore impact both the resulting parts lists and the price of each competing solution.

With the general review behind us, we can now begin to focus on evaluating each sizing with a critical eye when it comes to different layers in the SAP Solution Stack.

SAP Hardware, OS, and Database Sizing

In the next few pages, we will take a look at sizing the SAP Solution Stack from an exception perspective. That is, I will only drill down into areas or considerations that are either critical or unique to particular components of the solution. For more complete end-to-end sizing guidelines, on the other hand, refer to the sizing documents found on the Planning CD.

Common Server Considerations for mySAP Sizing

Server sizing is focused on determining not only how many servers might be required to host a specific number of end users or batch processes, but also on how each server is configured in terms of processors, RAM, and local disks. Server sizing is also impacted by your unique high-availability and disaster recovery requirements; as discussed in Chapter 6, certain HA/DR offerings are only supported by specific hardware, operating system, and database combinations.

Different form factors and types of servers might also best serve your particular needs. For example, many of my customers purchase only rack-mountable servers,

whereas others might prefer tower models. In the same way, some prefer dense form factors (slim-line servers that only consume a few inches of vertical space), but others prefer larger server footprints with correspondingly greater capabilities (like room for more disk drives, processors, PCI and other I/O slots, and so on). If you let your potential hardware partners know your preferences up front, you'll save a lot of time when it finally comes down to identifying part numbers and putting together pricing information.

Another obvious server sizing consideration involves the operating systems supported by the platform. If your OS standard is Linux, for example, you need to ensure that your hardware partners know this, so that they size your mySAP solution with Linux-compatible server platforms.

Finally, scalability of the platform needs to be taken into account. If your SAP system's database server requires four processors to address your estimated peak workload, you probably do *not* want to lock yourself into a four-processor platform; a larger eight-CPU-capable system configured for only four CPUs provides you with in-the-box vertical scalability if you later determine you need more processing power. The same philosophy should be applied to RAM, I/O slots, and so on, unless you are either quite certain of your needs or willing to risk being wrong. Beyond scalability of your database platform, you also need to share your sizing philosophy with regard to application servers. Do you prefer in-the-box scalability, or is your application server strategy centered around adding additional fully-configured servers? Like determining your OS and platform preferences, your scalability requirements should have already been fleshed out by this time, and shared with each hardware vendor.

- ▶ To read more about different core strategies for sizing, see “A Closer Look at the SAP Solution Stack,” **p. 51** in Chapter 2, and “Driving Scalability into Your System Landscape,” **p. 74** in Chapter 3.

Disk RAID Configurations and Features

Although a basic discussion of RAID was covered in Chapter 6, a discussion from the perspective of SAP sizing is in order now. The following list identifies different RAID implementations and the configuration options, advantages, and features of each as it relates to sizing:

- RAID 1, or mirroring. For a desired amount of disk space, you must purchase and install twice as much disk space; 400GB of usable space requires 800GB in raw disk space. Thus RAID 1 configurations can get expensive not only in regard to the number of physical disk drives required, but also in terms of the number of drive shelves and even storage cabinets needed to house the drives. In some cases, the number of array controllers and requisite cabling must be increased as well.

- RAID 1+0, or 0+1, or 10, or striping/mirroring. Like RAID 1 mirroring, half of the configured drives are consumed to protect the data on the other half. Thus, the same cost structure applies here. The benefit is maximum performance, however; both writes and reads are performed faster in RAID 1+0 configurations than in other implementations.
- RAID 5, or striping data and parity. This is usually the least expensive method of achieving high availability within a disk subsystem. Depending upon the implementation, a RAID 5 configuration will only lose a fraction of otherwise usable space to parity. A couple of things are very important, however. First of all, RAID 5 implementations will never realize the raw performance achievable by other RAID solutions. Second, to obtain even moderate performance, a minimum number of drives (often referred to as *spindles*) must be configured. Last of all, a RAID 5 configuration might be limited in terms of the number of drives that can be configured versus the high availability obtained; for a system with six drive shelves, a RAID 5 configuration is at risk (as is any RAID configuration) if two drives are housed in the same shelf. Thus, the value derived from the benefit of losing only a fraction of disk space to parity might be diluted a bit, depending upon the disk subsystem vendor's gear and how it implements no-single-point-of-failure disk solutions.
- Other RAID levels, like RAID 3/5 and RAID 4. These are usually best compared to RAID 5, where a certain quantity of raw disk space is lost to maintain parity data. Generally, the amount of required parity data is greater in these implementations than in RAID 5, however (or overall performance is worse), which is one reason why they tend to be less common today.

With the basics behind us, let's turn our attention to some of the disk subsystems and drives available on the market today, and how they can impact our SAP sizing.

Commonly Deployed Disk Subsystems for SAP

Various disk subsystems are available on the market today, from popular SCSI-based solutions, to those leveraging fibre-channel and other connectivity fabrics. From a sizing perspective, I have found that the most important consideration is scalability, especially in terms of in-place upgrades. Disk subsystems are expensive, and replacing your system with a new one is even more expensive. Money spent up front to address not only scalability but also the ability to upgrade drives, controllers, and so on can make incremental upgrades easier, less expensive, and helpful in extending the life of your solution. With this in mind, take care to buy a disk subsystem that is not at the end of its useful life cycle. And be sure to invest in a platform that has a historical and well-documented upgrade path—a system that allows you to replace all of your disk drives with larger and faster drives is a good example. Similarly, a disk subsystem that supports upgrades to its array controllers and cache makes financial sense, too.

Like disk subsystems, a variety of disk drives are available today, ranging in capacity, form factor, and speed. Consider the following:

- Drives that operate at 10,000RPM can boost I/O performance between 5–30% more than their 7,200RPM counterparts, depending on their application in a computing environment. Typically, write-intensive volumes such as database transaction logs, temporary/sort areas, and very active, or *hot*, tables and indexes, benefit the most.
- Further, 15,000RPM hard disk drives access data 26% faster than their 10,000RPM counterparts. These 15K drives deliver a theoretical 50% improvement in Input/Output Operations Per Second (IOPS) over 10K drives, and 108% over 7200RPM drives (though in reality the realized performance improvement is much smaller). Servers with multiple drives handling large numbers of transactions benefit most from this increased I/O performance.
- The largest drives on the market today can actually hold an entire SAP database, though actually doing so is *never* recommended. Why? Because individual disk drives can still move only five to seven sustained MB per second factoring in all latencies. Thus, as always, it's necessary to configure multiple drives rather than fewer.
- To save money, I typically recommend implementing something smaller than the largest drives available, though in today's economy, the cost difference between different sizes continues to decrease. A good example today is the use of 18GB drives in database design; with 36 and in some cases 72GB drives costing nearly the same, it simply makes no sense to invest in smaller drives. A year from now, the same argument will be made for 36GB drives, too, as 72 and 145GB drives continue to become more mainstream and subsequently fall in price. If you focus on spindle count rather than pure capacity, you'll be in good shape when it comes to performance.
- Disk drive form factors impact sizing in that the largest and newest drives tend to be physically larger than their more mature counterparts. So, to take advantage of the largest drives may require sacrificing in terms of the number of drives that can be physically installed in a particular disk subsystem. My general recommendation is to ensure that your disk subsystem can handle different drive sizes (for future upgrades or other needs), and then size for the number of spindles needed (to meet your performance goals) rather than installing the largest drives. And if a particular drive size (like 36GB) happens to be available in two form factors, push for the smaller form factor to make more empty drive bays available later when you need more disk space.

One other general area is important, too, that of using disk-based storage to house online copies of your database. One client of mine has effectively quadrupled the

amount of disk space required to house their database volumes simply because they maintain a number of “aged” offline copies on disk, too. Another client eats up disk space in a similar method, by maintaining a series of client copies. And many more refresh their supporting landscape systems with copies of the production database on a regular basis, consuming disk space throughout the system landscape. Don’t forget to factor in these disk copies, if that is part of your DR, client, or other strategy.

Storage Virtualization—The Latest Paradigm in Enterprise Computing for SAP

As I detailed in Chapter 6, the use of disk subsystems that support Storage Virtualization, or SV, is growing. SV allows a greater number of spindles to be configured than competing technologies, and in doing so creates a more flexible foundation to address both growth and changing subsystem needs. Storage virtualization improves storage utilization by minimizing wasted space associated with configuring lots of RAID 1 volumes, and by reducing the disk space required to maintain parity data in RAID 5 implementations. It also supports dynamic storage allocation and reallocation, which again minimizes wasted space by letting you “move” or allocate storage from static drive volumes to those volumes that are growing. Beyond typical storage subsystems, design considerations for a virtual storage array include the following:

- There is a need to design and size a new SAN abstraction layer, referred to as a *group* by a number of virtual storage vendors in the market today. This term refers to the collection of physical disk drives that are pooled together at a hardware level, on top of which storage LUNs are created. It is the LUNs and not the groups that are assigned RAID levels, like 1+0 or 5. The groups merely provide a way of providing a specific number of underlying disk spindles to a database or other set of files, or segregating different types of I/O from one another. Especially in the latter case, it’s common to segregate OLTP transaction loads (short discrete transactions typical of R/3, CRM, and SRM products) from OLAP or reporting data loads (like the transactions and queries inherent to BW and SEM).
- The argument for RAID 1+0 versus RAID 5 is beginning to fall apart, as the number of drives available to a virtual RAID 5 LUN has accelerated throughput and I/O performance levels to those nearly equaling that achieved by RAID 1+0 configurations. This is dependent upon a number of factors, however, including the hardware vendor’s implementation of both SV and RAID 5.
- Given the striping of data that automatically occurs across a group of drives, creating simple RAID 1 mirror pairs is no longer needed. The minimum number of disk drives required to create a virtual array is six or eight, so the LUN created on top of these drives benefits from both the performance and

availability inherent to multiple spindles. The real challenge then becomes laying out disk partitions, such that the performance requirements that need to be achieved by virtue of the role that the partition plays can indeed be achieved without losing a lot of disk capacity. For example, no one wants to place a 20GB SQL Transaction Log file on a RAID 0+1 (sometimes called vRAID 1) virtual array stripe of 36GB drives—the 108GB of usable space would be virtually wasted. So to better use the space, multiple OS partitions might be placed on it, like SQL Server's TempDB, SQL Server executables, and so on.

- Similarly, because LUNs are no longer locked in by physical disk size boundaries, more efficient use of expensive disk resources can be realized. For example, in a traditional SAN configured today, it would not be uncommon to consume a pair of 18 or 36GB drives to house something as small as an SAP executables volume, or perhaps a MSCS quorum disk; most of this space would simply sit idly by, never to be used. And it could not be effectively shared with other OS partitions because of the 5–7MB throughput limitations per spindle discussed previously. In a virtual storage solution, though, storage can be easily customized to provide only what is needed—the spindle counts underneath the LUNs allow this flexibility. The resulting savings in disk space can therefore be absolutely huge for a typical mySAP implementation of three or four components, each with four-system landscapes.
- Virtual storage arrays introduce new high-speed SAN fabrics (2 Gigabit and faster), which need to be taken into consideration when adding a virtual storage array to an existing SAN; they interoperate, but at the slower traditional SAN speeds.
- Spare capacity (the SV version of “hot spare drives”) becomes more critical as data is spread across more and more drives. Why? Because the Mean Time Between Failure (MTBF) of a large group of drives makes it more likely that a drive failure will occur within one of your LUNs in the next two or three years. MTBF is described as an absolute value, of course, but in my experience it’s more like an “average.” For instance, if the MTBF of a particular drive is one million hours, then half of this group will succumb to a failure before it ever reaches one million hours of service time. In my eyes, the effective MTBF for a group of 50 drives therefore drops to only 1/25th of the rated number, or 40,000 hours in this case. Thus, the likelihood of suffering from drive failures sooner than later is that much greater, and the need for spare drives becomes that much more imperative.

Keeping the previously noted design considerations in mind, sizing a virtual array is quite similar to sizing any other disk subsystem for SAP, with the following exceptions:

- The highest levels of disk performance are achievable, though at a price. Thus, a disk-subsystem-focused delta TCO analysis may be required in cases where the need for maximum disk throughput is not the primary sizing consideration.
- The greatest scalability is achievable too, again with the caveat that the cost per GB may be higher than in less scalable systems. Interestingly, though, as a virtual array grows, its storage utilization actually improves, paying off big dividends in terms of lower and lower costs per GB.
- Virtual storage arrays are still relatively immature and untested in the area of SAP production environments; I have only seen four implemented to date (though many more are in the works as this book goes to press). So for the most risk-averse SAP implementations, this lack of maturity constitutes a sizing factor.

Other factors can come into play, too. The fact that the VA disk controllers are two to six times faster than their predecessors and fibre-channel disks are faster than their SCSI counterparts is an obvious factor. The cost of storage administration, serviceability, multivendor hardware and operating system support, and other positive traits will push the adoption of storage virtualization across the enterprise, too. Similarly, difficulty in accessing virtual storage specialists, and the paradigm change in managing data will hinder this adoption. But I believe the most compelling reason we will see virtual storage take off in mySAP environments is simple—awesome performance. Review the “Virtual Arrays vs the Competition” PowerPoint on the Planning CD for compelling data illustrating my own findings and observations.

Operating System Factors

The capabilities of one operating system over another often come into play when sizing an SAP solution. At the most basic level, sizing a solution based on a preferred OS is quite common—I've often been told by my customers and prospects that a particular OS is their standard, and therefore needs to be taken into consideration. Beyond the general OS, though, lie differences in different versions of the same basic OS, be it Windows, Linux, S/390, or any number of Unix flavors. The following other factors can influence an SAP sizing:

- Support for new mySAP products is first provided on Windows-based (and to a lesser extent, Linux-based) platforms; other OS platform support is provided as the product matures.
- Memory, processing power, or other hardware-related factors may dictate a particular operating system. A classic example is the fact that you have to step up to Windows 2000 Advanced Server to *really* take advantage of more than 2GB of RAM, regardless of how much physical RAM might actually be housed in a server.

- HA and DR considerations will drive which OS you select for your SAP system. Even if you are clearly in one OS vendor's camp, your need for a certain level of high availability may force you to purchase a different version of an OS than you otherwise would. Case in point—Microsoft's Windows 2000 platform again, where server clustering is only supported on the Advanced Server and Data Center Server editions of the OS.
- Obtaining OS expertise may be a factor—as with any solution component within the stack, access to experienced and reasonably priced technical support staff can sway your decision to go with a particular operating system. For example, although Windows Server 2003 (WS2003) was recently released, it will be some time before we see it underpinning productive SAP installations; it simply takes time for a new product to mature, reflecting both people and product development learning curves.

Database Selection and Sizing Factors for SAP

Selecting a particular SAP-supported database often comes down to two things—taking advantage of in-house experience with a particular RDBMS, and seeking to reduce the cost of acquiring or renewing database licenses for your SAP implementation. As I mention later in this book, one of the biggest total cost of ownership factors for an SAP solution revolves around the database component. Other sizing-related factors include

- The OS and database (DB) are often tied together; SQL Server only runs on a few select Microsoft OS platforms, for example.
- High Availability and Disaster Recovery options are often tied to a particular database.
- Some database releases are inherently faster or more capable of performing a particular type of operation than other database releases.
- Database acquisition pricing varies widely, from free (SAPDB, for example) to quite expensive, and everything in between.
- The widespread adoption of certain database technologies with regard to mySAP implementations has varied as well, though I continue to see Oracle's and Microsoft's (and to a lesser extent, IBM's) database products dominate this market.

Regardless of the RDBMS deployed, understanding how quickly your database will grow is the biggest factor of all. Typically, I try to understand my client's three-year plan in this regard and size the disk subsystem with room to spare.

Another database consideration involves whether you plan to employ SAP's MCOD feature. MCOD (Multiple Components, One Database) allows you to install several

mySAP components on one physical database. For example, you can combine a couple of R/3 4.6C instances, mySAP CRM, and SAP Workplace all in one database. One sizing key is to combine only components that are similar in nature—though other combinations are supported in some cases, OLAP components should only coexist with other OLAP components (like SEM with BW), and OLTP components should only be paired with other OLTP components. This is because of the nature of these disparate systems; good performance would be more difficult to achieve otherwise.

The key sizing factor is the database server, though. A conservative approach to sizing includes sizing the individual components separately in terms of disk space, CPU processing power, and RAM required. To calculate total disk space required, simply add up the requirements of each component, and subtract 10% (which according to SAP is the typical disk space savings realized by MCOD deployments). To calculate CPU requirements, be sure that you capture the number of SAPS that characterizes the load of each system, and then add these together. Do the same for memory needs, and present your combined SAPS, disk, and RAM requirements to your hardware vendor so that an appropriate server and disk subsystem platform can be architected.

Finally, SAP AG requires that you combine only production systems with production systems, test systems with other test systems, and so on. Do not mix landscapes, as it is not supported by SAP AG.

Sizing Considerations for mySAP Components

With the bulk of the solution stack behind us, we can now turn our attention to sizing specific mySAP components. My goal in the mySAP sections that follow is to identify critical or differentiating factors, such as the need to answer component-specific sizing questions or understand component-specific configuration options. And in all cases, the need to identify the particular release of a mySAP.com component is assumed; minimum hardware and software requirements can vary widely between different releases of the same SAP product line, and therefore greatly impact sizing. Thus, it is not enough to say that you plan to implement SAP BW or R/3—BW 3.0B or R/3 Enterprise 4.7 must be specified, for example.

Another general requirement is to understand and characterize either the active high-water user load or peak transaction rate of a particular system. From these numbers, concurrent user numbers can be calculated (or extrapolated or assumed, worst case) and the general impact that these numbers make on the system can be taken into consideration.

With an understanding of the basic mySAP architecture illustrated in Figure 7.5, let's take a look at some of the most popular mySAP solutions to understand their unique sizing challenges.

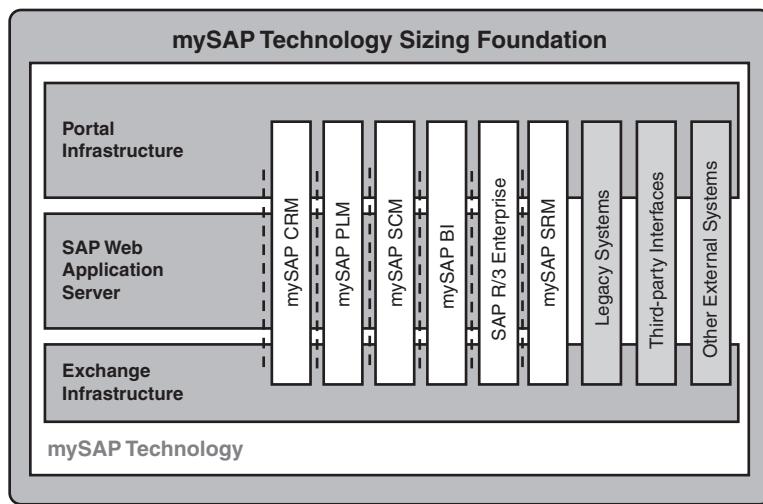


FIGURE 7.5 mySAP technology underpins various mySAP.com components and solutions.

Understanding the “New” Basis Layer—Web AS

As the underlying infrastructure for all new mySAP solutions, and the enabling component of SAP’s Exchange Infrastructure (XI) and Enterprise Portal, accurately sizing the SAP Web Application Server (Web AS) is critical indeed. Web AS is a hybrid application server, able to satisfy both J2EE (Java) and ABAP (SAP’s legacy native development language) needs with the advent of version 6.30. Additionally, it handles functions formerly handled by SAP’s Internet Transaction Servers—load balancing, DB connection pooling, serialization of update processing, and server-side data caching, all of which impact sizing. And Web AS leverages XML instead of HTML, resulting in a different load and different sizing criteria from those associated with ITS.

Web AS can be distributed such that the SAP Web dispatcher process resides on a server in your company’s DMZ, protected by a firewall both in front of and behind it. Meanwhile, the core Web AS functionality can be housed in your data center, and can be further isolated by means of another firewall between it and your core SAP back-end systems. This is very much like the ITS architecture promoted by SAP for its Internet Transaction Servers, where the Web and application components were distributed and surrounded by firewalls as described here.

The SAP Exchange Infrastructure

The SAP Exchange Infrastructure (XI) is an important new integration technology that can be used alone or in conjunction with the newest and future mySAP solutions. XI serves a number of purposes, including

- Integration Repository, used to manage components, interfaces, and mappings
- Integration Directory, for managing your mySAP system landscape, cross-landscape services, routing rules, and executable mappings
- Proxy Generation and Framework, supporting both ABAP and Java proxies from XML-based interfaces
- Integration Engine/Server, offering a runtime environment that supports routing and mapping

As seen in Figure 7.6, the latest version of XI takes these capabilities to the next level, providing a view into collaborative cross-mySAP business processes. For example, business processes that cross multiple R/3 systems, your CRM system, and your APO system can be tied together via synchronous and asynchronous messaging, and visually depicted and managed through XI. This allows heterogeneous system landscapes to be deployed, including SAP and third-party enterprise applications like Baan, Broadvision, JDE World Software, Oracle, PeopleSoft, Siebel, and others.

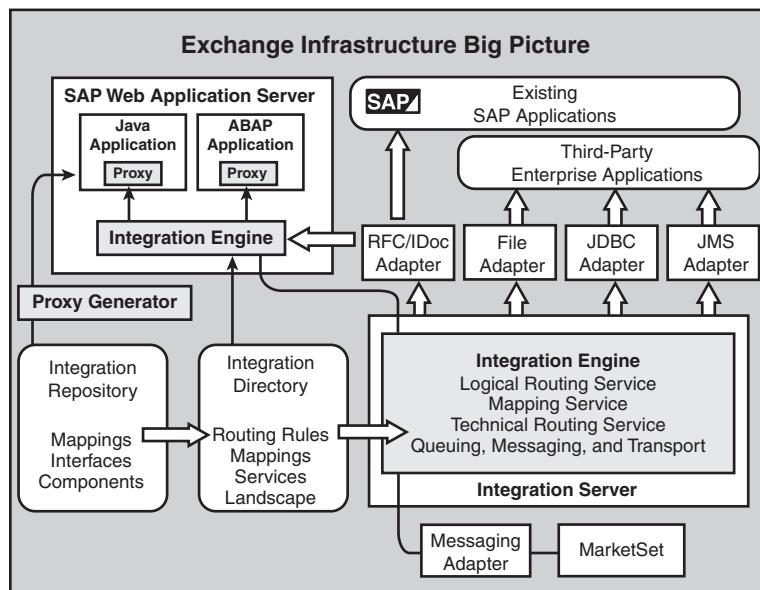


FIGURE 7.6 The SAP Exchange Infrastructure ties into Web AS and all forthcoming mySAP solutions.

In terms of sizing considerations, the number and size of the messages that the XI's Integration Server needs to process is paramount. SAP AG has been gracious enough to publish a sizing model that takes into account the following:

- Disk sizing is predicated upon the assumption that all messages are held in a compressed format for a day, and then either archived or deleted. Further, compressed data enjoys no greater than a 65% compression rate.
- The average size of a message is assumed to be 31KB (which not coincidentally equals the typical size of an iDoc with four line items). This can be easily changed to account for different average sizes however.
- Data held in Unicode format consumes twice the amount of disk space as non-Unicode formats.
- In terms of message traffic, only systems that are directly connected to the XI are factored into the sizing model; other systems, like those tunneling messages through proxy servers, are not taken into account.
- Routing is content-based.
- Average CPU utilization is not to exceed 65%, which is consistent with SAP's general sizing methodology.

As XI matures and additional sizing data comes to light, be sure to check with your hardware vendor's SAP Competency Center or SAP's <http://service.sap.com/sizing> Web site for updated metrics and sizing model considerations.

Enterprise Portal Sizing Rules of Thumb

Sizing Enterprise Portal involves determining CPU and RAM requirements for both the EP server and the Unification Server, which consists of a Web server and a database repository that allows data to be shared and mapped between SAP products. It is assumed that 20% of all EP hits will access the Unification server. Important sizing questions that need to be answered include

- Parallel logons, which is the total number of users logging into the Portal during the peak hour of operation (normally from 8:00 a.m. to 9:00 a.m., or from 2:00 p.m. to 3:00 p.m.).
- Concurrent users, which for EP is expanded to mean the total number of users *active* in the system in the same hour. The SAP QuickSizer assumes that 60% of these users have a think time of 600 seconds, 34% have a think time of 180 seconds, and 6% have a think time of 30 seconds—note that this is quite a different working profile than that observed in SAP's other mySAP components.
- Content Management hits measured as a percentage, or the number of hits per 100 that access documents stored in the Portal.

SAP's QuickSizer also assumes that each Portal page contains four iViews. This number along with information as to how dynamic the portal data is, the number of Drag 'n Relate operations, and the number of queries on non-indexed tables, is required to size the Unification Server.

Sizing R/3 Enterprise

Like R/3 4.6C and previous releases, SAP's QuickSizer process for R/3 Enterprise supports two different sizing approaches, based on the kind of information you have at your disposal regarding your planned R/3 Enterprise system. The first sizing approach is geared toward identifying the number of concurrent users in various functional areas, weighted for any combination of low, medium, and high users. Functional areas include the following:

- Financial Accounting (FI)
- Asset Accounting (FI-AA)
- Treasury (TR)
- Controlling (CO)
- Enterprise Controlling (EC)
- Sales and Distribution (SD)
- Materials Management (MM)
- Warehouse Management (LE-WM)
- Quality Management (QM)
- Plant Maintenance (PM)
- Customer Service (CS)
- Production Planning (PP)
- Project System (PS)
- Personnel Administration and Payroll Accounting (PA)
- Personnel Development (PA-PD)
- Basis Components (BC)
- Business Work Place (BWP)

It's critical to provide an accurate mix of users; users should only be counted once (place them in the functional area they will spend the most time using). Also, unless you understand the sizing risks, avoid stuffing unknown users into a single "catch-all" functional area, like SD. A system sized for 100 SD users has different CPU and

memory requirements than a system sized for 25 QM, 25 PM, 45 PP, and 5 BC users, for example, even though the number of users is the same in both cases. This is because each functional area places a different processing load on the R/3 Enterprise system. Further, more functional areas require a greater minimum amount of memory than fewer areas; there is a basic RAM requirement that must be fulfilled for each different functional area.

The second approach to sizing R/3 Enterprise involves answering a series of questions related to transaction loads and quantities of output structures created (like the number of planned sales orders created per day) or relevant to the various functional areas. Called transaction-based sizing, it's more complex than user-based sizing but in return it is more precise. For each functional area, you need to provide transaction data like the following:

- The number of objects created annually (like FI documents, or TR postings, and so on), as well as the maximum number of objects created in your peak hour of processing. Note that the peak is a delta peak; the transactions per hour in excess of your typical 9-to-5 transaction load.
- You can also enter an “execution” period, which is the span of time in which you want to run a particular peak load (like perhaps a payroll run). This impacts CPU and to a lesser extent RAM sizing—the smaller the execution window, the more processing power required.
- The number of subobjects of each object (like the average number of line items in your typical FI document).
- The retention period of these objects, or how long they will reside in the database before they are deleted or archived. This directly impacts database sizing.
- In some cases, you can also enter the mix of object changes to object display activities. This impacts disk subsystem sizing, as changes equate to more-intensive disk writes or inserts, whereas display activity is read-based.
- Some functional areas also allow you to enter the maximum number of objects that will reside in the database.

Sizing mySAP Business Intelligence

Most BI sizings relate in some way to SAP's Business Information Warehouse. Before specific CPU, RAM, and other hardware-specific metrics can be nailed down, it's helpful to understand the primary architectures employed for BW systems, such as

- Three-Tiered Architectures, where the operational systems (like R/3) feed consolidated data to an enterprise warehouse, whereas smaller data marts are stocked with summary data. This is the most time-consuming and expensive BI solution to implement.

- Two-Tiered Architectures, where the operational systems feed only an Enterprise Data Warehouse. This normally results in a faster implementation than otherwise possible, sacrificing potential scalability of the BI solution in favor of ease of implementation.
- Two-Tiered Architectures, where the operational systems feed one or more functionally specific (that is, SD, MM, FI, or APO, and so on) Independent Data Marts. This is the most common two-tiered implementation, as it allows an IT organization the chance to support granular departmentalization of popular functional data and queries. Further, this is often at the expense of a dedicated repository for extracted, cleaned, and transformed detailed data.

Beyond the core release version and underlying architecture, a number of key sizing factors exist for BW, like the adoption of a SAP BW Persistent Staging Area (PSA) and general Operational Data Store (ODS). Both of these affect the *database size* and *performance characteristics* of the SAP BW system as well as the speed and additional system load of the operational systems during the data extraction window.

- Persistent Staging Area (PSA)—This transparent table for storing detailed requests in the original format of the transfer structure is actually nothing more than another InfoSource. The source system maintains a key request number, packet number, and record number.
- Operational Data Store (ODS)—The ODS provides a location for all raw operational data to be combined and housed, before such data is extracted, cleaned, and otherwise reformatted. It is often “persistent” too.

In the context of the ODS, the PSA makes up the first level and the ODS table makes up the second level of the ODS. Therefore, the first level consists of the transaction data from the source system, and the second level consists of the consolidated system’s raw data, as well as data from several source systems and InfoSources.

With this information in mind, we can now turn our attention to identifying the sizing questions that the SAP QuickSizer needs to have answered:

- Three different types of *users*, classified by the frequency of their activity and the reporting performed. Frequency is measured in *navigation steps per hour*, one of which is equivalent to nine SD dialog steps. Users include *information* (low or normal users), *executive* (or advanced users), and *power* users, executing 1, 11, and 33 navigation steps per hour, respectively.
- Three different types of *queries*. Information users spend 80% of their time viewing reports and 20% performing OLAP analysis. Executive users split their time evenly between reporting and OLAP analysis. Power users, on the other hand, execute intensive data exploration activities 100% of the time. With

knowledge of the user types and their activity breakdown, it is possible to estimate CPU and memory requirements.

- Number and types of *InfoCubes*, which are the objects within BW upon which reporting and analysis is performed. SAP BW ships with standard preconfigured InfoCubes, which can be described by attributes like dimensions, key figures, and length—with this kind of data, a sizing engineer can begin to calculate roughly how large your SAP BW database will be.
- Number of *periods*, or the amount of time measured in weeks that you want to keep a particular set of data.

Sizing BW is a challenge, to say the least; the aforementioned questions and data being collected are valuable in the hands of an experienced BW consultant, but can easily be misinterpreted otherwise. And because one of the major concerns in sizing BW relates to determining how big your BW database will initially be, and how quickly it will grow, experience with BW beyond simply sizing will make a huge difference in how close you come to sizing a useful mySAP BI solution.

Rules of Thumb for Sizing mySAP CRM

In addition to documenting and understanding the business scenarios that drive sizing the mySAP CRM server, sizing CRM can involve any or all of the following special-function servers as well:

- The *InQMy Application Server* provides an execution environment supporting Internet Sales business-to-consumer and business-to-business processes, implemented via Java Server Pages (JSPs). This takes the place of the SAP Internet Transaction Server (ITS) found in the earliest versions of SAP CRM.
- The *TRex Server* (Text Retrieval and Information Extraction) provides indexing and search capabilities for information held outside of the SAP CRM database, like catalog information or product documentation.
- A *Workgroup Server* acts as nothing more than a standard Microsoft SQL Server database server. I suggest that you size this using a traditional database sizing tool. In my experience, the end result for a “typical” Workgroup Server equates to a system capable of hosting 100 transactions per minute for every concurrent *Mobile* user expected on the SAP CRM system.
- A *Communication Station*, which acts as a gateway between front-end Mobile clients and the CRM system, and was originally built on Microsoft’s DCOM product running Microsoft Transaction Server (now based on Microsoft .Net technology). Sizing the Communication Station is akin to determining the number of active users who are connected to the system and simultaneously

synchronizing their data between their client machines and the various data repositories maintained by mySAP CRM.

- A *Multi-Channel Interface Server*, which maintains the relationship between the CRM server and any CTI, email, chat, and similar middleware/services. These are not processor-intensive; something like a two-processor system with 1GB of RAM and 36GB of disk space will meet most organizations' requirements.

Of course, the CRM Server is the “central” server of any SAP CRM solution, and can be hosted on any number of UNIX and Windows platforms. Central to providing this functionality is another server, the Internet Pricing Configurator (IPC). The IPC runs only on a separate Windows 2000 server and is a horizontally scalable Java application that provides pricing calculation and product configuration capabilities. For sizing the IPC, I suggest that you determine the peak number of orders received per hour, where it is assumed that each order is composed of four line items and an associated four images. Or, if only transaction data is available, determine the peak number of objects created in the potentially heaviest day of CRM processing. Next, it's also important to determine and characterize the number of various CRM users, such as the following:

- *CRM Internet Sales Users*, who are usually split into either browsing users (the majority) and users who actually purchase goods. Browsing users tend to navigate the site and catalog, gathering information about products. Thus, they impact sizing the Web server component of CRM, but not the CRM server itself. Users who actually purchase goods go through the process of searching, and on top of this activity they eventually create and fill a shopping cart and then proceed to purchase the goods.
- *Mobile Sales Users*, who spend their time uploading data to the CRM system a few hours each day or so. For sizing purposes, the goal is to determine the greatest number of users expected to access the system in the peak one-hour period (usually in the very early morning or later in the evening, in my experience).
- *Customer Interaction Center Users*, usually tasked with supporting telesales, tele-marketing, and service/support functions. Interestingly, these users not only place a load on the CIC component of CRM, but by virtue of the work that they perform, they also generate transaction and other loads across the rest of the CRM system. Therefore, according to SAP's QuickSizer, these users are actually counted twice—once as a CIC user, and once in their respective business or functional area.

If details related to the type of users have not yet been determined, but you still want to estimate (for budgetary reasons, perhaps) what a CRM configuration might

consist of, you can instead estimate the number of concurrent users expected on the system during its peak period, using the following rules of thumb:

- One concurrent low user equals 3 orders per hour
- One concurrent medium user equals 30 orders per hour
- One concurrent high user equals 90 orders per hour

Certain activities tend to take place on the mySAP CRM system, too, which impact sizing. For example, managing opportunities and activities, creating customer orders, performing service-related transactions, and managing data related to customers, products, and even projects consume CPU, memory, and disk resources. It's also important to estimate items like the number of incoming and outgoing CIC calls handled per hour, the retention period (measured in months, not weeks as is customary in other mySAP solutions) of objects maintained in the CRM database before they are archived or deleted, and the peak volume of objects (like sales orders) created in a certain time period.

CRM is not an island unto itself; it ties into many other mySAP solutions, especially your core R/3 system. For example, each CRM sales order is eventually processed on the R/3 system. I suggest employing traditional SD modeling to determine this impact, using the SAP QuickSizer and its SD-SLS type of functional area.

SAP CRM can also touch APO—availability requests are transferred between the CRM server and APO system via SAP's Remote Function Call (RFC) communication service. And CRM's Mobile Sales users can leverage SAP BW for reporting and limited data mining; data is moved from BW to the SAP CRM server, and then downloaded to your Mobile Sales user in the form of an Excel workbook.

Finally, sizing for the CRM front-end clients (desktop and laptop interface devices, for example) must be performed. Client requirements are uncomplicated, but important. Three different user interfaces are available, but minimum front-end requirements dictate a 350MHz processor, 256MB of RAM, and about 30MB of free disk space, regardless of whether you employ the standard SAPGUI or the newer JAVA or HTML-based interfaces. Note that the addition of a mobile client bumps the minimum amount of RAM significantly, though, to 512MB. Further, at least 500MB of free disk space is required for mobile users.

Sizing Considerations for mySAP PLM

Product Lifecycle Management (PLM) supports users responsible with managing product, asset, and process information at any point in the product life cycle, from selection and purchasing through production ramp-up, installation, operation, engineering changes, maintenance/repair, retirement, and more. From a sizing

perspective, you need to understand what exact functionality you will be implementing. Choices are many, including the following:

- Lifecycle Data Management
- Asset Lifecycle Management
- Program and Project Management
- Lifecycle Collaboration
- Quality Management
- Environment, Health, and Safety (EHS)

PLM is closely linked with SCM and CRM, and is implemented as an enterprise portal solution. Like typical portal solutions, it facilitates communication and collaboration between users; this activity must be quantified and fed into sizing models. Until SAP includes mySAP PLM in the Quick Sizer, I suggest that you approach this as a Portal sizing exercise, and work closely with SAP AG and your hardware partner's SAP Competency Center.

Supply Chain Management Sizing Considerations

Sizing SAP SCM normally means focusing on SAP's Advanced Planner and Optimizer component. Hardware vendors require a lot of very detailed information to even come close to sizing APO accurately. Questions about how you will actually use APO need to be addressed first, including which APO components you plan to implement. These include Demand Planning (DP), Supply Network Planning (SNP), Production Planning/Detailed Scheduling (PP/DS), Global Available to Promise (ATP), and Transport Planning/Vehicle Scheduling (TP/VS). Next, requirements like the following need to be understood:

- *Characteristic combinations*, or the number of different properties that can describe an object. The greater the number of characteristics, the greater the CPU load placed on a system.
- The number of *key figures* that reside in liveCache, which directly impacts the amount of RAM required in the liveCache server.
- The number of *Demand Planning versions* and *Supply Network Planning versions*, which are scenarios used in forecasting or simulating demand. Each version is assumed to consume roughly the same amount of space; more versions therefore require additional disk space.
- Duration of each *Planning Run*, or how quickly you want to execute your planning run. Faster runs obviously require more CPU processing power.

- How *often* these Planning Runs will be executed; daily, weekly, monthly, and so on.
- The number of *periods*, measured in weeks. Different types of periods are requested (historical, retention, planning, and others); each type impacts disk sizing.
- Additional data related to the number of orders (sales, purchase, planned, and so on), orders transferred from R/3 to APO, available-to-promise (ATP) and capable-to-promise (CTP) checks, stock locations, and so on, all of which primarily impact CPU sizing.
- Number of APO users, which also impacts CPU sizing.

In the big scope of sizing APO, the number of users is really of little consequence, unlike most of SAP's other mySAP components.

Sizing mySAP SRM

SAP's Supplier Relationship Management offering is much greater than simply eProcurement, even though its roots are planted squarely in what was only a few years ago SAP's business-to-business procurement product line. As you see in Figure 7.7, an SRM solution can be quite complex. The eProcurement product line alone consists of SAP Enterprise Buyer, Requisite BugsEye, Requisite eMerge, SAP Business Information Warehouse, and SAP Enterprise Portal.

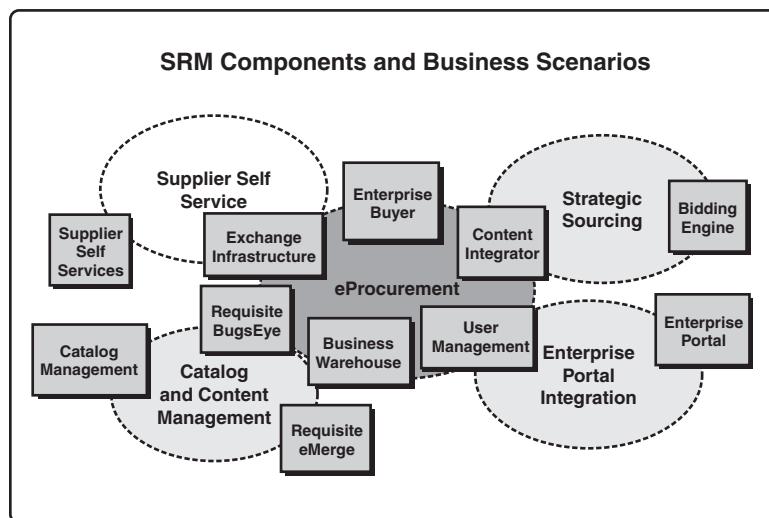


FIGURE 7.7 SRM business scenarios encompass much more than simple procurement processes.

Like other mySAP.com components, you must first determine the SRM business scenario(s) that are to be implemented; each has different required and optional components, and impact the sizing process accordingly. Other components come into play, too, enabling core SRM business scenarios, including SAP R/3, BW, Exchange Infrastructure (XI), SAP Content Integrator, SAP Supplier Self-Services (for supplier enablement, which itself requires XI), and the SAP Bidding Engine (which supports strategic sourcing). All of this, and the following factors, can complicate mySAP SRM sizing:

- As of version 2.0C, SAP Enterprise Buyer has supported MCOD, which is SAP's strategy for supporting multiple mySAP components on one database. As SAP R/3 4.6C SR2 also supports MCOD, some interesting synergies and economies can be gained by combining these two normally discrete databases into one.
- Support for an external catalog (running on Web-based marketplaces or your own supplier's site) minimizes hardware required to support catalog hosting activities, but impacts network activity, not to mention security.
- SRM also integrates well with SAP Enterprise Portal, allowing you to benefit from single sign-on access to mySAP SRM from the portal, among other things, but also blurring the line between the two from a sizing perspective.

Details like the following need to be understood or analyzed with regard to business scenarios that might be implemented:

- List all of the business scenarios you intend to implement in your SRM landscape.
- Identify your average document size (for example, the size of your typical purchase orders), measured in terms of line items created and processed in your SRM scenario.
- Identify how many procurement transactions will be performed each year, and how many will specifically be processed in your peak hour.
- Characterize shopping cart details—the creation, display, transfer, and update of a user's shopping cart needs to be determined as well as possible.
- Identify your total number of trading partners.
- Identify your total expected number of named and concurrent users (where concurrent users are actively logged into the system and browsing or otherwise performing work).
- Identify auctioning details, including the average number of line items and number of bidders.

Then leverage a throughput-based sizing model that addresses the processes to be implemented. A sample process might emulate a workflow commencing with browsing, then creating a shopping cart, then cutting a purchase order, then performing contract administration, followed by confirmation of delivery and subsequent invoicing.

After the specific business processes are understood, sizing can commence by using the SAP Quick Sizer to determine SAPS. This data may then be shared with hardware partners, to be further refined in terms of core CPU, RAM, and disk requirements for individual servers. The number of servers needed depends primarily on CPU resource consumption of each business scenario (other factors are not even considered). Business scenario CPU resource requirements can be measured and/or estimated if necessary, too. However, keep in mind that business scenario CPU consumption depends on the size of your business objects as well as transaction and user-based loads.

Next, the overall SAP system landscape can be looked at more closely. SAP AG's hardware and other technology partners can recommend the kind and number of servers that will fulfill the necessary "SAPS" requirements for development, Test/QA, production, and so on. Security requirements will come into play, too, as components will almost certainly need to be isolated and segregated by firewalls. And the need for high availability could easily require roughly twice the hardware otherwise necessary to satisfy your online user and transaction loads.

After an SRM system is in place, it's not unusual to continue the sizing process (at that point called *capacity planning*), to refine and tune the system. Java Virtual Machine (JVM) tuning is common, especially around memory heap and generational Garbage Collector (GC) tuning. JVM memory tuning can effectively optimize memory usage, and also increase runtime performance by reducing CPU usage for the Garbage Collection process. SAP has published an excellent note that reflects their latest JVM tuning recommendations—see *SAP Note 552522: Java Hotspot VM Memory Parameters*. Note that JVM tuning is specific to the operating system—Sun Solaris, HP-UX, and other operating systems present unique challenges in that they tend to include system-specific parameters. Tune the number of threads and number of dbpool connections, for instance, according to monitored usage.

mySAP Strategic Enterprise Management

Strategic Enterprise Management relies heavily on SAP BW; a business warehouse acts as the foundation for SEM functionality. SEM's analytic application components rely on multidimensional system-intensive OLAP data structures. Thus, it's key to architect and implement a well-performing BW system before any number of SEM modules can be implemented, including

- SEM-BIC, Business Information Collection.
- SEM-BPS, Business Planning and Simulation. SAP AG recommends that SAP Notes 358529, 411725, 407563, and 381022 be reviewed prior to sizing BPS.
- SEM-BCS, Business Consolidation. A rule of thumb for calculating the CPU and memory needs of BCS users is to perform a user-based R/3 sizing; multiply the number of expected BCS users by two, and then plug the result into the SD module.
- SEM-CPM, Corporate Performance Monitor.
- SEM-SRM, Stakeholder Relationship Management.

The SAP QuickSizer supports a combination user/transaction-based approach for sizing the individual SEM modules, too. For example, data like the maximum number of concurrent users per group, average number of manipulated data records, number of steps executed per user per hour, and so on can be entered.

Sizing the primary OLAP SEM/BW system, on the other hand, requires converting your expected SEM users into high, medium, and low BW users. Then, you need to calculate the data volume of a closing and the complexity of the various SEM groups (for example, the number of consolidation units, investment structure, and parallel hierarchies). Finally, to calculate disk space requirements, multiply the total number of expected SEM records by four (measured in kilobytes). These can be pretty difficult numbers to obtain—work hand-in-hand with the BW functional and technical teams to best position yourself to come even close to accurately sizing SEM.

Interestingly, SAP AG does not push a three-system landscape for SEM. If standard functionality is sought, only development and production systems are absolutely required. However, for consistency within your various mySAP component landscapes, I still recommend a Test/QA system. And in cases where you want to test different methods of slicing and dicing the individual SEM data structures, or provide some level of hands-on training to your development and super user communities, I highly suggest a business sandbox as well.

Selecting Your SAP Solution Stack Partners

The goal at this stage in the sizing process is to now perform a thorough “Side-by-Side Sizing Comparison” between the various SAP solution vendors that have made the grade thus far, to ultimately select the best SAP Solution Stack partners for your particular mySAP.com implementation. This final evaluation involves the following:

- A detailed sizing review
- Verification that each vendor’s proposed SAP solutions are indeed supported by SAP AG

- Verification that you will not be the first customer to do something “new” (unless that is part of a vendor’s value proposition and is consistent with your company’s technology perspective)
- Discussions with vendor-provided SAP Production references
- A review of your TCO numbers

I cover each of these key concerns next.

Pulling Out “New and Different” Sizing Data

With your Sizing Evaluation Team reconvened, the work of reviewing each vendor’s latest iteration at sizing an SAP system capable of meeting your business needs is necessary. Your first order of business is to be on the lookout for “extras” that you did not request. It is generally in the best interests of each vendor not to include these extras, as they tend to inflate the vendor’s solution cost. Things like ITS servers, or a SAN rather than a less-capable and less-expensive traditional direct-attached storage system, and even extras like enterprise management consoles and domain controllers need to be pulled out of sizings. In the end, each vendor’s sizing should be similar enough to facilitate a true apples-to-apples sizing comparison.

Of course, I have seen cases where a customer failed to completely understand the technical requirements of a requested solution, and therefore failed to request required components or failed to understand that an “extra” was indeed no extra at all, but more like a minimum requirement. In these cases, it’s critical to verify that the addition is indeed required, and then to share the new requirement with the other vendors so that they can update their own mySAP sizings. You also need to be sure to give extra points, so to speak, to the vendor who identified the shortcoming; this speaks well of their SAP Competency Center, and needs to be duly noted.

Verify Support for Architected Solutions

More than a few times I have seen an SAP technology partner recommend a solution component not actually certified or supported by SAP. Further, I have seen ambitious hardware partner account teams recommend SAP configurations not even supported by their own SAP Competency Centers. It happens. Technology changes quickly, but certification processes take time. Be sure to go through each proposed solution stack with a fine-toothed comb, taking care to review all recommended hardware and software components. If you have doubts or know that a particular piece of technology is quite new, don’t be afraid to check with SAP AG directly; leverage your SAP account team or SAP project manager. And be sure to work with each vendor’s SAP Competency Center to ensure that they, too, support the proposed mySAP solution—hopefully, they drafted the sizing for you in the first place, but this is not always the

case. In the end, the time spent verifying the integrity and supportability of your stack is time well spent, as it not only helps weed out troublesome or ill-prepared vendors, but also helps you avoid nightmarish post-Go-Live support issues.

Verifying Your Risk Level

Even though a particular solution or approach may be supported by SAP AG and its SAP technology partners, someone has to be first—one company somewhere in the world will be the first to implement a new technology or embrace a new implementation methodology in their mySAP project. With this dubious honor comes risk, of course, followed closely by potential reward. For example, many technology companies offer significant discounts to their customers “brave” enough to bet their business on new technology solutions and approaches. My biggest concern here is simply in regard to credibility. Ask to see previously used implementation plans or project schedules. Seek templates, vendor-published white papers and best practices, and other evidence of experience. Work with the vendor’s Competency Center to gain a sense of *their* feelings towards the risk versus reward equation. Finally, prepare to spend time with the vendor’s reference accounts, too, discussed in detail next.

Working with SAP Production References

Serious SAP Solution Stack vendors maintain a list of reference accounts that they can use to reinforce their credibility with potential customers. Your goal in speaking with a reference account is to validate the following:

- Your proposed solution stack is out there in the real world, running a production SAP solution.
- Their implementation and configuration experiences were in line with your own expectations, and what you have been told by your potential SAP sizing partner.
- The system actually performs well.
- Planned and unplanned downtime numbers are no surprise, or are explainable.
- Ongoing operations and support are consistent with your expectations, too.

This is a good way of checking on a vendor’s general past performance as well. If it’s possible, I especially recommend meeting with a reference face-to-face and spending time with a few members of their technical team, where you can often get unofficial or “off the record” information in addition to the usual data.

Revising Your TCO Analyses

Given everything that you have learned over the last few weeks and months regarding SAP sizing, it’s safe to say that something along the way probably changed.

Perhaps your original goal of a monolithic big-box solution fell apart. Or maybe your best-of-breed strategy began to look too difficult to support long-term. Whatever the reason, I am confident that at this point in your SAP project, you need to revisit the topic of Total Cost of Ownership (covered extensively throughout Chapter 5) and revise your own costing numbers. New numbers may well change your plans, after all. Only with new TCO numbers in hand can you truly be equipped to take the sizing process to the next level, and assemble the SAP partners and products necessary to solve your unique business problems.

Additionally, after the final round of SAP Solution Stack vendor reviews is pending, pricing is usually revisited as well. During this time, it's business-as-usual for vendors to take a fine-toothed comb to their overall solution pricing. Even a few percentage points can add up to big savings in projects ranging from hundreds of thousands to many millions of dollars. It is also common for you to structure a holdback or price/performance guarantee at this stage in the sizing process, too. In this way, your solution partners will continue to be motivated to perform to the best of their ability (and rapidly address inevitable integration, performance or service issues) all the way until the day of Go-Live and beyond.

Finally, understanding hardware acquisition costs versus ongoing operational and management/support costs for each potential solution needs to be revisited. Why? Because the grass may indeed be greener one way or the other. That is, over a three year life cycle, my own analysis has shown that Wintel-based mySAP solutions generally provide a better return on investment in most cases, for most customers. But I have also seen what happens when business requirements change overnight, or when an initially straightforward mySAP implementation grows to include many components and a hundred servers. In the end, my colleagues who talk of "pay me now or pay me later" acquisition versus management costing philosophies have a valid point in these cases. So I stress that you have this last opportunity to work through the numbers again before making a final decision—treat this opportunity to measure your end-to-end total cost of ownership seriously, and take advantage of it.

Selecting Your Core SAP Solution Stack Partners

With final pricing, apples-to-apples solution comparisons, and a host of other solution and partner-relevant data collected and analyzed, you can now work towards selecting your core SAP Solution Stack partners with confidence. This is often achieved by holding onsite vendor presentations, where any final questions can be answered and each vendor has an opportunity to present their overall value and solution proposition.

However, keep in mind that there is no single best way to assemble your final SAP Solution Stack. That is, in many cases I have seen an SAP customer select their *products* first—the various solution stack components necessary to deliver on their mySAP vision—and then select vendors. But in other cases, I have seen customers

select enterprise-savvy SAP technology and consulting *partners* first, only later to fine-tune the actual combination of hardware, operating system, database, and even mySAP.com components necessary to achieve their business vision. Either approach can prove successful, though you place a lot more faith in your technology and consulting partners in the latter approach—as you would expect, if they are truly strategic partners.

Evaluating Specialized Solution Stack Vendors

With your core solution stack products and partners selected, you are in a good position to begin filling in any “holes” left in your solution architecture. These holes may be the result of special high-availability requirements, disaster recovery requirements, the need to support special types of system accessibility, and so on. Thus, specialized or “niche” SAP Solution Stack vendors like disk subsystem manufacturers, makers of Internet-facilitating or load-balancing gear, test-tool vendors, enterprise management and other specialty software vendors can be brought in after your core SAP Solution Stack partners are identified.

The key to assembling a smoothly working solution from multiple vendors, rather than one or a few, is to first of all limit the players—if similar solutions are available from two different vendors, lean towards leveraging the vendor with whom you already use in your SAP or other enterprise environment. Next, in cases where different vendors are brought in, plan in advance how support issues will be addressed. In other words, you will need to iron out all of the details regarding how support is handled and how inter-vendor problems will be approached and solved *before* you ever arrive at a problem. SAP AG models the best support approach I’ve ever seen by way of their PartnerPort facility in Walldorf—here, many vendors work side by side with SAP AG and to some extent each other. Outside of PartnerPort, the best SAP technology partners also host Joint Escalation Centers, support relationships, and similar support mechanisms between various hardware, software, and other vendors. I suggest that you work to identify exactly with whom your core SAP Solution Stack partners have their best relationships, and lean towards taking advantage of these communication and escalation bridges whenever possible.

- To read more about support strategies in preparation for Go-Live, see “Addressing Future Service and Support,” p. 640 in Chapter 17.

Executing the SAP Infrastructure Planning Sessions

Now that your primary and probably most of your specialized solution partners have been selected, you need to commence the post-sizing implementation planning and scheduling sessions. In my experience, all of your solution partners should be engaged in the initial kick-off meeting, subsequent sessions of which actually span a number of days. This includes your hardware partners, OS and database (software)

partners, your SAP project manager, and any other project managers already identified. Further, the data center manager and any key technical resources should also be present at the kick-off meeting.

Initiating the SAP infrastructure implementation represents a critical milestone in your SAP project plan, and is indicated as so in the SAP Implementation Project Plan (SIPP) found on the Planning CD. As such, there are many tasks and related milestones of a smaller magnitude that need to be identified, discussed, assigned, and managed. These tasks are covered in the next few sections of this book, divided conveniently into three days worth of meetings and discussions. The overriding goals of these sessions are simple: to understand timelines for implementation, and to place orders for various hardware and software system landscape components.

Day One

Day One of the infrastructure planning sessions revolves around introductions of key personnel, review of the scope of work to be accomplished, review of the key hardware and software vendor's products and responsibilities, and reiteration of how the infrastructure planning sessions fit into the SAP implementation big picture. In the past, I have adopted an agenda for Day One that looks like the following:

- Introductions—Key personnel from each solution stack partner, SAP, and the client's project team are introduced. Contact information is shared, as is a "job description" for each organization and for each person assigned to the project.
- Vision and Definition—The reason for implementing a mySAP solution is quickly reviewed, followed by a high-level review of the scope of work to be accomplished by the assembled team. Like the pieces of a puzzle, each partner is made aware of how their own contributions augment every other partner's contributions, in the end making the project possible.
- Project Sponsorship and Accountability—The role of each partner and the importance of each role are discussed. The term *project sponsor* is used loosely here, as it refers to the individual person from each partner organization tasked with responsibility for completing their piece of the overall SAP project. Normally, the client has long ago identified what they expect from each partner. At this meeting, though, the partner comes prepared to put forth the name of their project sponsor—a high-level single point of accountability representing the partner's interests in the SAP project.
- Project Management—Each partner must also come prepared to introduce their incarnation of a project manager, another loosely used term that simply refers to the single person charged with completing tasks relevant to their role in the project. In my experience, it's best to clarify these roles as much as possible, especially with regard to expectations. The skillsets and qualifications of each

project manager are reviewed, too, as are the number of hours expected to be consumed in project-management-related activities.

- Project Structure—Here, the project reporting hierarchy is identified, discussed, and usually revised. That is, the structure of the project’s key players, in terms of who reports to who and how issues are escalated, is covered. In large projects, it’s common to publish a project organization chart along with a scope statement. Status reporting, the use and timing of meetings, and other control and communications processes are ironed out as well. Finally, a schedule for regularly updating the master SAP Project Plan should be put in place.
- Key Milestones—As they are currently understood, key milestones are identified and discussed. These are subject to change in the next two days of meetings, but discussing them at a high level now helps to prepare everyone for the project in general, and the next day in particular.
- Quality Management—Processes for continuous improvement need to be developed over the next few days and embraced by the team. These processes need to be designed in, rather than “inspected” in. For instance, the use of workflow diagrams, recipes and checklists, and Pareto charts (a bar chart format useful in identifying tasks that fall into the 20/80 rule, where 80% of a project’s issues are related to 20% of the tasks or resources) represent quality tools. They promote feedback and continuous updating of documentation, lending themselves to fine-tuning simply by virtue of their use, rather than something that is performed once and never used again.
- Project Administration—How to address time and expense reporting, billing/invoicing, change control, and other administrative functions conclude Day One. This might also include any contract administration that remains to be addressed, including updating contracts to reflect delivery and payment schedules, pricing, discounts, performance bonds, and more.

As you can see, this first session addresses the big picture. In doing so, it sets the stage for the next two days, preparing you to turn your attention next to project timelines with regard to the actual technology solutions and system landscapes being implemented.

The Second Day

Day Two commences with a quick synopsis of the previous day, and then the floor is turned over to the client or SAP project manager who leads discussions detailing the following points:

- Go-Live Date—By establishing the desired Go-Live date, an experienced SAP project management team can “work backward” to identify when critical

production milestones must be accomplished and therefore when tasks supporting these milestones must commence. This is a good way of constructing or validating a high-level project plan, too, as it typically contains no *slack time* (free time between tasks) and therefore clearly identifies the project's *critical path* (project milestones spaced in time such that they represent the shortest possible number of days in which a project can be completed). This by no means results in a truly usable project plan, however! Rather, it serves as a high-level template or perhaps simply as a tool for developing the actual plan.

- Critical SAP System Landscape Milestones—After production's Go-Live date is nailed down, you can work backward to determine when other SAP system landscape systems need to be in place, too. Thus, the need and timing for implementing production, Test/QA, training, development, a technical sandbox, and so on can all be determined based on your production Go-Live and how much lead time needs to be given to tasks related to other activities. I suggest reviewing the SIPP I developed specifically for this book to give you an idea as to how many tasks and milestones remain to be addressed. Your own project plan should be similar, and even more detailed.
- Key Risks—Identifying risks to the project plan is an excellent method of promoting and then refining timeline discussions. Any risk to successfully completing a particular milestone related to each partner's set of responsibilities needs to be covered, and addressed or mitigated. For key risks, a contingency plan needs to be developed as well.
- Landscape Assistance—As each component within the system landscape is discussed, it naturally promotes the discussion of how much assistance the client thinks they will need in achieving SAP system landscape-related milestones. For example, my team often gets involved with technical sandbox and development implementations, but less so with other environments (outside of production or at least a review of the production environment prior to Go-Live). Why? Because after we train our clients in how to plan for and install a mySAP.com component, and leave them with a detailed recipe or checklist for repeating the installation, they usually want to not only prove that the recipe works, but also to train their own folks and save budget money by doing it themselves.
- Identify Realistic Timelines—With the critical path and key milestones identified, and an idea as to which partners will be engaged to support achieving these milestones, it is desirable to pencil in hard timelines for Go-Live of each mySAP.com component to be implemented as well as each landscape system. Add as much slack time as possible, and spread it out among the various tasks (I like to see each solution stack partner allotted some amount of slack time). Working backward with these realistic figures should give you an idea as to

when hardware actually needs to hit the ground, and therefore when the data center needs to be prepared. Everything else going forward can then be filled in pretty easily on the third day.

- Knowledge Transfer—Before leaving for the day, work to clearly understand both when and how much knowledge transfer needs to take place between the various technology partners and the client. Assign names of responsible parties, assign and document the actual tasks associated with this knowledge transfer, and document the method by which the knowledge will be transferred; hands-on training, written process/procedure documentation, use of high-level or detailed checklists, creation of Web-accessible content, and so on. In the consulting world, we call these *deliverables*, as they represent something tangible that my colleagues and I deliver to our SAP clients prior to concluding our part of a project.

The second day of planning sessions can run pretty long. I recommend that each partner and the client come prepared with their individual project plans or detailed timelines. In this way, the master plan can be more easily updated, and valuable time is not wasted trying to recall how long certain tasks take to complete, or how critical dependencies relate between tasks. As an aside, it should go without saying that if your partner cannot produce at least a template project plan based on previous SAP implementations, you should be very nervous—if you are paying for specific expertise in a particular mySAP component of supporting technology, you deserve to benefit from other project's efforts. A previously and successfully used project plan is proof of this experience, for example, and equates to at least a minimum level of credibility in my eyes.

On the Third Day

Whereas the second day of planning sessions focused on the timelines and the technology being implemented, this final day concludes the planning sessions by assigning names and responsibilities to project plan tasks. Thus, by the end of the day, everyone leaving the sessions will understand not only their own roles, but the roles of the entire team and how everyone's tasks relate to the project as a whole. And in doing so, you will conclude the sizing process, taking it from inception through planning the deployment of your physical SAP system landscape. Tasks to address on the third day include

- Put names to Solution Stack technical roles—Each partner needs to be prepared to commit the technical resources necessary to achieve their unique solution-stack-related milestones. This will include SAP Basis/Web AS technical leads, Server Hardware Buildout specialists, SAN/Disk Hardware specialists, OS Installation/Tuning specialists, DBA/Disk Subsystem specialists, mySAP Component SMEs, ITS specialists, and Front-End Client specialists. A discussion

of the skillsets required to perform the work associated with each partner's tasks, and how their resource fulfills these requirements in terms of experience and education, is standard.

- Put names to other technical roles—Each partner must also be prepared to identify their resources satisfying other technical roles, like High-Availability/Failover specialists, Stress-Test/Load-Test experts, Functional Testing specialists, SAP Solution Tuning specialists for their particular contributions to the SAP Solution Stack, and so on.
- Identify administrative key roles—More than just technology folks need to be identified. Each partner's project administrator or coordinator, account manager, documentation specialist, and other support and administrative contacts need to be documented and shared with the team. In small engagements, it's common to see these people wearing multiple hats, perhaps even sharing technical responsibilities with administrative ones.
- Identify client contacts—Although this varies, I tend to see most of the following positions held strictly by my clients. Regardless, it's important to understand who is actually responsible for the following areas that fundamentally support or underpin the SAP project: environmental/datacenter manager and specialist, network infrastructure specialist, data and application migration specialist, training coordinator, operations manager, and help desk manager.
- Refine timelines—with the details surrounding the technical implementation understood now, it should be possible to refine the timelines associated with each task such that the hours and days expected to complete each specific task are documented. This is called a *Work Breakdown Structure* (WBS) in project management circles, as it breaks down a project into smaller elements and then ultimately into *work packages*, which are simply tasks that can be accomplished in 80 hours or less.
- Refine and validate budgets—Although all partners can now provide their final costing numbers, not everyone needs to be involved with reviewing the budget numbers. Normally, this kind of activity is reserved instead for project management representatives tasked with managing the project's financials. This exercise also helps validate how close each partner came to estimating their costs, based on the imperfect information provided to them during the sizing process.

Students of project management have probably noticed that the three-day planning sessions covered many core project management fundamentals. The sizing process in and of itself represents a large subproject of your overall SAP implementation. So it only makes sense that managing scope, timelines, cost, quality, risk, procurement, human resources, and communications needs to be given the same level of attention here as at the higher overall SAP implementation project level.

Tools and Techniques

I have included a slew of sizing-related documents on the Planning CD, some of which facilitate number crunching, whereas others simply provide for side-by-side sizing comparisons. Miscellaneous SAP/hardware questionnaires have been included, as have sample tools and sizing documents in Word or Excel format. I included a special document entitled "LINK - Quicksizer SAP Solution Brief and Other mySAP Sizing Links," too, which points you to a variety of Web-based publicly accessible SAP sizing-related documents and resources. Finally, all of the figures found in this chapter are on the Planning CD, too, in Microsoft PowerPoint format, in addition to a slide that contrasts Web AS with ITS and another presentation that identifies the real-world performance benefits I've seen virtual arrays provide to my SAP customers.

Summary

An SAP hardware or software technology partner should be prepared to work with you long-term on your mySAP sizing process. Considerations like weighting factors, assumptions as to heavy, medium, and low online users, batch load assumptions, and other solution characteristics unique to each layer and component within the SAP Solution Stack need to be brought into the open and analyzed via the solution sizing process. This chapter walked through the sizing process end-to-end, starting with an overview of the sizing and blueprinting process, and culminating with selecting your key partners and kicking off your SAP infrastructure planning sessions. In the next chapter, with your solution stack finally nailed down, we can now take a closer look at staffing the technical organization tasked with implementing SAP.

8

Staffing the Technical Support Organization

Best Practices Approach to Staffing the SAP TSO

The SAP Technical Support Organization (SAP TSO) is the single most valuable resource an SAP customer has. That is, a carefully selected group of support personnel can impact every facet of the system, from minimizing downtime through intelligent change management/planning, to proactively monitoring system performance and database statistics, to ensuring an excellent user experience, to working calmly through emergencies and crunch times. In Chapter 4, I set the stage for designing and initially creating the TSO. Key positions were staffed, and important organizational decisions were made. Here, we take this initial work to the next level. With information gleaned from other chapters in terms of what your own general SAP solution stack will look like, and which SAP components you will implement, it is now time to begin the critical tasks of staffing the bulk of the TSO.

Not all staffing decisions can be made at this time, however. In fact, consistent with best practices, my goal at this point is to staff to the 50-75% level. You must focus on filling positions that directly support your near-term objectives covered in Chapters 10 and 11, which are to develop and begin installation/implementation of the SAP

IN THIS CHAPTER

- Best Practices Approach to Staffing the SAP TSO
- Staffing the SAP TSO: Rapid Deployment
- Bringing in the Best of the Best
- How to Retain Your SAP Staff!
- Tools and Techniques

Data Center. Skillsets and experience related to the following areas will therefore be crucial, or of the utmost importance, at this stage:

- Data Center infrastructure, the foundation of any IT project—this includes finding folks versed in calculating requirements for and implementing power, thermal/cooling, and space/rack solutions.
- Network infrastructure, especially in areas of static routing, designing highly available public links, designing high-performance backend links, and knowledge of three-tier client/server architectures.

NOTE

The individuals tasked with supporting these first two areas—network infrastructure and Data Center infrastructure—will all probably work under the leadership of the SAP Data Center Lead discussed in Chapter 4.

- Server and rack configuration, including capabilities in building out servers, loading OSs, optimizing rack infrastructure, addressing cable management, and so on.
- Disk subsystem design/review and installation. The disk subsystem and server/rack support specialists most often report to the SAP Infrastructure/Basis Lead.
- Security, high availability, and disaster recovery specialists (with enough knowledge and skills to facilitate the initial installation of the SAP system landscape, based on the Solution Architect's needs). In my experience, these folks report to the SAP Infrastructure/Basis Lead or the Solution Architect himself.
- Database administration experts, to work hand-in-hand under the leadership of the Senior Database Administrator (another one of the core SAP TSO positions covered in Chapter 4).
- Basic knowledge of operational activities like backup/restore, help desk requirements, and other traditional data center “operations” tasks.
- Experienced SAP basis-layer installation, including knowledge of bolt-on or complementary software components. For example, if SAP Enterprise Buyer Pro is on the schedule for implementation, you will need more than just SAP basis skills on the team—you will need Requisite BugsEye expertise, experience with Java Virtual Machines, and more. These folks also report to the SAP Infrastructure/Basis Lead.

On the other hand, people skilled primarily in performance tuning and tweaking, interface-level integration, and operations/help desk roles will not yet be required.

Not to say that you can't start looking for qualified people too early! However, if a particular phase of a project is not scheduled to kick off for six months, the people in the job market will be completely different by then—you're probably wasting everyone's time if you begin the process *too* early.

- ▶ To read more about when and how these final “holes” in the SAP Technical Support Organization will be filled, see “Identifying and Staffing the Remaining TSO Roles,” p. 425 in Chapter 12.

Three Ways to Approach Staffing

In response to historically inconsistent staffing practices, I have seen the rise of three different ways to staff SAP IT organizations. The method employed most often to staff the bulk of the organization is sometimes called the *resume-to-interview process*; even today, this is still the most common approach. Companies typically engage their human resources organization to seek out resumes of qualified candidates, hopefully based on really good input from the SAP hiring manager or team. Because this input is subject to technical interpretation, though, a series of phone screens and quite a few one-on-one interviews are required to even begin to qualify hopeful applicants. Not only does this process take time, but it puts a lot of faith into what was written in the resume.

A second method of staffing thus naturally evolved out of the necessity to ensure that a candidate's resume reflected their actual skills. Called the *staff-testing approach*, it forgoes much of the screening-focused interview process in favor of instead dumping a pre-qualified SAP candidate into a typical situation (or real-time hands-on “test,” hence the name). The candidate then has a certain period of time to solve the problem, or configure the system, or design a solution—whatever is applicable to the job being filled. What a prospective employer discovers using this approach helps them to qualify a candidate in terms of the following:

- How the candidate approaches the problem at hand, from a number of different perspectives—organizational skills, problem-solving aptitude, and so on
- Actual abilities versus what was presented in the resume
- How well the candidate performs under real-world stress

The staff-testing approach has its own limitations, though, in that some folks simply don't test as well as others. And the test itself may not be an accurate representation of what the work environment will be like. Finally, when word of the “test” gets out, it tends to lose its impact or surprise-potential, and other tests therefore need to be created on a regular basis.

Out of the shortcomings of these first two staffing approaches came a third approach, *try-before-you-buy* (similar to the more traditional *probationary period*)

approach). This approach shortcuts the interview process, and eliminates the requirements around creating and maintaining tests, by focusing exclusively on putting a candidate to work not only immediately, but for “free.” Then, their ability to interact successfully within the framework of the team, and the quality of their actual work, speaks volumes for them. This approach is tempting when there seem to be two or three equally qualified candidates vying for the same position. *Try-before-you-buy* is limited in that it’s only really possible when a candidate hails from a consulting organization; the consultant is therefore paid by his consulting organization, but no bill is ever presented to the potential customer during the *try-before-you-buy* period. This approach is risky, too, in that you are letting a relative “unknown” potentially impact your project. Past working references are therefore critical, as is creating a structured environment for the *try-before-you-buy* time period of anywhere from a few days to a few weeks. This makes it easier to evaluate and determine whether the candidate is truly a “keeper.”

As is obvious, all of these approaches have good points and not-so-good points. What you really need, though, is an approach that lets you verify the skills of the candidate, and how well they interact in real-world situations, without spending too much time and money playing around with multiple interviews, test scenarios, and so on. What you need is a comprehensive and quality “rapid deployment” approach to staffing the SAP TSO, covered next.

Staffing the SAP TSO: Rapid Deployment

The *rapid deployment approach* to staffing an SAP Technical Support Organization is based on a morphed resume-to-hire and staff-testing approach, with the following modifications:

- The human resources organization is neither responsible for developing technical job descriptions nor performing technical references; rather, the SAP support organization is (leveraging templates that I include on the *Planning CD* found at the back of the book).
- A technical phone screen takes the place of separate general and technical phone screens and interviews.
- Role-playing via *customer simulation scenarios* during a single “group” technical interview takes the place of testing and multiple technical interviews.
- This single role-playing technical interview allows for observing both technical and soft skills in action (more detail on *soft skills* can be found later in this chapter), including how well the candidate performs under real-world stress.

This approach also side-steps the risk inherent to *try-before-you-buy*, without sacrificing all of the value gained by seeing a candidate in action.

So, with the image of “rapid deployment” in mind, let’s take a closer look at a proven process used by more than a few of my own SAP customers (and partner consulting organizations, for that matter). Note, however, that I will be writing mainly from the perspective of the customer (or employer) implementing SAP here, not from the perspective of the technical staff seeking employment or a contract.

TIP

If you’re more interested in the flip side of this perspective, perhaps for your own personal employment education, simply “flip” what is written here. And pay attention—the more you understand what a client is looking for in an SAP job candidate, the better equipped you will be to actually land that position; you will understand your employment opportunities that much better, and how to best sell yourself.

Best Practices and the Rapid Deployment Approach to Staffing

Before anyone is hired, before even a single interview is scheduled or offer letter is sent out, a process needs to be outlined and followed to support rapid deployment, like the one shown in Figure 8.1.

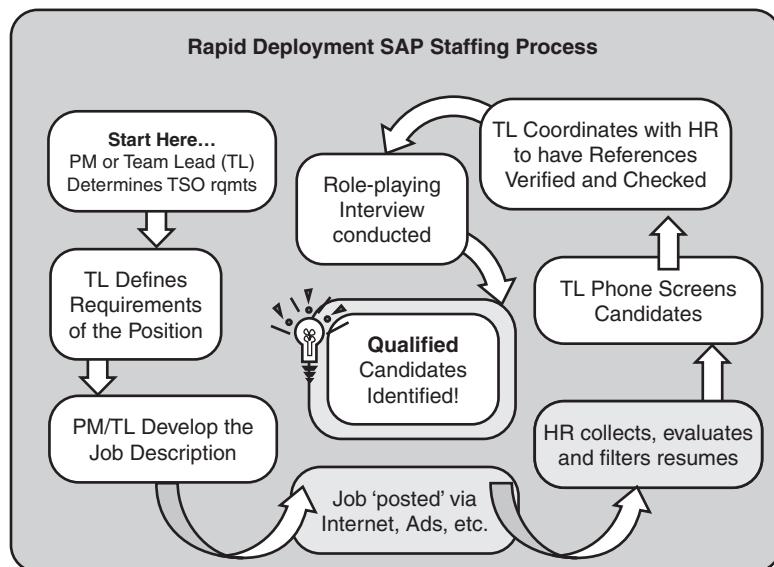


FIGURE 8.1 The Rapid Deployment SAP Staffing Process is not unlike many IT staffing processes, though a few caveats exist.

The *Rapid Deployment SAP Staffing Process* (RDSSP) is simply a documented set of steps to follow in support of the hiring/contracting process. What underscores the importance of the RDSSP as compared to other IT staffing processes is the fact that the SAP environment is nearly always “mission-critical.” This therefore impacts the *quality* of folks that need to be considered for positions in the SAP support team. And this in turn affects both the steps and overall timeline in the process, especially with regard to the following exceptions or differences:

- Pay careful attention to following the *process*. Side-stepping or unduly speeding up the hiring process will only get you into trouble—I have worked with these marginal hires, and it was no fun.
- The importance of technical and general *detailed references* cannot be stressed enough. Sure, many companies have a policy of not giving references. Approach it from a personal perspective, though (in other words, simply get the reference to talk to you), and listen and learn. In my experience, even in those companies that frown on giving references, it’s easy to find someone willing to talk—ask a candidate for the names and contact information of his former colleagues, and then approach your questioning from a “personal reference” perspective.
- Be prepared to spend *time* developing job descriptions, checking references, and performing phone screens, interviews, and other follow-up. These need to be handled professionally and thoroughly, in a *quality* manner. You get what you work for.
- The interview process itself needs to be much more rigorous than that utilized for most IT positions. Follow-up, and follow-through, in terms of detailed reference checks, in-depth technical interviews, and well-rounded group interviews utilizing role-playing and customer-simulation environments helps ensure that job offers are extended to “keepers.”

Given the nature of SAP implementations, the right candidates will be flexible, eager to learn, intelligent, and possess excellent people and communication skills. If in doubt, move on to the next candidate.

One final note, though—each candidate needs to be given the proper expectations as to the scope and timing of the review process. This benefits both parties; nobody needs to waste time on a thorough review and check only to find that the candidate has gone elsewhere in the meantime.

Next, let’s take a look at some real job description examples, an approach for comparing and shredding resumes, how to perform an SAP technical phone screen, setting up and conducting the face-to-face interview, and a method for ranking and evaluating candidates. So, whether you are the manager or a team leader performing

these tasks, or a candidate presenting himself for an open position, the following sections will definitely be worth your time.

Step 1—Developing and Posting Job Descriptions

For each SAP implementation, there are literally hundreds of ways that different jobs or occupations can be combined into broad positions of responsibility, or broken down into smaller cubbyhole positions (which I generally frown on, but they exist nonetheless). Some of my SAP customers have created very interesting, and in the case of some very large implementations, very narrow positions. Creating job descriptions, then, relates directly back to how your support organization is laid out.

Because we've covered organizational issues previously in Chapter 4, I will assume that you have an idea as to the approach you are taking. Again, refer to the *Planning CD* for sample job descriptions.

Most clients find advertising for SAP-specific "generalist" positions the best use of advertising budget and space. For example, a company advertising for an "SAP W2K Server Administrator" or "Network Technician supporting a mission critical enterprise" or "Oracle Database Administrator for SAP" will tend to yield more qualified candidates than similarly described non-SAP-explicit positions. On the other hand, too much detail might scare away applicants who don't feel they meet 100% of the requirements. For example, advertising for an "Oracle 8.1.6 database administrator to support a clustered Windows 2000 environment running SAP 4.5B" might inadvertently imply to prospects that *only* this specific solution stack combination is desired; others need not apply.

Step 2—How to Evaluate SAP Technical Resumes

When the pre-qualified resumes start coming in from HR, it's time to take action quickly for a number of reasons:

- The "best of the best" in the SAP job market will find a home somewhere; if you sit on their resume too long, it won't be *your* home.
- The hiring process takes time. From posting to technical screening to interviewing, and then negotiating to sending out an offer letter and setting up start dates, the process can take months.
- Even those candidates who are not qualified deserve respect for the time and effort they put into responding to your advertisements. Besides, the SAP job market is still small enough such that word travels quickly of clients who are difficult or irritating to work with. The junior consultant today could very well become the star expert you need next year!

Resumes need to be grouped into job positions, and then reviewed against the basic requirements of those positions. A matrix is handy at this point, as it forces an even comparison between all candidates vying for a single position. Further, as shown in Figure 8.2, it gives you a single source document, which you can begin using to track candidates seeking a position within your SAP team. Such a tool is especially useful over the Web, too, as it facilitates sharing this information anywhere, anytime, across the SAP TSO, the human resources department, and other teams.

Resume Evaluation Matrix

	Prospects/Candidates				
	Joe Dart	Mike Abe	Bob No	Lim Pho	Ash Ley
Date Resume Rec'd	12/19/02	01/07/03	01/07/03	01/10/03	01/10/03
Reviewed by (initials)	MDA	MDA	MDA	KT	KT
Date forwarded to TL	NO	01/08/03	NO	01/12/03	01/12/03
TL Feedback (go/no)	N/A	GO	NA	NO	GO
TL Phone Screen		01/15/03		NA	01/15/03
TL Feedback (go/no)		GO: 			GO 
Interview Scheduled		01/17/03			01/17/03
Role/Scenario Dev'd		YES- "A" 			YES-  "A"
...

 = data attached

FIGURE 8.2 The Candidate Status Matrix allows for rapid and even comparisons between candidates, and serves as a simple way of initially tracking applicants.

I like the term “Candidate Status Matrix,” or CSM, and have included a sample CSM on the *Planning CD*. The Candidate Status Matrix serves many purposes. First, it is an excellent tool for ensuring that timelines are adhered to throughout the hiring process. For example, when a resume hits HR’s inbox, there should be a time limit set for how long the resume sits unattended. The CSM helps everyone understand who is responsible for the next step in the Rapid Deployment SAP Staffing Process, and how long it should take to move the candidate from one step to the next. The CSM thus imposes work flow and forces accountability.

The CSM also allows for tracking all vital data on each applicant, from their name to contact information to the results of phone screens, interviews, and more. The CSM becomes your tracking system, in essence a single point of reference. And in the beginning of the staffing process, the CSM provides a way to start comparing candidates simply based on their resumes—professional appearance, written communication skills, completeness, all-important spelling/grammar skills, and so on—all of this is basic information that must be maintained.

Finally, the Candidate Status Matrix helps you manage closure on each candidate. In the end, a candidate is either hired or sent a “Thank You For Your Interest” note—the CSM assures you that no candidate falls through the cracks (in the sample CSM found in Figure 8.2, the absence of any more scheduled activity triggers a “Thank you but no thank you” response from HR).

Don’t make the mistake of underestimating the power and value of the CSM. Its use is obvious during the staffing process itself. But the CSM also provides the following compelling benefits:

- The left hand will know what the right hand is up to. If you have already interviewed and documented your staffing process, you will not make the mistake of (or waste your time) phone-screening or interviewing the same candidate twice. It happens!
- Even if it’s six months down the road, and you are looking for a position similar to an old one, and happen to see a familiar resume in your inbox, the CSM provides a easy way of “going back through your notes” to determine whether another conversation is worth everyone’s time. For example, if you passed on a candidate the first time because he hadn’t bathed in a week, or “exaggerated” too much on his resume, there is little reason to look at him again.
- The CSM can be used by a company’s corporate Human resources organization to document that the staffing process does not discriminate against any particular race, creed, color, and so on. It’s your way of ensuring that you are complying with self-imposed or otherwise governed hiring practices.

There are many ways to deploy and use the CSM. Historically, I have seen it deployed in the form of an Excel spreadsheet shared between one or two managers. Problems with this method of sharing data exist, though, especially in terms of version control. Better methods of sharing and accessing this data exist, and include:

- Create and maintain a true candidate *database* (that is, SQL Server Personal Edition, or even Microsoft Access, if that is all that is available).
- Make the database accessible through company-internal file shares, to facilitate access and data-sharing.

- Even better, provide Web access to the database, to allow home-based and remote parties access to a single repository of pre-employment data.
- Scan and attach all resumes to their individual candidate records.
- Tie your Internet or intranet posting process into the database, such that the resume and other posted information is automatically included for each candidate's record.

The more available the CSM is, the more it will be used. And the simpler it is to update, the more valuable it will become as a tool for truly managing your SAP staffing process. Be careful, though—this is truly confidential data and needs to be treated as such, as issues associated with maintaining this database potentially impact the whole corporation from a legal perspective. Plus, it's important to understand network and other bandwidth requirements when you begin deploying databases over wide area network or dial-up links—a poor database choice can bring a network segment to its knees if not implemented correctly.

Step 3—How to Perform an SAP Technical Phone Screen

Although many companies perform both a general screen and a technical screen, I feel that combining the two is in everyone's best interests. That is, if the only real purpose of a general phone screen is to validate that a candidate is indeed available, that he possesses sound communication skills, and that he seems to fill the basic requirements of a position, this can just as easily be addressed in the first five minutes of a technical phone screen.

Although collecting the responses to phone screen questions is critical, the manner in which the questions are answered is nearly as important. The following list of questions for the technical phone screen emphasizes these “soft” areas. Note that the items placed closer to the front of the list are more weighty—in other words, if you have concerns over the answers you are hearing to the first few items, you might consider cutting the phone screen short and simply moving on to the next candidate!

- Perform the tasks of a **general phone screen**, including assessing the candidate's availability, communication skills, general experience and background, and perhaps short-term and long-term goals. Does the candidate meet the basic requirements of the position and seem to “fit” in to your organization?
- Next, assess the candidate's **preparedness** for this phone screen. Does he have the answers to your questions? Does he ramble on from one question to the next? Is he trying to read from a resume or set of notes? Does he have a clue what your company does (has he done his homework), or what the project is all about? The answers to these questions should give you an idea as to the

level of importance the candidate places on *preparing in advance*, and even in *customer satisfaction*. You are his customer; how you are treated and spoken to probably reflects how the candidate will treat his customers in the SAP project.

- Record the candidate's **attitude or level of respect** toward this phone screen. Like his level of preparation, this too will likely reflect what you'll see day-to-day from him should he join your SAP team. Is the candidate chewing gum or trying to watch the kids? Worse (I've heard of this!), is he trying to catch up on email while talking to you? Or maybe taking in a ball game? Pass.
- Note the candidate's **tone**. Is he eager to both listen and talk, in a manner consistent with someone anxious to join your team? Does he answer your questions from a negative perspective, rather than a positive one? For example, when asked about why they want to leave their current employer, candidates might bad-mouth everyone from their team leader to the SAP project manager to their current company's lack of something or other.
- Did the topic of **salary** or other compensation come up? At this point, it should not—the information you have shared through advertising the position should provide candidates with a general compensation range. I highly recommend putting off any conversation about compensation until both you and the candidate feel that there is a good fit for employment. A candidate who starts nailing down his salary and benefits requirements five minutes into a phone screen scares me, personally. He may simply be short on common sense or a lack of awareness in how interviews should be conducted. On the other hand, he may just be playing the rate game (those of us in the SAP field call these folks *rate mercenaries* or *curious shoppers*, depending on their motivation). In today's market, there are plenty of SAP fish in the sea.
- Did the candidate ask any **questions** after the floor was opened up for questioning? Did these questions reflect that he was actually listening, or were they canned? Importantly, did the candidate *ask for the job*, or in some other way indicate that he is the right person for the position? These kinds of questions give everyone an idea as to how *interested* the candidate is in the position at hand.

I recommend intentional open-ended questions to encourage a candidate to talk. Specific questions may be key to a particular position, too. Regardless of the types of questions, ensure that these are also included in and tracked by the CSM.

Step 4—Setting up and Holding Interviews

Based on a candidate's phone-screen performance and attitude, they may be asked to actually "put on a suit" and show up to be interviewed (parenthetically, if a candidate asks what he should wear to an interview, note this in the CSM! It's always been

my opinion that unless a company *volunteers* a different dress code, an assumption of business attire should be made). The interview is where you get to see the candidate shine. Here, he puts his best foot forward and proves to everyone that he not only has the experience and aptitude to do the job, but that the project simply can't be successful without him.

Interviews take time to plan and conduct. In the interest of saving time, the following list describes a detailed process flow.

1. Start with introductions, and then talk a bit about yourself and the team. Move into discussing the project from a high-level perspective.
2. Move into getting the candidate to talk about his background, including general education. Then move into job-by-job experience, starting with his most recent relevant assignments.
3. Commence the role-playing scenario. Drill down into technical details. Stick with the scenario (get back on track as necessary).
4. Wind down the role-playing scenario. Wrap up with questions and answers and thanks.

Although it might seem like common sense, it's important to note that sometimes attitude, work ethic, and an ability to work with others can be more important than who has the best technical resume or experience. In fact, as more and more individuals join an SAP team, a new person's ability to work with and on behalf of the team only becomes more and more critical.

And on the flip side, we have all probably heard of losing a good candidate because someone on the *interview team* blew it. In one unfortunate incident, an overenthusiastic colleague of mine actually had the gall to throw a telephone at one of our candidates when the candidate answered his question with "I would call one of my technical contacts" during a role-playing session. Needless to say, we made an impression; the wrong one. The interview team therefore needs to *demonstrate* that it is both professional and personable, presenting itself as an extension of an excellent team of potential future colleagues to every candidate interviewed. Doing otherwise risks losing much.

Step 5—Key Interview Techniques and Approaches

The phone screen got rid of the unqualified candidates, and most of the marginal players, too. Now, with a candidate in front of you, you need to determine whether this is indeed the best person for the job. I like the following real-world approaches:

- Getting the business folks engaged in business-focused positions, such as those requiring liaison-like responsibilities, makes perfect sense.

- Relocating high-level, high-visibility candidates, or candidates interviewing for high-communication positions, from formal corporate settings, to observe these candidates in more “real-world” settings. A lot of insight can be gained by noting their behavior, language, and opinions at places like the company factory, refinery, or distribution center.
- A self-assessment form can aid in determining whether a candidate has the right approach or best mindset for tackling a particular job. I include such a form on the CD—see *Competency Self Assessment.doc*.
- Role-playing with three or four interviewers, especially when it comes to hard-to-fill technical positions surrounding SAP Basis or integration positions, is one of the best methods of confirming hard-to-prove skillsets.

As I alluded to before, a quality role-playing scenario can bring to light lots and lots of really interesting traits and characteristics in a candidate. What each scenario takes in terms of planning and execution time is more than made up in the quality of the SAP team that is ultimately assembled. Refer to the *Planning CD* for a number of different role-playing customer simulation scenarios, including:

- The first scenario involves troubleshooting a performance problem in a productive SAP environment, and covers much of the SAP Solution Stack.
- A “solution vision” question appropriate for solution architects and other senior technical resources.
- An “availability” issue can help identify how well a candidate balances customer orientation with restoring a system according to procedures (excellent for support, help-desk, and similar roles).
- A “people” issue rounds out scenario number four. This is a good scenario for team leads and managers, to observe how they might handle a common employment problem—the habitually late worker.

Basic Sample Interview Questions

Aside from role playing, one of the most effective methods of basic technical interviewing revolves around leveraging the technical and business requirements identified in the position description (a job posting or advertisement, for example). This assumes, of course, that the position description or advertisement itself was descriptive and fairly comprehensive. In the sample that follows, we are looking for a person to fill the role of an SAP Infrastructure Specialist—the basic questions that follow are asked from the perspective of the interviewer:

- Tell me about your experience supporting other SAP projects.
- When it comes to availability, scalability, and performance, which do you believe is most important from the SAP end-users' point of view, and why?
- Explain what you know about a standard three-system landscape for SAP, and how it differs from other system landscape approaches.
- How did you gain your experience designing high-availability network solutions for SAP implementations?
- What roles did you play in designing, installing, and supporting the servers and disk subsystems in past assignments?
- How do your current technical certifications and college education fit in with your experience on SAP projects?
- What would be your approach if you were asked to install a patch or upgrade into the SAP Production environment?
- How would your colleagues describe you in terms of your work ethic and ability to get things done in a team-oriented environment?

With the preceding list, it should become apparent whether the candidate understands what an SAP solution looks like from an infrastructure perspective, how changes are implemented, and how well he might work as part of an SAP team.

Advanced Skills Interview Questions

When it comes to determining whether a candidate possesses the deep or advanced skills required of a position, it is usually not enough to ask questions related just to the subject at hand. For example, an applicant looking to hire on as an SAP Security Specialist needs to be asked questions both specific to, and surrounding, SAP security. A team with which I once worked called this a *Deep 360*, as it implied drilling both down (deep) and around (360 degrees) a candidate's experiences and education. Thus, in regard to the security specialist position here, SAP authorizations and policies have to be covered as they make up the foundation of the job. But so too do OS-level permissions and policies, and database-level rights and permissions as well. In fact, a quick walkthrough of the SAP Solution Stack will reveal a number of layers that need to be discussed during the interview.

I also think there is a lot to be learned about how well a candidate understands a particular technical area when they are asked to *troubleshoot* a specific issue. Consider the following troubleshooting scenarios as we try to fill an SAP DBA position:

- The performance issue previously outlined in Customer Simulation Environment #1.
- A network issue causes problems with an organization's approach to Disaster Recovery—log shipping.
- In this case, a database simply stops working (because its logs become full).
- This final troubleshooting scenario details an organization's lack of capacity planning.

It's evident from these troubleshooting scenarios that a lot of real-world value is gained through this type of interview process. Of course, in this particular case depending on the scenarios that you choose, perhaps only a truly skilled SAP DBA might be in a position to intelligently administer these questions. I suggest that you use my scenarios as templates for creating your own.

Step 6—How to Rank Prospective SAP Candidates

After the interview is wrapped up, the real fun starts. Again, I recommend leveraging the CSM or some other kind of candidate assessment tool to help compare and evaluate each applicant in the following *paper* areas—paper refers to diplomas, certifications, recommendations/references, and so on, as evidenced through the phone screen, interview, reference checks, and other background checks:

- General experience
- College degree(s)
- Certification(s)
- SAP-relevant organization(s) and affiliation(s)
- Company-employee-based recommendation(s)
- Other references and recommendations

Next, a closer look at the area of hard technical or business-relevant skills is in order:

- Specific experience/skills related to the position
- Other skills that may prove complementary to the position
- Troubleshooting ability
- Verbal communication skills
- General intelligence

Although the aforementioned technical and business skills tend to help a candidate *get* a position, the items in my next list tend to help him *keep* his job in your SAP support organization. Use the CSM to compare and evaluate each applicant in the following soft-skill areas:

- Level of professionalism (range from high to low, reflecting the general feel of the interview or any specific events that either transpired or were discussed)
- Level of enthusiasm or eagerness (range from high to low, addressing to what degree a candidate seemed genuinely excited about the prospect of joining your SAP support team)
- SAP social personality (social skills range, from Team Player to strictly “Computer Room” personality)
- SAP skills personality (range from “new project/high stress” to “maintenance mode/low stress”—refer to the end of this chapter for more information)
- Level of adaptability (range from high to low, as evidenced in discussions of how the candidate managed long-term changes in his work or IT environment or career)
- Level of flexibility (range from high to low, addressing how the candidate reacted to short-term problems or issues, like the need to work over a weekend or stay late to resolve an issue, or the need to travel)
- Goal orientation (range from high goals to some goals to weak goals to unrealistic goals to no goals, where a candidate is really only penalized if they cannot fashion a response to specific questions like “what are your short-term and long-term goals?”)
- Overall potential (range from high to low, stating whether the candidate has potential to succeed in the organization beyond the specific position you are trying to fill)

Given everything in the preceding list, it's apparent that both position-specific skills and soft skills carry a lot of weight in determining the best candidate for a particular SAP position. The ideal candidate tends to be “balanced” across both areas, being not only technically adept or business-process savvy, but also quite strong in many of the soft skills.

And I would be remiss if I failed to mention the following—any applicant who says he can do anything and everything you want is not someone you want. Why? Because he *lies* (and not very convincingly). Always beware of candidates who seem to answer each of your questions with “yes, I can do that”—if this becomes a pattern, simply start asking for detail surrounding where he has done such-and-such

before. Drill down into the detail, pull it out of the candidate via role-playing, or whatever, and pay attention to whether a “yes” turns into a “well...uh...no” after a few minutes.

Interviews in the Real World

It can be amazing how often a lack of soft skills (and common sense!) gets the better of otherwise-qualified candidates. This real-world section is focused more on the candidates being interviewed rather than the organizations doing the interviewing—as we are all interviewed at one time or another, I believe there is value in this approach.

In this first true story, I vividly remember one case where an individual phone-screened with my team quite well for a senior SAP R/3 basis position, and an interview was set up for the next day. During an excellent interview over salad and pasta, the candidate impressed us with his experiences and troubleshooting abilities. We enjoyed dessert, and even sprang for cappuccino, as our future SAP star wowed us with stories of international travel and global roll-outs. We hated to see it all end, but eventually we had to head back to the client site, presumably with great news about our latest “find.” As we were all making our way out the door into the parking lot, the candidate casually tossed out the following question: “You guys don’t do drug tests, do you?”

We all had an uneasy feeling, as the senior managing consultant among us replied with something like “Yes—there’s a new-hire screen, and then the potential for random testing exists throughout your tenure with the company. No problem for you, I’m sure.”

The candidate was visibly affected. “Oh...no. Well, you know, a lot of technical guys like to smoke a bit every now and then.”

They *do*? Then it was our turn to be visibly affected. “Uh, OK, well, then you have a great day and we’ll get back to you ASAP.” Thanks for playing. Next.

In hindsight, our senior managing consultant should have told our candidate then and there that he had disqualified himself from getting the job, that the interview process was over, and that we were sorry.

In another case, we were interviewing contenders for a position as a SAP Data Center Lead. One gentleman impressed us with his demeanor, intelligence, and what seemed to be great personal drive or motivation. He was the senior operator in a mid-size SAP shop across town, and breezed through our customer simulation scenario, well on his way to landing the job. In the closing few minutes of the interview, we sat around and generally got to know one another, and an associate of mine fired off another question:

“Describe your favorite thing about your current position.”

We could never have been suitably prepared for his answer. It went something like this:

"Sometimes I like to go into the data center late at night, and turn off all of the overhead lights. I seal the doors to operations, close off the tape room, and get it really good and dark. Then I just sit back and watch all the blinking lights. Man, it's kick a@@!"

In retrospect, this guy was probably hanging out with the senior basis guy who asked about the drug test.

In my final real-world interview case study, I was wrapping up a two-hour interview process with a candidate who showed real potential. This individual had posted for an SAP technical support position advertised in the local newspaper, and in all respects he looked good. He met the qualifications and experience levels, his references checked out, a short customer simulation scenario went smoothly, and in general he interviewed well. My part of the interview, as the hiring consultant/manager, was to go over expectations, timelines, and eventually compensation. And I was going through his updated resume one last time, a resume that he had provided that morning.

Something caught my eye. In a different font on his resume, at the end of a list of technical certifications, were these four letters: MCSE, Microsoft Certified Systems Engineer. Today, that wouldn't phase me, as there are tens of thousands of MCSEs around the world. But back in mid-1997, this was a big deal, and it really set him apart for the particular assignment I needed filled—supporting a growing Microsoft NT/SQL Server/SAP account. It seemed as though the MCSE designation had been added as an afterthought, however. As a relatively new MCSE myself at the time, I would have felt compelled to place this in a prominent spot on the first page, ahead of other less worthy certifications, had I been the one doing the interviewing. So I questioned him. Although I don't remember my candidate's precise answer, I do remember asking additional questions as to which core exams and which electives he took to achieve this then-premier Microsoft certification. And I remember feeling quite awkward as he fumbled his way through a certification process he obviously knew so very little about—in an attempt to land the job, he had falsified his resume at the last minute.

My point with these three preceding interview cases is simply that everyone needs to take interviews seriously and professionally. It's important for interviewees to remember that the job is not signed and sealed until the day you start. Further, it's the little "side conversations" that get candidates into trouble. If you've been looking for a job a while, you tend to have the "interview" questions down pat. Questions like "what's your goal in five years," "Tell me about your weakest personality trait," and so on simply don't phase a seasoned interviewer. What *does* get the best of many of us tends to be the elevator talk, the discussions "before" or "after" we think the

interview actually starts and ends. Believe me, the interview *does not end* until you get in your own car and drive away. Our candidates could have done better if they had followed these guidelines:

- Do not ask stupid questions. Asking about the company's drug testing policy during an interview is a stupid question (it will come up naturally after an offer is made).
- Stay away from profanity. Unless you are interviewing for a squad leader position in the Marine Corps, there's little room for colorful expressions.
- Anecdotes or references to the strange little things we all do that make us "us" have no place in an interview.
- If unsure whether you should ask or say something, don't. I call this the "Keep your mouth shut" policy. It works.

I am sure many of my readers could add a few more compelling items to the preceding list. Again, the point is straightforward: Be professional.

Bringing in the Best of the Best

With the ideal candidates finally identified for your SAP Technical Support Organization, the task of actually bringing these folks on board lies ahead. That is, now that you know *who* you want to bring on board, you need to figure out *how* to make it happen.

Internal Transfers

Though it seems contrary at first glance, extending offers and transitioning people into the TSO can be especially difficult for folks moving from other company-internal positions. This includes folks already associated with the organization's IT function, or contractors supporting another functional part of the business, even though the SAP project has been established as a high-priority business transformation project throughout the company and at all management levels. Drafting internal players into the SAP TSO is tricky because although many different organizations may be expected to contribute to the SAP project, they will be losing key contributors themselves. Thus, the potential for alienating these organizations needs to be considered. An astute Project Manager will remember this as he exercises his power to draft individuals from other organizations, as both the project's and his own personal post-Go-Live viability rides nearly as much on the relationships he preserves as anything else. The following points are therefore worth reiterating:

- Internal transfers need to be treated with respect. What we have here is a member of the corporate “family” wanting to make a move to better both himself and the larger company. As the hiring entity, this is therefore neither the time nor place to insist that no transition plan is necessary, or to ignore responsibilities that need to be distributed among the internal transfer’s current team. Allow the internal transfer to put his affairs in order, out of respect to the organization losing him.
- The organization *losing* the head-count deserves some level of flexibility in start dates. A month is not unusual, though sometimes the critical nature and compressed timelines initially seen in an SAP project tend to limit transition time. Regardless, everyone needs to be sensitive to everyone else’s predicament—do unto others...
- Importantly, the losing organization needs to also remember that the internal candidate is not leaving the *company*; he is simply changing positions. This is no time for that organization to play power or control games, trying to hang on to their resource until the last possible second—the internal transfer and the company as a whole are the only ones who will be hurt, after all. And a perceived lack of cooperation on the part of the losing organization could very well make its way to the top echelons of the company, tarnishing an organization’s image in the eyes of the SAP Steering Committee and its various stakeholder constituents.

In the next few sections, I take a closer look at bringing external resources into the SAP TSO.

External Consultants and Contractors

Before any external consultants are actually brought in, it is imperative to ensure that you interview the actual person or people who will be onsite. That is, it’s still not too uncommon for a third-party consulting firm to allow their best folks to interview, only to follow up with their less-senior or less-capable individuals when they win the deal.

Another common mistake is putting the financial arrangements on the back burner during the third-party consulting interview process. Unlike the process for bringing in an employee, both the hiring and consulting organizations must understand financial arrangements up front. In this way, problems surrounding not “getting what you pay for” are mitigated. These financial arrangements are often documented by the consulting party, too—simply request their consulting price-list early in the process, with the understanding that all published prices are typically negotiable by 20% or more (especially for long-term contracts, where discounts of 30-40% are not uncommon). A sample “Consulting Price List” is shown in Figure 8.3—note the

detail in terms of classifications of consultants and pricing. It's also not uncommon to obtain sample resumes that reflect the real-world skillsets held and projects worked by various levels of consultants.

Technical Consulting Positions/Pricing

Consultant Job Code	Job Title/Basic Job Description	Part Number	Hourly List Price \$	Discounted Price \$
TCSUP	Admin & Tech Support	C-ADMIN-EX	65.00	60.00
TCX1K	Junior Consulting Ass.	C-CA0A9-EX	110.00	90.00
TCX1L	Consulting Associate 1a	C-CA1A9-EX	150.00	120.00
TCX1N	Consulting Associate 2a	C-CA2A9-EX	175.00	150.00
TCX1O	Consulting Associate 3a	C-CA3A9-EX	195.00	175.00
TCX3P	Project Manager 1a	C-CP1A9-EX	220.00	190.00
TCX3Q	Project Manager 2a	C-CP2A9-EX	250.00	210.00
TCX3R	Project Manager 3a	C-CP3A9-EX	280.00	230.00
TCX3S	Project Manager 4a	C-CP4A9-EX	315.00	250.00
TCX4E	Consulting Manager 1a	C-CCM1A9-EX	220.00	190.00
TCX4F	Consulting Manager 2a	C-CCM2A9-EX	250.00	210.00
TCX4G	Consulting Manager 3a	C-CCM3A9-EX	280.00	230.00
TCX4H	Consulting Manager 4a	C-CCM4A9-EX	315.00	250.00
TCX4A	Consulting Director Id	C-CD0A9-EX	315.00	250.00
TCX4B	Consulting Director IId	C-CD0A9-EX	315.00	250.00
TCX4C	Consulting Director IIId	C-CD0A9-EX	315.00	250.00
DIR6E	Practice Director	C-CD0A9-EX	315.00	250.00
DIR6A	Practice Director Id	C-CD0A9-EX	315.00	250.00
DIR6B	Practice Director IId	C-CD0A9-EX	315.00	250.00
DIR6C	Practice Director IIId	C-CD0A9-EX	315.00	250.00
TCX1Q	Technology Consultant 1a	C-TCAA9-EX	220.00	190.00
TCX1R	Technology Consultant 2a	C-TCBAA9-EX	250.00	210.00
TCX1S	Technology Consultant 3a	C-TCCA9-EX	280.00	230.00
TCX1T	Technology Consultant 4a	C-TDAA9-EX	315.00	250.00
TCX2K	Solutions Architect 1a	C-SA1A9-EX	220.00	190.00
TCX2L	Solutions Architect 2a	C-SA0A9-EX	250.00	210.00
TCX2M	Solutions Architect 3a	C-SSAA9-EX	280.00	230.00

FIGURE 8.3 Before bringing on third-party consultants, request a list or document that reflects technical job classifications/descriptions and pricing, like the one here.

Finally, to protect everyone and help ensure that business requirements are addressed, every consulting organization should be required to execute a scope-of-work (SOW) or letter agreement (LA). The SOW, or LA, should include the following:

- Scope of the engagement, including what is not included
- Specific services to be delivered
- Expected length of the consulting engagement
- Qualifications of the specific consultants to be engaged

- Assumptions made by both parties, in terms of who is responsible for providing office, telephone, and PC resources, who provides software/hardware, how access to systems are provided, and so on
- Price and payment terms and conditions
- Who owns the work developed
- Confidentiality and rights to ideas, concepts, and techniques developed
- Warranty of work to be performed, including recourse if work is performed in an unsatisfactory manner
- Contact information for engaging resources and well as escalating problems
- Signature page, to be executed by both parties

I have included a number of sample letter agreements for consulting services, which can be found on the *Planning CD*. It is the responsibility of the consulting organization to provide these, however. In fact, an organization incapable of producing a first-class scope of work or letter agreement should raise concern as to their ability to deliver professional consulting. Pay particular attention to terms and conditions, recourse for work performed in an unsatisfactory manner, and contact information. A quality consulting organization is proud of their processes for ensuring that their work is delivered in a quality manner, and they have the references and repeat business to prove it.

External Employee New-Hires

Even after all interviews are completed, all references are verified, and everyone agrees that the right person for the job has been found, it's still a lot of work bringing on a new employee. For example, the compensation package itself could take days if not longer to prepare. Details might include nailing down or explaining:

- Start date
- Annual salary requirements
- Benefits packages
- Sign-on cash bonuses, stock options, or other equity arrangements
- Regular bonus schedules, for example, for meeting performance objectives, financial objectives, or other goals
- Workday expectations/typical hours expected
- Confidentiality and rights to ideas, concepts, and techniques developed on the job

- Drug testing and other such programs, including one-time and regular or random nature of testing
- General company-wide information, including how to access employee data, information on employee portals and other similar resources, company-wide organization charts, and more
- Specific organizational information, specifically regarding the SAP TSO and other sister IT organizations, which would prove useful in solving or escalating problems
- Access to and provisioning of productivity tools and other resources, like a laptop computer, Personal Digital Assistant (PDA), pager, cell phone, email, voicemail, and so on.

Then, assuming the candidate actually accepts the offer, there is much work to be done to simply prepare for the start date. These details are addressed next.

The First Week

Regardless of whether a new-hire is an employee, contractor (consultant), or internal transfer, many tasks need to be done before perhaps any real SAP-related work is actually accomplished. This can often consume as much as a week's worth of time, and includes tasks like:

- Setting up and attending company and/or department-wide “orientation”
- Providing access to the facilities (parking, building, and so on)
- Completing new-hire paperwork (W4, benefits forms, and so on)
- Setting up physical workspace (office, cubicle, war room, shared area, and so on)
- Arranging for the configuration of a desktop/laptop, including access to applications or services like email, the Internet, word processing, presentation packages, spreadsheet applications, and so on
- Arranging for printing, faxing, and other business-related day-to-day services
- Providing business cards
- Providing all contact and organization information germane to the position
- Providing data or access to whatever workflow processes make sense for the position

- Attending safety training, legal compliance training, or other typically in-house-sponsored training
- All of the detailed information required to support the SAP TSO and other sister IT organizations—identification of and access to specific Internet/intranet Web sites, file shares, physical file cabinets, and so on

Given that 90% of the preceding list applies to any new member of the SAP TSO, we thought a checklist might prove handy. Refer to the CD for my *New Hire Orientation Checklist.doc*. And get started as early as possible on these tasks!

How to Retain Your SAP Staff!

At this point in the SAP support organization staffing process, you have spent countless hours building an elite team capable of carrying the company through the transition from a legacy or old system to a new enterprise-enabled solution. Or maybe you're like thousands of other SAP customer sites and are simply interested in preserving your excellent team! Either way, this section is for you.

Why go to so much trouble to write about staff retention? Simple—you really don't want to spend your time, money, and other resources training folks for other companies. And that is exactly what happens when staff retention becomes something you *plan* to do, rather than a mindset embraced every day. Other companies will steal away your best resources, leveraging all of the experience and training dollars you have sunk into your finest people.

This practice of wooing away the “best of the best” represents business as usual, and is a common plight in today's world. But you are not powerless—if you understand what motivates your SAP team, you can reduce attrition from a flood to something less than a trickle.

No, I'm not crazy! I have SAP customers who have had practically no attrition in their SAP technical support and SAP Basis/DBA teams in over five years. Yes, five years. Let's take a closer look at how they have managed to keep their best people through the high-dollar/high-demand years, and how they continue to keep their teams motivated and productive today. Pay attention, or risk keeping only your slackers, as your high flyers move on to bigger and better things at your competitor's SAP sites.

Understanding the Two Key SAP Support Staff Personalities

Remember earlier in this chapter when we discussed SAP skills personalities? They ranged from “new project/high stress” personality types to “maintenance mode/low stress” types. In reality, no one is truly one at the exclusion of the other. In fact,

most employees represent a mix of the two. But there is much to be gained in understanding what motivates each personality type, and how to use this information to the benefit of your SAP project. For example, employee and contractor turnover is hugely expensive when transition time, new-employee ramp-up time, formal training, and other on-the-job training is factored in. And a team's morale can suffer long-term damage if new projects, tasks, or assignments are delegated to the wrong team member. It is therefore important to note that a project can simply not be successful if only one type of personality is represented by the team, as we see next.

New Project Types

If your SAP implementation was performed exclusively by *New Project* (or NP) personality types, you would probably finish the design and implementation ahead of schedule and under budget. These folks work hard, love to learn, and typically thrive under pressure and deadlines. They embrace change, too, making them perfect candidates for SAP implementations and upgrades.

Unfortunately, there's a downside (as always!). Some of the trade-offs associated with employing a team of New Project types are as follows:

- They tend to be the most challenging to work with, as they are typically accomplished and experienced, with an ego to match.
- NPs tend to get bored with repetitive tasks, so they would not be happiest in operations and on-going support roles.
- NPs might be inclined to put off documentation and other such details, interested instead in "making it work" or "setting it up" and then moving on. If you value timely and precise documentation, you may therefore be disappointed. NP types prefer to churn and burn, leaving documentation to their support/maintenance colleagues.
- NPs find little value in standing still. As such, they need to be kept busy with projects and assignments that challenge them, force them to learn, and generally push them out of their comfort limits. If you can't do this, they'll find an employer who can.

New Project types make things happen. But as I alluded to in the preceding list, when things are more steady-state, you will need to begin transitioning a different kind of SAP technology person into your organization. That is, as the SAP TSO matures even prior to Go-Live, a good balance of both New Project and Support/Maintenance folks needs to be achieved.

Support/Maintenance Types

The weaknesses of the New Project type tend to represent strengths of what I label *Support/Maintenance* personality types, or SMs. SMs are simply motivated by and

attracted to different things than NPs; neither personality type is necessarily better than the other. For example, SMs prefer regular hours, less travel, more routine or scheduled tasks, and proven documented procedures. They thrive in operations roles, and generally pick up the slack for their New Project colleagues. The results I have seen in the field include:

- SMs often become long-term and loyal employees (or in some cases, long-term contractors).
- Steady and dependable, these folks show up every day and often represent the “core” of an IT organization.
- Support/Maintenance types are less apt to embrace change as quickly as their New Project counterparts, however.

My recommendation is simple—for a new project team of 20, bring on perhaps ten of each personality type, keeping in mind that five of the NPs will probably transition into new roles or areas as your project nears Go-Live. An excellent transition from an R/3 implementation, for example, might include moving some of these New Project types into roles supporting new SAP BW, APO, or Enterprise Portal deployments. Or, if you’re intent on Internet-enabling your enterprise, a position with responsibility for designing and installing SAP Web Application Server or Internet Transaction Server solutions makes sense.

As long as you understand what motivates your team, and take action to meet these needs, you’ll be that much more apt to maintain your organization. NPs and SMs complement each other in a symbiotic kind of way—it’s up to the various SAP TSO team leads and project management folks to ensure that the balance of NPs to SMs reflects the state of the project.

Communication—Keeping It Regular and Meaningful

Good communication is essential right from the beginning, during the interview process, and never really takes a back seat in an SAP deployment. The team kick-off meeting or new-employee introduction to the team sets the stage for formal communication; subsequent regular “stay in the loop” meetings reinforce how important sound verbal communication is. Finally, weekly or even daily *meeting minutes*, or details as to who attended a meeting and the subsequent action items discussed and delegated to meeting members, represent a critical written communication forum. Other written forums include regular status updates collected from each team member and assembled into a single SAP TSO-wide status update. Collected every Monday or every Friday, these often serve as agendas for further discussion, or to ensure that action items are indeed addressed and “closed out.”

In all of the time I have spent in countless meetings and conference calls with SAP customers and partners, I feel the following few points are most worthy of the ink in which they are printed:

- Show up on time, and start on time.
- Meetings must be short enough to stay focused, and long enough to cover the material to be discussed.
- If no action items come out of a meeting, the meeting was a waste of time in the first place. Avoid these! Meetings perceived to be a waste of time will be infrequently attended, and thus will serve less and less purpose over time.
- If no one documents the action items (that is, via meeting minutes), there is a better chance that they will not get addressed. Action items promote accountability in that they tend to remind the responsible parties that things need to happen before the project can move into the next phase.
- If no one is assigned to take meeting minutes, and publish these to the team within a day or so, less will get done. The person calling the meeting is ultimately responsible for documenting the meeting minutes (though delegating this task is common).
- If no one follows up on the status of action items out of meeting minutes, they are less effective, too. In most circumstances, I like the idea of the meeting organizer using the previous meetings' minutes as an initial agenda for the next meeting. For increased efficiency, it may be better to decide on a review schedule for each action item, such that subsequent meetings don't have to address each and every status item, only the ones scheduled for review that day.
- Take the time to document all but the most obvious of acronyms in all communications—I am usually in awe of the number of three-letter-acronyms assumed to be universally understood by customers, partners, and vendors.

Remember that regular communication instills confidence in and within a team or organization. And because they allow the team to be involved as a cohesive unit, regular productive meetings and other communications forums tend to promote improved team morale. In essence, you are more apt to feel like part of a successful team charged with a mission, and therefore happier, when regular meetings force action and ultimately progress.

Finally, there is also a link between communication and staff retention in the form of visibility of accomplishments. Most everyone enjoys a meeting where their own accomplishments are noted and praised. Similarly, checking off action items and other to-do's via posted meeting minutes makes accomplishments visible, and ensures that an individual's effort is noted and acknowledged.

Salary Requirements—Just the Beginning

I would be remiss to not talk at length about fair compensation plans, leveraging mySAP market and technology conditions to reward low supplies of these folks with high dollars, and so on. Here it is, then—pay your folks fairly, and review salary levels at least annually.

In the real world of SAP consulting and project work, I have said for years that a good salary motivates people to show up at work for only a few weeks. A good raise might even keep someone's interest another week or two. But it's a short-term motivator at best. After that, it's up to the SAP project manager and TSO leads to stay on top of what motivates their team members to continue to excel in their positions. Said another way, pay is an excuse for leaving but not a motivation for staying. One essential key to this is understanding and complying with the "most important thing," discussed in the following section.

The "Most Important Thing"

Have you ever been told "good job!" by someone who sincerely valued your effort? Words of appreciation like that often do more for morale and team-building than any annual raise or Christmas bonus. Sure, we all show up at work to get paid, but what really keeps most of us the happiest is encouraging words of thanks and appreciation. I like to refer to this as the all-important "attaboy" or "attagirl" approach to reducing employee turnover. Sincere and warranted positive feedback means so much to everyone.

Of course, like anything else, if too many attaboys are tossed around, they will lose their significance. A smart SAP Project Manager or team leader will use these just enough to keep them both unanticipated and "special." And the best project managers and team leaders will understand that timing is everything, and do their best to deliver timely well-deserved attaboys.

One way to keep one-on-one attaboys from sounding canned is for a Project Manager or Lead to give the attaboys publicly, increasing a member's visibility to his team and with regard to the SAP project in general. In doing this, the PM or Lead is putting his people before himself, taking a potential risk by implying that a component of the project's success is less the result of his own personal contribution than it is one of the team member's. In my eyes, this makes the Project Manager or Lead that much better to work for, though the following guidelines must be observed:

- Public attaboys need to be given to the person or team actually responsible for completing the work or achieving a particular milestone. Doing otherwise risks infuriating or disheartening others.
- No real good comes out of praise that is distributed for political instead of performance reasons. It's simply better left unsaid, as the team tends to know the truth anyway, and personal credibility is therefore tarnished.

- If a person is predisposed to giving public attaboys, he needs to ensure that they are distributed between different teams and different individuals. This should be easy, as no mySAP solution was ever single-handedly implemented. Doing otherwise can easily damage morale.
- Too much public praise, like private attaboys, loses its impact.

Performance and Other Incentive Bonuses

Performance bonuses are great for motivating behavior aimed at meeting goals or deadlines. For example, it's not uncommon to hear about things like \$5K, \$10K, and larger bonuses being handed out after SAP Go-Live. Similarly, cash bonuses and paid time off are also popular ways of rewarding the individuals responsible for meeting key SAP project milestones, like the completion of the SAP stress test, or rapid configuration of a development environment.

Many other incentives exist, too, like providing training. In the next chapter, I address training the SAP Technical Support Organization in detail. In this following section, though, I cover how to leverage this promise of training to not only bring the right people on board, but to help keep much of the overall team intact.

Training Opportunities

Holding end-of-project training out like a carrot in front of a hungry rabbit is becoming more and more common in SAP projects today. The idea is simple—help make a project successful, and you'll be rewarded with the opportunity to learn new technologies, mySAP.com components, or functional areas so as to assist with the next "big thing" at the company or client site. What really pays off with this kind of approach is patience.

It has been my own experience, for example, that sometimes the tasks required of a particular technologist for a particular project become mundane. Maybe the technologist has set up 10 SAP/Oracle clusters in the last year, and they just don't really challenge him any more. Or maybe the technologist has yet another solution sizing that needs to go out the door for yet another prospect that's only leveraging the particular vendor for better pricing from yet a different vendor. In these cases, the excitement is gone, and the SAP technology honeymoon is over. Our technologist, if he is a Support/Maintenance type, might be quite pleased with his role. A New Project type, however, would probably love to move into a new technology area, or get involved in a new project. Enter now the promise of new training and new future work or project opportunities for the NP. This could very well be enough to keep an individual committed and on schedule with regard to the current project.

Career Pathing

Like training, the promise of a new career in SAP is often held out as a compelling reason to remain in a particular SAP support position until specific project milestones or Go-Live or something beyond this is achieved. Some of the best “career pathing” I have been privy to includes:

- Building on a DBA’s value by adding SAP Basis and other SAP infrastructure skillsets is quite common.
- Extending the SAP Operations team’s responsibilities to include the implementation of basic change management waves introduces them to SAP Basis.
- Adding incremental skillsets to the SAP hardware/OS team, like Storage Area Network support or high-availability network support, provides great career advancement opportunities for these folks, and gives them another reason to stick around a few more years.
- Providing SAP functional on-the-job-training to your senior SAP integration people is a great way to push them into the realm of business-oriented support positions.
- Rewarding outstanding performance by increasing someone’s level of responsibility and moving him into a Lead position also increases his visibility within the organization.
- Moving SAP Leads into more senior management or other leadership roles of greater responsibility also promotes the idea of career pathing.

The best companies to work for use sound career management as a matter of “business as usual.” It makes sense, as it represents one of the best win-win scenarios in the employer/employee relationship. The company retains its talented workforce while leveraging the skills and experiences of its own people in different organizations of the company, and the employee meets his needs for recognition and advancement in-house.

Other Compensation Alternatives in the Real World

Although positive feedback, bonuses, training opportunities, and the promise of career potential certainly motivate most SAP professionals, other compensation alternatives play a key role in maintaining and retaining staff. I think very highly of many of these, and count myself lucky to have participated in quite a variety of interesting “compensation alternatives,” like those in the following list:

- One company my team and I supported for three months would surprise the SAP war room folks at least once a week with a catered lunch. Both the company and the individual consultants benefited greatly. We ate for free

without messing around with lunch-time traffic and lines, and the company got an extra 30 minutes of productive time, not to mention the benefit of informal meetings over lunch and the improved morale and loyalty that only really good New York style pizza and subs can provide.

- During the daily status meeting, and usually some time during the middle of the week, one of our SAP Client Project Managers would start the meeting by handing out a new \$100 bill to the member of the team who solved the nastiest customer problem that week (or achieved some other notable goal). Regardless of the reason for the award, if you're looking for a way to get everybody to your meeting on time, this method works! Even a high-dollar SAP consultant likes a new \$100 bill.
- An SAP Basis manager I know of would take the whole team out for an incredible no-expenses-spared steak dinner once a quarter. It was always an event to look forward to. And because this guy loved to eat, there was never any pressure to skip dessert. Or eat in a hurry. Or even go home.
- Another client of mine in a very rural part of the Midwest insisted that we, as consultants, get the “good rooms” at the local hotel for our short SAP infrastructure upgrade project. As it turned out, the town was something of a resort in the summer, but nearly emptied out after snow started falling (in September!). And the “good rooms” wound up being enormous suites with huge televisions, high-speed Internet access, and yes, gigantic Jacuzzi-style tubs.
- Another SAP Project Manager I know enjoyed treating different members of the team to an afternoon of non-competitive golf at any number of expensive or select club courses. He did a really good job of getting around to bringing nearly everyone on the team at least once. Not only was it a great way to put off work for half a day (because the work never actually goes away in the world of SAP consulting or support), it also provided each team member the Project Manager's ear for five or six hours.
- An SAP customer support center with which I worked used the “SAP Engineer of the Month” award as a means of rewarding the team member who demonstrated unique troubleshooting or problem-solving skills. Although the honor was nice, the best part of the deal was a paid parking spot next to the client's downtown building. Believe me, it was a coveted award!
- In a twist on the preceding example, one company would give the “SAP Consultant of the Week” two tickets to the city's NBA basketball team when they were playing at home. Even though the team was in pretty sad shape that year, it was still a fun and yet very inexpensive way to break up the week.

- Sometimes, just having a nice ride makes all the difference. Some of my favorite clients provide (or at least pay the expenses for) convertibles. After a 14-hour day doing transports or preparing for the installation of an SAP production cluster, there is nothing quite like throwing the top down in a rental. The beach, the highway, Sunset Boulevard, the middle of downtown Cleveland, near the Bay in Frisco, or along a lonely stretch of backwoods in Canada all become that much more exciting in a ragtop.
- During the two weeks prior to Go-Live, one client of mine sprang for lunch every day. The only rule was that we could never go to the same place twice. We wound up sampling probably every kind of food available in this small town—what a great couple of weeks!
- Finally, one of my absolute favorite places to work insisted that every Wednesday be declared “Aloha Wednesday” during the nine months prior to Go-Live. Why Wednesday, you ask? For maximum turnout—all of your consultants are in town Wednesday, whereas many will opt to leave early if you hold this kind of event on a Friday. Aloha Wednesday equated to half-day work days, volleyball, plenty of liquid refreshments, BBQ chicken or hot dogs or whatever the budget could spare that week, and just plain ol’ camaraderie. It is amazing how many technical and business issues can be solved while barbecuing chicken, or out on the volleyball court. Good times, good times.

Other kinds of compensation alternatives abound, too, in terms of very simple and inexpensive regular rewards systems. You may prefer, for example, to treat your SAP project team to “consumables,” like gift certificates, tickets to the latest movie, and similar one-time-use rewards. Vouchers for \$50 meals at really nice or interesting restaurants can be a lot of fun, too. The following are some examples of small gifts and tokens of appreciation that can go a long way in building and maintaining team morale:

- One company gives out weekly awards for the cleanest and messiest work areas. Prizes vary from flowers to gift certificates to company logo shirts and other merchandise. The owner of the messiest work area gets a broom or other such token that’s rotated from week to week, and they provide the gift for the following week as well.
- Another gives a rotating crystal plaque for the SAP team member of the month, with the name of the latest recipient etched below names of previous recipients. At the end of the year, a ceremony “retires” the plaque, where it is hung next to other IT and similar group awards near their break room.
- A large SAP hardware vendor’s enterprise marketing organization used to treat its SAP Competency Center to regular tickets to basketball and hockey games,

with the caveat that they take a customer or other key member in a partner organization to the game with them.

- Another favorite SAP customer of mine brought in donuts, bagels, or something similar practically every single morning during the project. Oftentimes, a huge pot of coffee would show up as well.

As you have seen, there are plenty of ways to motivate and ultimately retain your SAP Technical Support Organization. When staffing is completed for the core positions discussed in this chapter, and processes or practices are put in place to keep these core folks on staff, you can begin in parallel looking at planning for the selection of the SAP Data Center location. The tasks associated with planning and designing the data center are covered in Chapter 10, “Developing the SAP Data Center.” First, though, in Chapter 9 we will take a closer look at the training requirements that must be completed by the SAP Technical Support Organization.

Tools and Techniques

Following the outline of the chapter, the *Planning CD* contains a wealth of useful templates and documents, including:

- Job descriptions for positions covered in this chapter
- Interview questions for use in phone screens
- Other questions (both basic and advanced) for use in face-to-face interviews
- Candidate-ranking matrix complete with technical expertise and soft-skill areas
- A document entitled *Customer Simulation Scenarios-samples.doc*, which includes sample role-playing simulations and related questions
- Another document with a number of troubleshooting scenarios geared towards SAP DBAs
- Sample letter agreements, which might prove useful for comparison purposes to the documents provided by your selected consulting organizations
- An SAP “New Hire Orientation Checklist” document

The figures found in this chapter round out the CD’s offerings.

Summary

You have seen that staffing an SAP Technical Support Organization is a lot of work! Hopefully, the process I have shared, complete with templates and techniques and

more, will help streamline things. First and foremost, though, take care to bring in the right people in the first place—it's infinitely harder and even more time-consuming to manage under-performers out of your organization, not to mention the time that everyone else will waste picking up the slack.

Additionally, take the time to understand the two key SAP support staff personalities before you begin tossing around offer letters, or hire into an organization yourself. And if you are a manager or team leader in an SAP project, pay particular attention to the mix of compensation and other incentives with which you surround your team. Like managing under-performers out, it is usually even harder and more time-consuming to replace your SAP superstars. Finally, never forget the "Most Important Thing"—make attaboy's a natural part of your management style, paying special attention to neither miss the right opportunity to praise nor overuse this most important method of keeping your SAP team motivated and "in the game" for years to come.

9

Critical SAP Training Requirements

Introduction to SAP Training

In a perfect world, you would staff your SAP project with highly successful experienced personnel, people who keenly understand your particular environment and your unique Solution Stack. The SAP Technical Support Organization would have all the answers, and the project would come in way ahead of schedule and way under budget. Sleeping in would be encouraged, as would long lunches and lazy afternoons on the golf course, because the team was solving all the inevitable project/deployment problems in record time. By the same token, all of your end users would quickly grasp the new SAP-crafted business processes replacing their old way of doing things, training would be delivered to the entire body of end users a few days before Go-Live, and this training would be perfectly and forever entrenched in their minds.

In the real world of SAP implementations, though, a company is fortunate to assemble a technical team where everyone is SAP literate, much less experts. Each member of the team typically has a few “holes” in his or her experience that need to be filled in with training or on-the-job experience. Some of the team members will exit long before the big Go-Live handle is pulled, too, leaving you with gaping support holes to fill quickly. Other team members may simply not be up to the task at hand. And shifting technology or functional decisions (no matter how fiercely they are avoided) will create skillset holes where none existed previously. To top it off, there will be more

IN THIS CHAPTER

- Introduction to SAP Training
- Training and the Role of the SAP System Landscape
- Approaches to SAP Training
- Feedback Loops—Improving the Value of Training
- Tools and Techniques

than a few end users who resist or never attend training, and even for those who do attend, much of this training will unfortunately go in one ear and out the other before it's ever retained by putting it into practice.

Sound training of the SAP Technical Support Organization (SAP TSO), combined with effective and timely end-user training, can save the day, though. In this chapter, we will look at

- Primary SAP TSO roles and the “delta” training needs that are so often overlooked for each position
- How the SAP system landscape aids in training everyone from end users to various members of the SAP TSO
- Different types of, and approaches to, effective time-honored training
- Methods of ensuring that delivered training is indeed worth your company’s time and money

This last point is especially important in our knowledge-based economy, where unlocking the door to your SAP TSO’s potential can really pay off in terms of improved productivity and greater system availability. To this end, I conclude Chapter 9 with a rundown of individual technical and end-user roles, noting how customer-based feedback can be used to fine-tune or endorse a particular training curriculum.

Who Needs Training?

Who actually needs training? Certainly, everyone benefits from it. My focus initially in this chapter is not on the end users of the mySAP solution being deployed but rather the folks tasked with designing, implementing, and supporting this solution. And to help thoroughly cover the broad topic of training, I use (surprise!) the SAP Solution Stack model.

This method of analysis tends to ignore the leadership positions like project management professionals and solution architects, which for our purposes here is fine. That is, I believe that the folks charged with fulfilling these senior positions require deep long-term experience to be effective. Training can still be useful in filling in tiny gaps or holes, but the benefit overall is minimal compared to other positions. After all, who wants to hire an SAP implementation manager and train him in core project management practices? By the same token, who wants to bring in an inexperienced solution architect and teach him mySAP technology from the ground up?

In solution stack order, then, let’s walk through the following SAP TSO positions, focusing on *delta training*, or training beyond the minimum requirements necessary for each position:

- Data Center/Infrastructure Specialists—These specialists benefit from training in infrastructure installation (rack best practices, for example) and management/monitoring tools used in support of managing data center power and cooling.
- SAP Network Specialists—Training in static routing, designing and deploying highly available public links, and three-tiered client/server architectures in general is appropriate. Further, hardware-specific training is beneficial when high-performance or high-availability deployment options are being considered, like network bonding or teaming.
- SAP OS Specialists—Server deployment strategies and any OS-specific approaches to high availability or disaster recovery need to be taught. If a scripted approach to re-creating or deploying servers is deemed appropriate, training in the specific scripting language or tool needs to be provided, too, including access to test resources.
- Storage/SAN Specialists—Although training in SAN and disk subsystem architecture can be very valuable, in my experience it's the foundation provided by storage-platform-specific hands-on training that keeps things running smoothly. This is especially true of training that is specific to the OS being deployed on the servers attached to the storage systems, and even more so of the HA/DR alternatives being deployed.
- SAP Database Administrators—Specific database-layer performance tuning (training offered by the database vendor) as well as SAP-specific database training (training offered by SAP, for example) are both crucial. Knowledge of the storage system, especially with regard to HA/DR, is essential as well.
- SAP Basis and Web AS Experts—These experts should already be comfortable with the mySAP platform foundation being deployed. Delta training and custom workshops are often quite useful, though, in terms of identifying real-world challenges posed by the newest Basis and Web AS releases.
- mySAP.com Component Specialists—In addition to Basis/Web AS training and expertise, it's wise to train each component specialist in their particular mySAP product so that they understand core *capabilities and shortcomings*. Beyond that, hands-on training leveraging the SAP system landscape is key.
- SAP Functional Specialists—Each specialist needs to understand their functional area completely. It's also beneficial to train your functional people in related functional areas. And for functional specialists tasked with deploying business processes across multiple mySAP.com components, training in each component is wise.

- SAP ABAP, J2EE, and other developers—Developers must understand their development tools and platforms, especially how each tool is used and optimized in the context of a mySAP solution. SAP's formal training and certification programs are highly effective here, as is leveraging various systems within the SAP system landscape.
- SAP Integration Experts—The specific toolset or approach is important here, though access to test resources is absolutely essential.
- High-Availability and Disaster Recovery Specialists—The best HA/DR folks understand their entire solution stack and are aware of the various availability options that may be deployed at each layer of the stack. Training that orients these specialists to the issues and solutions relevant to creating a DR plan, along with how to successfully engage their IT organization and end-users, is crucial as well. Beyond this, they must also be provided with specific formal and hands-on training in the core HA/DR solutions to ultimately be deployed, including access to test resources.
- SAP Trainers—These trainers benefit from exposure to all mySAP and other bolt-on solution components (to gain at least a high-level end-to-end understanding of a “typical” mySAP enterprise), as well as introductory solution stack training. Beyond this, developing their presentation skills and attending other “how to be an effective trainer” training is important.
- SAP Security Specialists—The best security specialists understand all integration touch points that impact security, and how to defend their mySAP solution at each layer in the solution stack. Given the evolving nature of a mySAP solution, delta training in new access methods, the purpose and placement of firewalls, and applicable mySAP.com components represents the best use of company time and money, as does solution stack-specific training to fill in any technology holes.
- Front-End or User Interface Deployment Specialists—Each specialist must be comfortable with planning and deploying the chosen client access strategy, and therefore must understand the pros and cons of why a particular strategy may be selected over another (like deploying the classic SAPGUI versus the WebGUI, Citrix, or other similar approaches). These specialists tend to interact face-to-face with your end users too, and therefore need to not only be experts in your company's desktop, laptop, and other access devices, but also be trained to reflect the professionalism with which you want the SAP TSO to be associated.
- Documentation Specialists—Documentation specialists are experts at writing clear technical documentation, where options and details cannot be taken for granted. This is especially true for those folks tasked with creating custom checklists, installation cookbooks, and operations how-to procedures. Although

they do not need to be experts, technical “overview” training in the areas in which these specialists are asked to support is important—it will result in less ambiguous documentation.

- SAP Operations/Systems Management Professionals—These folks benefit from training specific to the tools they use. To become adept at monitoring and managing the stack, they will need hands-on experience with the various enterprise applications and other similar utilities, too, via a test environment.
- SAP Help Desk Analysts—As another customer-facing organization, help desk analysts benefit from broad solution-level training but even more so from training related to the tools used to troubleshoot and manage customer issues or “cases.” They need to be experts in these tools, so that they do *not* need to be experts in the entire SAP Solution Stack and each mySAP component deployed. Further, soft-skills training in communicating with different levels of end users is appropriate. And *Troubleshooting 101* training will serve front-line help desk analysts well, too, especially if this training is focused on issues that usually manifest themselves in terms of end-user response times or access to the system.

The preceding list is by no means exhaustive. But it’s *realistic*, reflecting the best places to spend scarce training time and dollars. So to really help a company track and manage its people from an education and experience perspective, a method like the simple spreadsheet illustrated in Figure 9.1 is necessary.

I call this a “Skillset Matrix,” and include a sample of one on the Planning CD. Although the idea is simple—a grid of skillsets, certifications, and products cross-referenced to each person—the value is tremendous. This spreadsheet not only identifies who possesses what knowledge, but it also in effect identifies *skillset backups*, or people who back each other up in terms of skillsets, certifications, and experience.

Timing Is Everything

All of us have attended training in one area or another, only to return home and either never put that training to use, or use it sparingly months later. Others have attended last-minute training courses but failed to grasp the truly pertinent materials, or be given time to do so afterwards, before needing to actually use the knowledge that was supposed to have been imparted. In both cases, the *timing* of the training, and not the training itself, became an obstacle to success. When possible, I recommend that all students attempt to schedule training—whether formal classes, workshops, hands-on sessions, or whatever—far enough out that there is time between the conclusion of the class and the need for what the class taught. To me personally, one to three weeks of time works out best—this gives me the time I need to address any holes in the knowledge I just gained by researching or re-reviewing the training materials, or simply discussing my questions with colleagues.

Sample Skill-Set Matrix													
			INSTRUCTIONS: Enter a "1" for EXPERT, "2" for COMPETENT, "3" for INSTALL ONLY, "4" for EXPOSURE ONLY, and enter "comments" as necessary or for clarification										
Manager	Name	Org	Data Center Infrastructure	Cisco/Networking	Compaq/HP Proliant Server	HP Superdome Servers	Sun UET/000	CPQ/HP SAN/Disksubsystem	HP SAN/Disksubsystems	HP -UX 10x 11i	Linux (SuSe or RedHat)	MS NT 4.0	MS W2K
BROWN	ANBONS, JOSE	SAP TSO	1		1	3		1		2	2	1	1
BROWN	BURDOCK, JOHN	SAP TSO		1	1				1	1	2		
BROWN	SANDERSON, GEORGE	SAP TSO		2	1						1	2	
BROWN	TALVE, JON	SAP TSO									1	1	
CLARK	BOOKING, MARTIN	SAP TSO	1				2				3	2	
CLARK	CLEXLEY, THOMAS	SAP TSO	2	1							3	2	
CLARK	GONIN, PAUL	SAP TSO	1								2	2	
CLARK	HEART, ROBERT	SAP TSO	1	1									
CLARK	MITTEN, DAVID	SAP TSO	2										
CLARK	PAIGE, DEVEN	SAP TSO		1	2								
CLARK	PLORTIER, CAROL	SAP TSO	2										

FIGURE 9.1 The Skillset Matrix helps an organization manage training, certifications, and more by identifying who possesses what knowledge.

On the other hand, training received too soon can equate to nearly wasted time and money. People forget things pretty quickly even in the best of worlds—training them on something they don't need to know or use for more than a month is like throwing away money. At this point, only the course-provided materials and really good notes hold any hope in helping a student refresh his or her memory. Another common method of battling this is to encourage students to actually use their newfound knowledge in test systems. Even better, holding them accountable for teaching their coworkers really reinforces training! This practice truly cements new knowledge into place, in fact, and can help to create pockets of subject-matter experts over time, too.

Training and the Role of the SAP System Landscape

I have mentioned generic *test systems* a few times already in this chapter. However, the term is relative, as you can see next in this detailed SAP system landscape breakdown—different systems serve different testing/hands-on training purposes, for very

different customer organizations. Remember, not all systems exist in every SAP implementation, either. It's possible, of course, but budget constraints typically limit most SAP customer projects to three to four systems per mySAP component being deployed.

- ▶ For SAP system landscape details beyond how each addresses training needs, see "A Closer Look at the SAP Solution Stack," p. 51 in Chapter 2.

- Technical Sandbox System—This system is specifically earmarked for *technical* training, along with providing a platform for technical testing. Thus, it services the training needs of every single member of the SAP TSO (which gives rise to scheduling challenges as different SAP TSO groups jockey for access to the technical sandbox—it's usually in high demand!). The entire Production SAP Solution Stack should be duplicated in this environment, especially when it comes to the server and disk subsystem platforms, high-availability and disaster recovery options, backup/restore products, and other critical or unique solution components.
- Development System—When it comes to training, the development system should only provide perhaps one or two “sandbox” clients, to allow some degree of trial-and-error testing and experimentation. A better solution, however, is to implement a “Business Sandbox” or “Development Sandbox,” which for all purposes is like a Technical Sandbox, except that it services developers and the like—the folks tasked with supporting the business.
- Training System—This is the most misunderstood system, in my experience. A “Training” system is normally dedicated to supporting SAP end-user training—how they execute business processes in a particular area, how they use the SAPGUI, and so on. But too many times I've seen training systems pulled in a hundred different directions, supporting technology training, development training, integration testing, disaster recovery testing, and more. All of this extra hands-on training and testing needs to be done, of course, but by using the Training system, end-user training time is compromised.
- Test/QA System, or Quality Assurance, or Consolidation, or Integration—Training in the integration and testing of configuration or functional changes is performed on this system, in a “training” client.
- Staging System—Emulating the most mission-critical tasks and business processes to be performed ultimately in Production is the role of this system. Thus, performance testing and hands-on training are commonly executed here, supporting the entire SAP TSO much as the Technical Sandbox does, but in a more formal, less “crash-and-burn” manner. Stress testing is often performed against this environment as well, as Staging typically duplicates Production.

- Production System—No formal training of any type should occur on this system (though informal post-Go-Live “training” is inevitable and expected among end-user colleagues).
- Disaster Recovery System—This system supports training specific to the disaster recovery solution implemented, or any number of high-availability options that may be used.

As you can see, a number of the systems described in the preceding list are especially designed to support training. Different systems are focused on providing training environments for technology staff, development staff, and end users. These three *core* training systems—the Technical Sandbox, Business Sandbox, and Training—are each visited next.

Leveraging the Technical Sandbox

Because the Technical Sandbox should duplicate or closely emulate Production, it serves many wonderful purposes:

- It supports basic “familiarity” training in your particular solution stack, including the hardware platform, OS, database, mySAP solution components, and so on.
- It also allows in-depth HA/DR training, such as that typically needed to support a high-availability cluster or log-shipping process. The Technical Sandbox is perfect for hands-on demonstrations and testing.
- It gives the staff the opportunity to learn how to perform backup/restore processes, and then actually put the process into use.
- The Technical Sandbox sets the stage for creating initial current-state and how-to/process documentation, all of which can be easily modified later to fulfill real production-level documentation requirements.
- Because the Technical Sandbox will also represent the system from which technology changes are tested and eventually promoted into Production, it makes sense to begin establishing and using these procedures as the starting point for change management processes.
- All of the operations/management toolsets and applications will eventually need to be learned, used, and understood; a Technical Sandbox facilitates this best.
- Finally, all of the experience gleaned from working on this system should help the SAP TSO begin building the contents of the DR Crash Kit.

With clear value like this, it's no wonder that more and more companies are implementing Technical Sandboxes for their mySAP enterprises.

- ▶ review the Crash Kit referenced in the preceding list, **see “The Disaster Recovery Crash Kit,” p. 210** in Chapter 6.

Development and the “Business Sandbox”

In the days of “simple” SAP R/3 implementations, the Development system normally was configured with a couple of sandbox clients, and development was simply managed by managing these clients. Today, though, complex multiple-component mySAP solutions require a greater level of integration and therefore deeper skillsets. Not only that, but the timelines can be such that not everything goes “live” at once—phased mySAP implementations might introduce R/3 and Enterprise Portal first, followed up later by BW, and then by APO, for example.

Such an approach demands discrete client-driven integration testing and training. But the risk to client-independent data found in a shared development/business sandbox system would place each client’s development code in jeopardy, hence the real need for a dedicated Business Sandbox. With a Business Sandbox, not only can multiple clients be created for education and experimentation, but additional ones can then be focused on figuring out integration challenges inherent to supporting different project phases. And consistent with best practices, the real development Golden Client would still only reside on a single Development system, the one housing the data to be actually transported to Test/QA. In this light, it occurs to me that the term “Development System” is really a misnomer; a better term would be “the pre-production repository housing the production future state,” in the same manner that the “Production System” is really the production business system. It’s a bit wordy, unfortunately, so I think we’ll stick with simply Development!

Wringing All the Value out of a Training System

The role of the Training system, or “Training,” confuses many people—the label “training” is simply too vague. As I said before, I view the Training system as the answer to your end user’s hands-on education needs. If the Training system is to be shared between different end-user groups, I recommend creating a schedule to help manage who is training with, borrowing, or testing what resources, and how long they intend to keep these resources. With this in mind, the Training system becomes useful in these ways:

- You can use it to train different groups of SAP end users. Most companies leverage a “classroom approach” to training end users, where 15–25 students show up in a physical classroom and learn the specifics of their functional area, for example. Each functional area is provided with its own client, to make it easy to refresh the system afterwards (as compared to restoring the entire database).

- The act of refreshing training clients from the “Gold Master” acts as training for SAP Operations or the Basis team, too, who will be tasked with refreshing clients on a regular basis before and after Go-Live.
- You can *also* use the Training system for assembling and refining further “how-to” documentation, especially documentation needed in support of transports, change management, and other multi-system activities.

When it comes to making the decision as to which training environment to deploy, I recommend erring on the side of conservatism. Why? Because it is easier to set things up first, and distribute training-focused systems to their respective user communities, rather than spending time managing schedules and priorities for these same communities when it comes to training.

Training to Support Your Unique SAP Solution Stack

Hypothetically, the SAP system landscape should reflect how much you need or value training. That is, an SAP customer with a Technical Sandbox in addition to Development, Test, and Production systems obviously values their SAP technology team having access to testing and training resources. Another customer with a Business Sandbox and Development, Test/QA, Staging, and Production systems places a whole lot of value on functionality, coding, and creating the best possible end-user experience.

In the real world, most SAP customers deploy dedicated Development, Test, and Production systems only, and try to take care of technical, development, and end-user training by leveraging both a client strategy and a system-wide resource schedule. In my experience, training systems are growing in popularity, though. If the budget is available, I would consider adding a training system according to the following guidelines:

- Add a Technical Sandbox if your HA/DR solution is complex, or if the SAP Basis/Infrastructure team is new to the solution stack being deployed.
- Add a Business Sandbox if the coding environment is complex. For example, a shop implementing mySAP.com by using ABAP/4, Java, and some older HTML ITS-based connectivity would benefit from the trial-by-error opportunities and other testing capabilities inherent to a Business Sandbox.
- Add a dedicated Training instance if multiple clients need to be maintained and refreshed often to support new-user and similar functional/business process training, or if a large number of end users need to be trained in a short period of time.

This is by no means money wasted. One of my customers actually redeployed their dedicated training system after Go-Live, in fact—they added another batch application server to Test/QA, and tossed the rest of the training system's resources into the Technical Sandbox, to be used for solution stack testing and education.

Finally, another general way to determine whether one of these dedicated training systems is right for your particular mySAP solution is to review the skillsets of your people, and identify weak areas across the technology, development, and end-user communities. Then consider implementing one of these training environments based on your weaknesses. And keep in mind what I discussed previously—after your training “holes” are filled, don’t be afraid to redeploy or reallocate your training resources to a new or current system within your landscape! Such frugal management of scarce resources can be one of the keys to a successful training program, much less the entire project.

Approaches to SAP Training

In the earliest days of SAP deployments, SAP and its implementation partners were so busy deploying new systems and developing a methodology for doing this efficiently (eventually labeled the “ASAP Methodology”) that little time seemed to be left to dedicate to training. Sure, formal training courses existed. But these were highly sought-after and impossible to attend unless you were a partner of SAP or an SAP employee yourself.

Consequently, those of us in the industry back then saw a lot of interesting things occur. Hardware partners like Compaq, HP, IBM, and others developed their own SAP project-team training, basis-layer workshops, and integration-focused SAP training courses. Third parties and other niche players sprung up out of the void, anxious to grab a piece of the SAP implementation budget earmarked for training. And internal organizations within many SAP customer sites were created or mobilized to prepare and deliver their own end-user and project-team-focused training curriculum.

SAP AG finally caught its collective breath sometime in 1997 and released a number of products that really turned the heads of its partners and customers, including

- *Information Database* (InfoDB). These small training systems consisted of two to three dedicated servers—one R/3 server, one IFS (Information File Store, or data store server), and optionally one IDES (covered later in this list). Most of these systems were sized for 20 to 40 users, based in large part on how many users would ultimately use the R/3 system productively.
- *Information File Store* (IFS), which housed all training materials from SAP, including PowerPoint presentations, various white papers and best-practices documents, configuration documents, technical notes, working labs, and more. Processing power was not critical, then, for IFS servers. Data storage needs, on

the other hand, grew quite rapidly from 1997–1999, culminating in the need for 70–90GB (which, back in the day of many 30GB SAP R/3 databases, was actually quite large). Later the combination of InfoDB and IFS grew into what became SAP's first Knowledge Warehouse product.

- *International Demonstration and Education System* (IDES). Later, as the Internet really took off, SAP renamed this the *Internet* Demonstration and Education System, and updated its reach somewhat. Originally based on a fictional motorcycle company, IDES systems were often sized for 20 to 40 users, and allowed these users to enter orders, move materials, perform production planning, and so on—all very real and very typical business transactions. This was accomplished by virtue of the fact that the IDES system was completely customized—it was no less than a fully configured and highly capable training solution “in a box.”

In the next few pages, I will walk you through the many approaches I have seen used at customer sites to get their teams up to speed. I will touch on a couple of newer approaches, too, including some that I believe are the most effective training methods employed today.

Formal Classes and Courseware

SAP AG and its key education partners offer formal classroom-based courses scheduled and delivered across more than 70 different training centers over the globe. For our purposes, the typical course offerings can be divided into three categories:

- SAP Project Team Training
- Consultant Education, which focuses on educating SAP Functional and Technical analysts
- SAP End-User Training

Project Team Training

SAP AG provides *Project Team Training*, which may be delivered to both consulting partners and customers tasked with supporting their SAP projects. At the core of this training lie more than 250 courses, which may be taken at one of SAP's training centers or at your own site. Courses are arranged in a hierarchy of three levels representing basic awareness, readiness training, and advanced high-proficiency curriculums. As students progress through their custom training programs, they typically move through these hierarchies for each functional or technical area of expertise in which they are learning. The curriculum overall is modular, too, lending itself to supporting the different needs and depths of competency sought by different students.

SAP's education courses and various technology and functional paths may be viewed at the SAP Service Marketplace, which is SAP's extranet site for customers and partners. The SAP Service Marketplace also supports online registration for classes delivered through their training centers. Simply access the URL <https://pgtravel.sap-ag.de/scripts/wgate/yhwbus/> when you have a course identified, and book it.

Consultant Education

The second formal type of course offering provided by SAP is *Consultant Education*, sometimes also called *Consultant Training*. Once granted to only a small number of SAP technology and consulting partners, Consultant Education is wide open to everyone today. It provides training in state-of-the-art SAP solutions, often leveraging SAP's own certified and usually very experienced consultants as trainers. Other courses are taught by SAP certified Technology and Business partners—regardless of the source, these instructors possess firsthand experience with the subject of the materials that they teach. And they tend to come equipped with not only the latest mySAP solution information, but also the most relevant.

Today, SAP is quite open to sharing its training resources and curriculum with all potential students, from SAP Technical Support Organization folks tasked with supporting their company's mySAP solution, to self-employed SAP consultants selling their services on the open market, to niche and Big 4 consulting partners. SAP vendors, hardware partners, and a number of systems integrators make up the balance of those attending SAP Consultant Education. As shown in Figure 9.2 and detailed in the following list, Consultant Education includes the following kinds of training and other education services:

- “Solution awareness” education, which is designed to teach students about the impact and capabilities characteristic of a new mySAP solution or related offering. This type of training appeals to many, but especially to senior-level project team members. I like to think of it as training for people who don’t know what they don’t know.
- Early product training and workshops offered either in a classroom environment or online. This represents the first-line training presented to teach others about new products, and is typically therefore more “marketing-oriented” than most of SAP’s other technical training approaches.
- SAP Academies represent the third level of Consultant Education, popular with technically deep SAP consultants tasked with completely understanding a product end-to-end. Characterized by intense knowledge transfer, SAP Academies build highly sought-after skills in their students.
- Certifications represent in many cases the culmination of SAP’s formal training efforts. Consultants certified in a particular discipline tend to stand out from

their non-certified colleagues when all other factors are about equal. SAP certifications also tend to communicate a certain level of core knowledge and expertise, which makes identifying and selecting the “best of the best” consulting resources easier than otherwise possible. Note that although certification is possible without the benefit of formal training, most consultants tend to achieve their certifications after either attending an SAP Academy or fulfilling a customized training agenda in any number of disciplines.

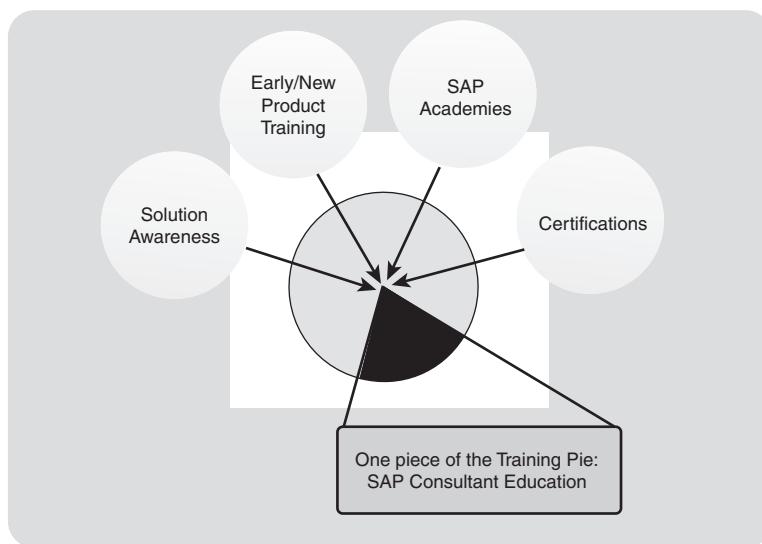


FIGURE 9.2 SAP Consultant Education is broken down into the four areas of Solution Awareness education, Early/New Product training, SAP Academies, and Certifications.

All of the formal courses offered by SAP AG are hands-on to a great extent, and many of the functional courses are quasi-technical as well (such as those where integration to another system is covered). You should know that the huge majority of formal classroom training administered by SAP today is functional configuration training, rather than pure technology-focused or technical/Basis training.

End-User Education

The third and final type of formal course offering provided by SAP is simply called *User Education*, and represents training administered to SAP end users—anyone who will access the productive mySAP.com solution in the normal course of conducting business. SAP certifications are achievable here as well.

End-user education is available in many functional areas and mySAP components, and is often administered on-site via workshops and similar customer-specific approaches, discussed next.

Onsite Training Workshops

Instead of forcing a customer to develop and deliver their own training materials, SAP AG introduced what it still refers to as Customer-Specific Training. These fundamental educational services are focused at team-wide training, geared to teaching the SAP Technical Support Organization and more what they need to know to make a new mySAP.com solution a productive reality. With more than 250 training courses available today, covering a comprehensive standard curriculum, custom training courses and workshops can be quickly developed to fit an organization's specific education requirements.

These workshops can be fine-tuned to cover all application, functional, and technical areas related to your SAP implementation. And the workshops can be hosted either at an SAP training facility or (with the right hardware and other infrastructure) at a customer's own site. Perhaps best of all, though, SAP understands what a customer is looking for in terms of training—they've implemented thousands of solutions, and they know what it takes to make it happen and keep them running. So if you simply provide SAP with the training topic, the location, the date, and the time, they can take care of everything else. And that lets SAP's customers refocus their energies on what they do best—their own unique core competencies, which are probably far from developing and delivering customized training courses!

Creating and Delivering Custom Training Curriculum

Despite what I said earlier, there are times when developing and presenting your own materials can still make a lot of sense. In looking back at some of the custom training I have personally created and either presented or left with my SAP customers, I am a bit surprised at the variety—there seems to be no particular area that is more common than another. Bottom line, though, these 3- to 10-day training engagements saved everyone on the project teams a lot of travel and training time, and were certainly cheaper than any alternatives. Why? Because, as a virtual project team member, I knew exactly what each client really needed—I knew where their support or Go-Live holes were biggest, so to speak, and I knew with what materials and information to fill these holes. Consider the following real-world training engagements in which I have participated:

- SAP Help Desk training—This customer needed “SAP Help Desk 101” training, and visibility into the resources they had at their disposal after a trouble-ticket was created (“who” to call both within the SAP TSO and externally). A variation of a document that I developed to use in help-desk troubleshooting can be found on the Planning CD.
- SAP Junior Basis training—Many of my clients have benefited from this hands-on and presentation-based training, where I walk through the SAP Solution Stack, preparing SAP infrastructure, performing mySAP installations, executing basic Transaction Codes (T-codes) used in support of performance monitoring, and so on.

- SAP Operations training—Here, I trained SAP operators in basic solution stack monitoring, from the server and SAN hardware layers, up to the OS, database, and applications. I also created custom operations checklists and process documents, similar to documents I have shared in Chapter 14.
- SAP Sizing training—At one site, the customer wanted to better understand the mySAP sizing process (especially as it related to the user/transactional data that can be captured in existing systems, to aid in sizing a new SAP solution). I taught similar workshops for third-party partners and vendors, too.
- SAP BW training—Similar to the Sizing training in the preceding point, I was also asked to share my experiences with designing and deploying SAP BW, including best practices for sizing the database and methods used to segregate what I call data warehousing “farmers” and “miners.”
- SAP Linux training—After attending a Train-the-Trainer course delivered by a good friend and former Compaq colleague, I shared my newfound insights and observations with various SAP/Linux prospects via subsequent training workshops.
- SAP MSCS Cluster training—This client requested a hands-on workshop demonstrating not only a SAP/MSCS installation, but also common design flaws, installation challenges, and post-installation clustered SAP/SQL Server troubleshooting issues.
- Monitoring Procedures/Approaches training—Here, the client wanted their Enterprise Management group to hit the ground running. We covered the stack, their specific enterprise tools, and how to monitor the system via SAP CCMS.

Another customer simply wanted assistance in developing a custom SAP Basis Training Plan for their SAP TSO. I have included a sample training plan on the Planning CD—it’s a bit dated, but the approach is still sound. I suggest using it as a template of sorts, to create your own custom SAP Infrastructure training plan if necessary.

All of these custom training courses had more in common than being cheap and effective. First, they served to teach those who would eventually turn around and teach others, too. That is, the materials I created and delivered included speaker notes and handout materials, such that they became “train-the-trainer” self-documenting courseware. And because each set of materials was always disseminated in Microsoft PowerPoint format, the training courses could be easily modified to reflect new conditions or changes in a particular customer’s environment (it is for the same reason that I share all of the figures in this book in Microsoft PowerPoint format—so that you can edit them to meet your own custom needs).

Second, in lieu of another person standing up in front of a room and talking for four or eight hours at a time, these course materials also served as basic orientation materials for new hires and other personnel tasked with supporting my particular customer's unique SAP Solution Stack.

Third, simply by virtue of leaving the materials with their respective teams, they served as a foundation for both current-state and how-to documentation. This allowed the writing curve to be shortened significantly for some of my customers, and in all cases helped each team to stay fresh themselves as the project progressed and different team members were pulled back and forth between their SAP-specific and potentially other job-related responsibilities.

In closing, custom training and workshops can prove extremely valuable simply because they tend to be pointed and customer-specific. Unlike formal offerings, they are not locked down to a explicit set of curricula.

NOTE

After I complete such an assignment, I like to leave a memento with my clients in the form of an unofficial "Training Certificate" or "Attendance Certificate"—this gives my students a sense of accomplishment much as a formal classroom certificate does, and can serve as a useful reminder of what we covered or learned during our time together. I've included a sample Training Certificate on the Planning CD that may be customized to fit your needs.

SAP Knowledge Warehouse

The SAP Knowledge Warehouse (SAP KW), first popular a few years ago, allows you to create and manage custom training materials. A "version management" feature supports country-specific, plant-specific, and role-specific training. This, combined with a modular approach to reusing training materials, encourages rapid development of training courses, self-tests, assessments, and even course-specific certifications. And SAP KW helps you get a jump start on both project-team and end-user-based training in two ways. First, it ships standard with the SAP Library (extended application-specific help), SAP Glossary, and SAP terminology. Second, more than 500 pre-developed fee-based standard training courses are at your disposal, ready to be customized for your specific needs. And because all of this is shipped in multiple languages and SAP release versions, SAP AG has made it pretty easy to support a mixed SAP "legacy" and mySAP.com enterprise.

SAP's Knowledge Warehouse also allows full-text searches, attribute-based searches, process and work-flow modeling, check-in and check-out of documents, and other basic document management features. All authoring is carried out using Microsoft Word and PowerPoint, tools common enough to most training developers today, thereby speeding up development. And it supports transporting documents in much

the same manner as development objects are promoted to test and then production systems.

A number of interfaces—browsers, WorkPlace, the WebGUI, and the Knowledge Workbench—make user access to KW-based training easy. And finally, it supports most of the file formats used in the last few years as well as today, including:

- DOC
- PPT
- PDF
- HTML
- XML
- AVI
- VCM

As if these were not enough, SAP Knowledge Warehouse also allows you to associate other file types to any editor application that might be used by a particular customer or organization.

All in all, SAP KW is an awesome training tool across the board, useful to the end-user community down to the SAP Technical Support Organization and even the project team supporting an SAP deployment. Things changed after the introduction of Enterprise Portal, though, and the “bar” was raised again. Many of the features and much of the functionality initially introduced in SAP WorkPlace and Knowledge Warehouse found a new and improved home in the Portal, as you’ll discover in the following section.

Using the Enterprise Portal for Knowledge Management

SAP’s Enterprise Portal offering takes us beyond where the standalone Knowledge Warehouse could ever go—it opens the door to managing distributed knowledge across not only your own enterprise resources, but potentially the resources of your technology and consulting partners, and even the Internet, as shown in Figure 9.3.

This is an infinitely “wider” and smarter approach to managing information. Figure 9.4 illustrates just how wide this approach is, depicting the range of knowledge resources available to Enterprise Portal, from company-internal Web/file servers, mySAP PLM content, and knowledge management repositories, out to partner information resources.

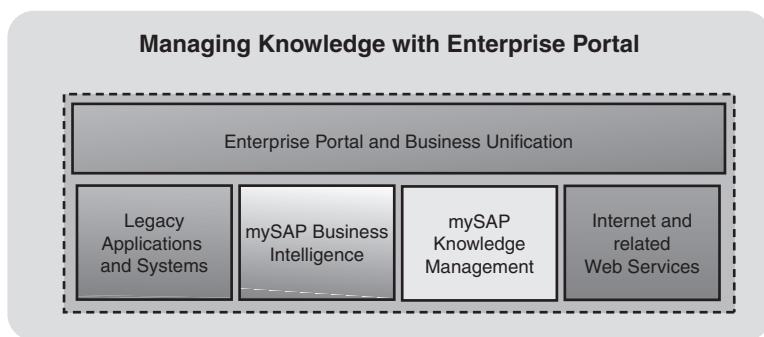


FIGURE 9.3 Enterprise Portal brings legacy applications, mySAP BI, mySAP KM/Knowledge Warehouse, and Web services all under one roof.

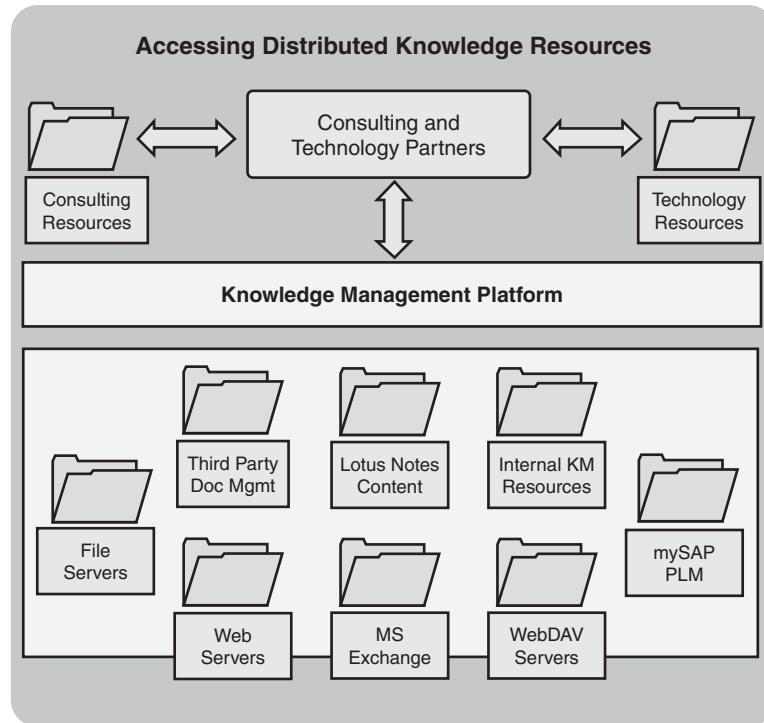


FIGURE 9.4 The reach of Enterprise Portal clearly shows the value that can be derived in support of end-user, consultant, and other types of training.

And Enterprise Portal takes content to the next level as well, integrating robust services like retrieval, classification, and collaboration with general content management capabilities. In effect, then, the Portal takes many of the organization, collection, and communication issues relevant to training out of the way, allowing people to instead focus on *learning*.

Other Computer-Based and Online Training

SAP was not always the knowledge management powerhouse it finds itself today. Before Enterprise Portal, and even before Knowledge Warehouse, back in the days of InfoDB and IDES, other companies were eager to fill a rather large training and knowledge management void. RWD Technologies, for example, brought to bear its extensive knowledge and understanding of SAP end-user training to create customized training courses unique to their customers. They were one of the first companies to offer compelling low-cost delivery options, too, like CD-ROM and online. RWD Technologies was so successful at this that they eventually became one of a select number of SAP Knowledge Warehouse partners.

Today, companies like RWD compete head-on with SAP's own Educational Services group when it comes to online training. SAP AG offers something it calls e-Learning, a training approach that is flexible enough to facilitate employee and project-team training both in the office or on the road. SAP describes this as "the ideal choice for a fast-moving work force" and positions it as a way of "ensuring" the success of a mySAP.com implementation by ensuring the success of the underlying training. Courses are introduced to a project team by way of their PCs or laptops, bridging the void between the classroom and the fast-paced workplace. And given that any client with a minimum-version browser can attend these training sessions, online SAP training is both easy to access and flexible, allowing the team to maximize their time, and work at their own pace, while avoiding travel costs, lost work time, and scheduling conflicts.

SAP TechEd and Similar Venues

I am a big fan of SAP's annual technical conference, SAP TechEd. Of course, SAP hosts a number of other annual conferences and events, but none of them are geared toward how to design, implement, integrate, and support mySAP solutions, as TechEd is.

TechEd is by far my favorite software-sponsored technology event primarily because it has not (and hopefully never will) succumb to the "marketing guys." Not that there's anything wrong with that, but unless I'm attending a future-product-based technical session, I prefer to skip the marketing hype. Just take a look at some of the following highlights from SAP TechEd 2002 alone, and you can begin to understand

why the technology guys skip SAPphire and ASUG (less technically focused annual SAP conferences) when travel and training budgets are tight:

- Hands-on workshops of between two and four hours each were presented. These alone are worth the trip. Everything is staged, ready to go—hardware, software, client/access devices, you name it. Plus, with supporting PowerPoint files and custom exercises and instructors that actually have used their products and know them inside and out (with a few exceptions), these activities represent the best ROI a technical consultant can obtain outside of actual experience.
- As I just mentioned, the instructors have real-world knowledge. And they tend to be technical consultants and other technology support personnel rather than “trainers.” Thus, what the overall presentations might lack in “polish” is more than made up for in substance.
- A nice “101” level course in mySAP platforms and solutions, backed up with deep exposure to every mySAP component on the market (or drawing board) today, was available. In other words, my colleagues and I were treated to technical discussions related to everything “hot” and everything out there today—Enterprise Portal, CRM, APO.
- Customer and partner presentations helped to round out the predominantly SAP-delivered technical sessions. These sessions are usually quite special to me in that they tend to focus or tie back to actual lessons learned—again, the real world.
- Business development played a part in my interest in SAP TechEd, too. Every year, my colleagues and I tend to walk away with a few consulting engagements. Finding and spending no-pressure time with an expert in a certain mySAP.com discipline has to be worth a lot in and of itself.
- Finally, for the marketing folks out there who absolutely needed to give away pens and shirts and reams of brochures, along with other garbage even my own otherwise undiscriminating kids found less than appealing, there are the vendor booths. The real value for me is wandering from booth to booth, relationship-building with a few key players (like SAP partners with whom I might be engaged on a project, for example).

The training, relationship-building, and technical insights received over the three and a half days at TechEd paid for themselves in a matter of weeks. And the comradery alone—seeing old friends and making new ones—is invaluable in our small world of SAP. But what TechEd offers that no other education medium offers is a chance to learn from the experts—the product developers, the designers and engineers, the customers and consultants who work with mySAP solutions day in and

day out. These people know things that can't be found on any Web site or in any book, even this one. Bottom line, I consider SAP TechEd a smart investment of my company's time and money, and expect you will, too.

Creating "Cookbooks" from Product Documentation and User Manuals

Sometimes I refer tongue-in-cheek to the need to read product documentation as "a last resort." Indeed, there are many consultants out there who shy away from product documentation and user/installation guides altogether until it is forced upon them. The reality, though, is that a really good SAP technical consultant or administrator will *start* with these resources, not somehow wind up there after wasting hours with an aborted installation. It is these latter consultants who "make it happen" day in and day out, while their cowboy colleagues occasionally get lucky but more often than not barely manage to scrape together an operational system.

It's sad that so many of us ultimately have to rely on the various read-me files, installation guides, and online help available for a product to actually get it installed. But it's an enterprise application's complexity that keeps many folks like myself employed to one extent or another—ignoring a 100-page installation guide ultimately wastes everyone's time. That is, trying to install a complicated mySAP.com component without the benefit of a checklist, guide, or "cookbook" is mindless.

Training is nice, and online courses are great, but at the end of the day we all still wind up with a stack of CDs and countless downloads waiting to be installed on a particular piece of hardware. For my team back at HP, this is certainly business as usual, and we are nothing if not comfortable with this fact. But we have a huge advantage over most of our customers—we have a multi-million dollar enterprise solutions lab and the bulk of the entire North American SAP Competency Center at our disposal. So, although we tend to have a lot of fun doing what we do, for the rest of the world I can see how a lack of knowledgeable engineers and the weight of project-driven timelines takes most of the fun out of new-product trial-and-error training.

Fortunately, SAP's documentation is really quite good. Not perfect, no—I spend a considerable amount of time identifying, circumventing, and documenting "gotchas" for my paying enterprise customers. But all things considered, SAP has come a long way from the days of thinly clad and imperfectly translated installation documents.

The following list solidifies much of what my colleagues and I go through to teach ourselves a new mySAP.com product or a supporting layer in the SAP Solution Stack—consider it a road map of sorts. I hope it proves useful to you, and shortens your workdays a bit:

- Visit <http://www.sap.com>, and search for the relevant topic, such as Web Application Server, CRM, or APO, and so on, picking up general knowledge of the application or component in which you are interested.
- Visit the <http://service.sap.com/instguides> Web site to locate relevant documentation. Although SAP's Instguides (Installation Guides) are mySAP component-specific, I find the most value in looking through their Master Guides, which provide the "big picture" in terms of solution architecture, role of the solution, and pointers to more detailed technical resources.
- You may also access SAP's "SAP Library" at <http://help.sap.com>, to obtain deeper knowledge in a particular area.
- Visit <http://service.sap.com/swcenter>, and enter your SAP Service Marketplace (the successor to OSS, SAP AG's Online Support System) ID and password, to download actual mySAP.com components and supporting software/utilities. You must register with SAP AG, and provide them your customer number or partner data to gain access to this and many other SAP Web sites.
- After you've reviewed the Master Guide for your particular mySAP solution, I suggest carefully reading through the Install Guide(s), looking for specific SAP notes and other notes to provide clarification.
- Visit <http://service.sap.com/notes>, enter your SAP Service Marketplace ID and password, download each of the SAP Notes mentioned in the installation guide(s), and read these carefully.
- Consistent with the recommendations found in the Install Guide, configure a server and storage system that meets the minimum installation requirements.
- Finally, follow the Install Guide and walk through an actual installation, being careful to perform every step. Note the "gotchas" and other problems/issues that occur. The best way to accomplish this is to create a document of your own that references the Install Guide step-by-step (avoid using page numbers, as these numbers change both over time and based on unique printer setups). I call this my "Delta Guide" to the Install Guide. Delta Guides are covered at length in Chapter 11.
- Now, using both the Install Guide and your own custom delta document, go through the entire process again, ensuring that everything needed to perform the installation is indeed covered between the Install Guide (perhaps 90%) and your own document (the remaining 10%).
- It's important to document the hardware and OS configuration in this custom delta document, too (I like to keep all of my notes and supporting documents in a single umbrella document). I include everything relevant to the SAP Solution Stack—system board and hardware component firmware revisions, Operating System patches and Service Packs, and so on up through the stack.

- To discover more about how to create an effective Delta Guide, see “Custom Checklists and Recipes,” p. 391 in Chapter 11.

By adhering to the process described in the preceding list, I help ensure that an installation is indeed repeatable, and that others who follow me in performing this installation do not experience the same hold-ups and issues that I encountered and eventually solved. All of the steps described in the preceding list are therefore important. But the steps that save the most time are probably the first few—quickly reviewing a high-level Master Guide and then a detailed installation guide, followed by obtaining the appropriate software, gives you a huge advantage in terms of minimizing training time.

And the last few steps are important as well, for it is these tasks that allow us to create a “cookbook” for a particular solution stack, such that we can run through subsequent installation processes quickly and efficiently, over and over again. As we see in Figure 9.5, these cookbooks create a custom knowledge base over time, serving a host of needs.

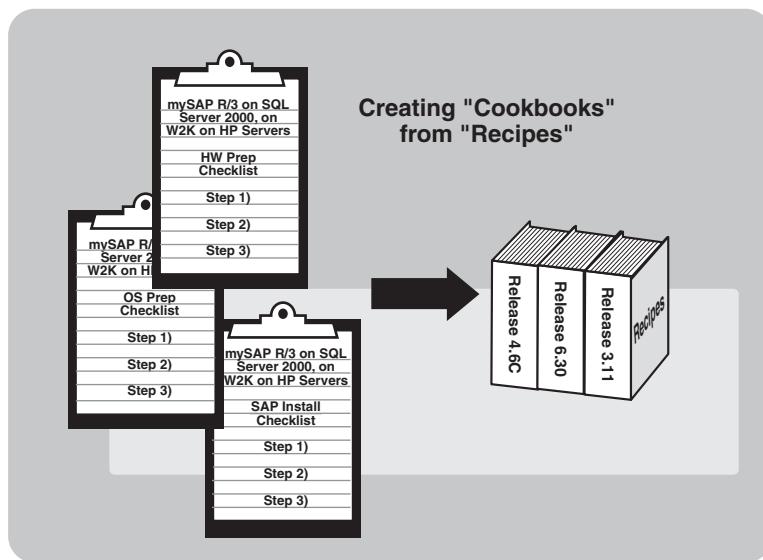


FIGURE 9.5 By creating and maintaining a suite of custom cookbooks for each mySAP.com installation routine, you benefit from a repeatable and rapid system installation process.

So in the end, although formal training courses and custom workshops are great ways to obtain training, many technical professionals learn the basics of doing their jobs by following the same process I outlined in the preceding list.

Feedback Loops—Improving the Value of Training

Feedback loops not only improve the quality of training delivered over time, but they can actually help *prove* that the training being delivered is effective, timely, appropriate, complete, and accurate. The best way that I have found to create a really useful feedback loop is to determine who a particular organization's customer is, and then to talk to or otherwise obtain feedback from that customer. For example, to improve the value of end-user training in a particular functional area, it only makes sense that training feedback needs to originate from the people receiving the most benefit from the training—the end users.

It gets more complicated, however, as support organizations grow larger and therefore the “customer” lines grow broader, or get fuzzy. For example, identifying the *real* customers of a particular subgroup within the Technical Support Organization can be especially challenging. But ensuring that the SAP TSO is trained well enough to deliver on all of its commitments is vital to the success of a mySAP.com project, without exception. Thus, it’s imperative that each subgroup within the SAP TSO recognize its customers, with the understanding of course that everyone is ultimately working for the benefit of the SAP end-user community.

Functional and Development Consultants

The functional and development teams responsible for configuring a mySAP.com solution have one obvious customer—the end user who will log in to SAP one day in the future and execute the business processes designed and implemented by this team. However, that day—Go-Live—lies far out in the future. Meanwhile, the expertise gained as a result of any specific training that the functional/development teams possess *today* needs to be ratified and probably tweaked throughout the life of an SAP implementation.

Who is the functional/development team’s customer today, then, before Go-live? In the simplest sense, it’s the power users and other folks tasked with reviewing and using the code, and ultimately proving that the code being written works. Their regular and impartial feedback is essential to improving the coding and configuration upon which the system relies. This in turn should drive what kind of training the functional and development teams receive, and help to prioritize this training as well.

SAP Technical Consultants

What about the various technical consultants and other team members tasked with designing, installing, integrating, and supporting SAP both before and after Go-Live? Who is *their* customer? Bear in mind the following:

- SAP Basis and other mySAP-specific implementation specialists are charged with assembling the system to be configured. Thus, I look at the functional and development folks as their customer.
- mySAP administrators are tasked with keeping the system available and well-tuned while under development, and later while in production—their customer is first the functional/development team, and later the end-user community.
- Database administrators are asked to maintain a well-performing and highly reliable database foundation, secure from corruption and resistant to tampering. As the database serves the SAP instance, so too should the database administrator serve the SAP Basis or SAP Infrastructure organization within the SAP TSO.
- mySAP integration specialists work with other integration specialists and the SAP Basis group in general to ensure that a mySAP solution operates correctly at a technical level. But they also play a role in ensuring that the integration points provide the required functional integration as well. For these reasons, the functional teams of the core system as well as other mySAP and legacy systems represent customers to these integration specialists, as do the SAP Basis and similar technical teams supporting the technical foundation.

Timely and pointed feedback from each of the customer groups just identified will go a long way in steering training dollars down the right path—where it's needed most. Not all SAP TSO roles fit well into the groups described in the preceding list, though, and have instead been loosely grouped into a general *SAP Infrastructure* organization, covered next.

Other SAP Infrastructure Roles

Data Center Infrastructure teams, staffed with people who oversee the power, rack, and network infrastructure, serve a whole slew of customers. But from a “chain of support” perspective, I believe that their immediate customer is the Hardware team. This makes sense in a lot of respects—see Figure 9.6 for an illustrated chain of support.

The SAP Security team is another subgroup of the SAP TSO that serves many customers while providing an essential and core infrastructure service. Although it could be argued that their primary customer is the SAP Basis or even the Development teams, I'm convinced that the end-user community holds the greatest mind share as to how well the security team is performing. If the security team does not do their job well, they will make life miserable for the end users. On the other hand, if they do their job exceptionally well, the end users will not even remember that there *is* a security team!

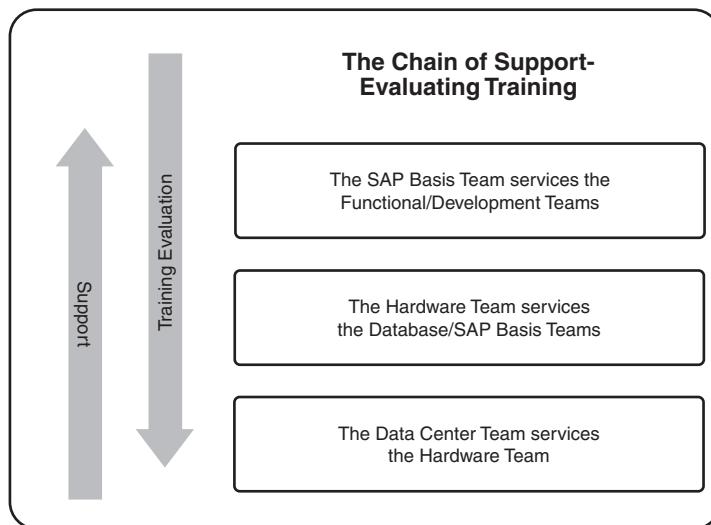


FIGURE 9.6 The chain of support works to promote customer-consumer relationships, all of which ultimately serve the SAP end-user community.

SAP Operations/Help Desk

The SAP Operations team may be deployed as a subset of the Data Center Infrastructure team, a general hardware support team, or even the SAP Basis organization. But they are accountable to the same primary “customer,” nonetheless—the Basis group, charged with providing a highly available computing environment.

On the other hand, the SAP Help Desk’s customer is quite clear. The Help Desk and its support folks service the end-user community directly. In terms of feedback to guide how training dollars are spent, though, the Basis team will be in an excellent position in this regard. Why? Because many of the issues not resolved by the Help Desk will naturally be escalated to the Basis team.

Additional SAP Support Specialists

Other support specialists play important implementation roles, too. But by the nature of their positions, it can be very easy to determine their customers. For example, the deliverables created by documentation specialists should be evaluated by whomever the documents are created for. Training specialists should similarly be evaluated by their students. Performance tuning specialists assisting the Database group should be evaluated by the DBAs, and so on. In this way, the combination of training and experience brought to bear in a particular situation is evaluated by the people in the best position to evaluate it, the customers.

In conclusion then, as we are reminded in Figure 9.7, the real customers serviced by each of the subgroups covered over the last few pages vary considerably. It's my hope that the matrix in Figure 9.7 will help those tasked with managing the quality of training to focus on the proper customer organizations—the groups best positioned to provide really valuable training feedback, and therefore improve the overall SAP implementation across the board both before and after Go-Live.

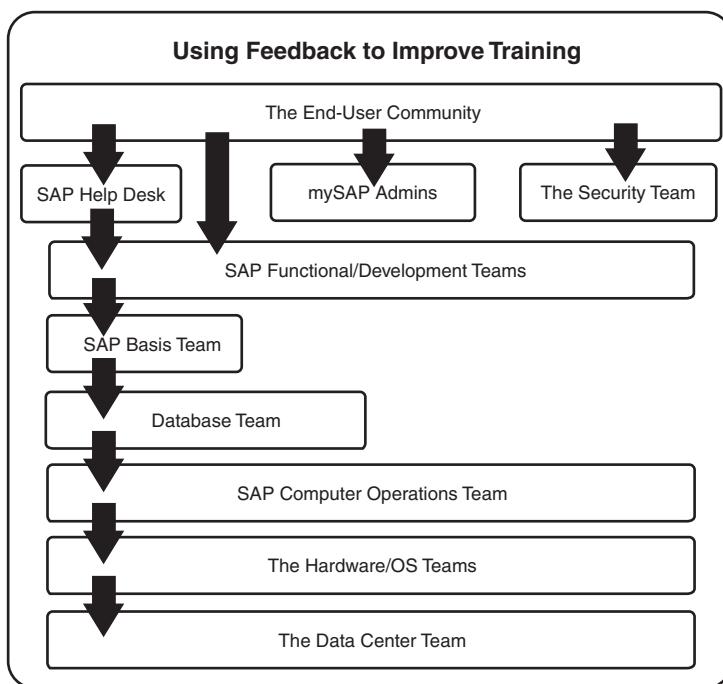


FIGURE 9.7 The quality of training received by one group is easily evaluated by a different group—this constitutes the foundation of my “feedback loop” approach to improving training.

Certification Programs

The last method of feedback and evaluation that I cover is certification. SAP's Educational Services, and other SAP Solution Stack vendors, all offer certification programs that not only reflect competence in a particular discipline or product, but also act as excellent feedback. A certification tells others that the holder has achieved a certain level of competency. The certification closes the loop, so to speak, regardless of whether the knowledge being tested was gained through formal training, hands-on experience, self-study, or other methods.

The value of training is nicely captured in process or task-oriented exams, like those offered by SAP, Microsoft, Oracle, HP, and others. This is why so many companies offer certification programs for their products—they help customers to select people who are qualified or well-positioned to be successful in designing, installing, or supporting their products. And certified individuals act as an extension of the company's sales and technical force, too, typically promoting the products with which they are most familiar and certified.

SAP offers a number of certifications, none of which absolutely requires formal classroom training—all of these certifications can be obtained as long as the test-taker has adequate knowledge and experience:

- Application Consultant, which is specific to certain SAP solutions.
- Technical Consultant, which is focused on core SAP Basis, SAP system administration, database administration, and other technology elements and layers.
- Certified SAP User, which can not only prove your proficiency in using SAP, but as SAP AG says “enhance your employability,” too.

Certification exams are scheduled and given in several major cities worldwide. The tests are very comprehensive, and therefore require quite a bit of preparation to pass. But the results can be impressive in terms of opening doors where none were open previously.

With the SAP Technical Organization trained and ready to deliver, we may now turn our attention to Chapter 10 and the SAP Data Center—it's time to get to work!

Tools and Techniques

A sample technical training plan and sample training certificate may be found on the Planning CD, as can a SAP Help Desk 101 document useful in discussing common troubleshooting scenarios. Two sample “Table of Contents” outlining the goals of an SAP Computer Operations workshop and a SAPGUI training session are available here as well. Other sample documents, like end-user training guides, are also on the CD—use these as templates in support of your own training endeavors.

I've also included a “Skillset Matrix” in Microsoft Excel format, and a PowerPoint presentation regarding how to create useful documentation. And as always, all of the figures illustrated in this chapter are also on the CD, in Microsoft PowerPoint format.

Summary

In Chapter 9, I identified, clarified, and described the immeasurable role that training has on an SAP project's success. I then looked closely at the different subgroups

within the SAP Technical Support Organization, and how these groups benefit from training directly supported by a particular system within the SAP system landscape, as well as by various training approaches. Proving that the training received by a particular group indeed met our needs for quality and completeness wrapped up the chapter, positioning us to advance to the next phase in our SAP implementation—SAP Data Center deployment.

10

Developing the SAP Data Center

Introducing the SAP Data Center

In this chapter, I will discuss in detail the process inherent in building a new data center facility, or transforming your current data center into a foundation capable of supporting a mission-critical enterprise SAP application and its requisite solution stack. The goal is clear—to create a stable and highly available facility for hosting your SAP implementation. I take this process from building out the power, network, and rack infrastructure through installing servers, configuring disk subsystems, and installing server operating systems. In essence, by the time you have completed the activities described in this chapter, the SAP system landscape should be ready for the SAP Basis installations to be performed.

Although you may have already decided on many of the factors that will drive the design and deployment of this facility, including location, hardware and software vendors, and SAP products and components, “availability” should typically represent the single most important consideration driving the data center build-out. Remember, this application and its resident data will ultimately prove critical to your company’s well-being. Should this application and data become unavailable, even for a short period of time, costly and otherwise huge ramifications could result:

- Thousands of users may sit idly by, waiting for the system to come “back.”
- Trucks may stack up in the loading docks, waiting for bills of lading and shipping orders.

IN THIS CHAPTER

- Introducing the SAP Data Center
- The SAP Data Center “Big Picture”
- Data Center Physical Requirements
- Rack Planning for Data Center Resources
- Network Infrastructure for SAP
- Network Server Preparation
- General Storage Considerations
- On the Road to Implementation
- Tools and Techniques

- Customers may call in or “click in” looking for status updates on their orders, only to be told to “try again sometime later, the system is down.”
- Manual processes may need to be invoked to keep new orders coming in. And to make it worse, eventually these manual orders will need to be keyed into the system when it again becomes available, further impacting the users’ time to place new orders.
- Reports will be unavailable, impacting decision making from the boardroom down to the assembly line, and everywhere in between.

► For more details on the impact that high availability plays in the SAP Data Center, see “A Final Look at the SAP System Landscape,” p. 217 in Chapter 6.

So, keeping “availability” uppermost in our minds, let’s move on, noting that the data center in some way affects all layers of the SAP Solution Stack. For example:

- At the lowest layer of the stack we find power requirements. I have worked on a number of mySAP.com implementations where as many as 80 new servers and related disk resources are deployed over the course of a year, for example. Such a formidable collection of hardware pulls a considerable amount of amps and volts, not to mention the raw power infrastructure requirements needed to simply keep all of this gear running. An incorrectly architected power infrastructure will bring down an otherwise highly available clustered SAP solution in a heartbeat—all of the high-availability offerings at other layers in the solution are “powerless” without a well-architected Uninterruptible Power Supply (UPS) solution or power distribution system.
- Similarly, cooling requirements must be addressed at one of the lowest layers of the stack. Those same 80 servers I mentioned not only pull a tremendous amount of power, they also generate a considerable quantity of BTUs (British Thermal Units, or units of heat). An inadequate cooling and air handling system can wreak havoc with availability statistics.
- Servers, disk subsystems, network infrastructure, and other SAP-related infrastructure needs to all be neatly racked and cabled. Lack of attention to detail, poor planning, and more can quickly become contributing downtime factors. In more than one case, I have seen racks of servers cabled neatly and in an organized manner, only to have all of this work ripped out after someone finally thought to pull out a server for servicing. Why? Because the cabling did not allow enough “slack” for the servers to be actually pulled out more than a few inches! In other cases, I have seen high availability compromised simply because otherwise redundant pairs of cables were routed through the same cable conduits, and the conduit itself failed or was damaged.

- SAP's tiered architecture requires forethought in regard to network planning. As you will see later in this chapter, neglecting the impact that a three-tiered architecture can have on public and private network segments can impact not only availability, but also overall performance. In addition, we cannot forget about network firewall security and other network infrastructure considerations that will impact availability when it comes to Internet-facing SAP Web servers, eCommerce/procurement systems, and so on.
- Server infrastructure and design directly impact availability, both "in the box" via single points of failure, and "out of the box" when it comes to higher-layer solution stack matters like failover clustering.
- Your disk subsystem, whether direct-attached SCSI, fibre channel arbitrated loop, switched fabric, or network attached, tends to be the single most important performance factor in my experience, outside of really bad coding. But it is also one of the most easily misunderstood solution components when it comes to high availability.
- Operating system installation, configuration practices, and more can easily impact availability. I will note how and why this seems to be so often overlooked, and help you to conform to best practices in this regard.

As you have just seen, the design and implementation of a company's specific SAP Solution Stack through a well-planned data center deployment affects availability at all levels of the stack. Single points of failure (SPOFs) abound everywhere. A lone power source, single power distribution unit, nonredundant power supplies, single network segment to a server, single server hosting a database, and more all represent opportunities for downtime. And I have not even begun to discuss the database and SAP application layers!

THE SAP SOLUTION STACK OR THE OSI MODEL?

If you prefer to look at things from an OSI model perspective, you'll be happy to note that many of the key layers of the SAP Solution Stack map nicely to a layer in the OSI model. Therefore, every aspect of planning for a highly available SAP Data Center also tends to "snap into" one of those seven OSI layers, from the physical layer—power, through the data, network, and transport layers, up to the session, presentation, and application layers—where your end users are provided with the SAPGUI interface.

In the remainder of the chapter, I will take a closer look at each of the availability factors in the bulleted list provided earlier, as well as operational and other processes that will ultimately impact the net availability of your SAP solution to your end-user population.

The SAP Data Center “Big Picture”

From a “big picture” perspective, you need to ensure that no detail is overlooked when planning for your SAP Data Center facility or facilities. Availability, after all, is all about details. Each layer in the SAP Solution Stack presents you with single-point-of-failure challenges; each layer must therefore be thoughtfully considered with an eye toward eliminating these SPOFs, or at least noting and mitigating risk to the point where such risk becomes financially acceptable.

First Things First—Standardization

Before we dive into identifying and addressing single points of failure common to SAP Data Center designs and implementation, it first makes sense to discuss the role of standards and standardization in your data center. Standards impact everything. Fortunately, taking a close look at standards early in the SAP Data Center planning process forces you to think ahead, and ultimately avoid many pitfalls potentially lurking in the future. Consider the following:

- Server naming conventions must be descriptive enough to promote manageability, but short enough to be technically supported by the particular SAP component version (and any other applications that might need to reference this name, including systems management applications like HP Openview or BMC Patrol).
- IP naming conventions should help identify what type of server network connection is being made. For example, standards that help to identify public, private, clustered, and internal/other network connections are quite prevalent in the world of SAP infrastructure.
- Disk naming (and drive letter, for Windows 2000/NT) conventions should be published and leveraged for consistency. Such consistency by its very nature impacts availability as well, as the chance of someone “accidentally” bringing a disk resource offline is less likely to occur when the disk name or disk drive letter for the database (for example) tends to be the same throughout the SAP landscape.
- Even something as simple as color-coding network cables and power cables can improve system availability. Similar to IP naming conventions, a good color-coding scheme helps avoid unplanned downtime due to inadvertently uncabling or miscabling a vital network or power connection. I have witnessed companies leveraging factors other than simply color, too; the number and thickness of bands in a cable, and even the cable thickness itself can also be used to differentiate otherwise similar cables.
- Standard “high-availability server configurations” are normal in most SAP shops. This usually equates to servers configured with redundant power

supplies, fans, power distribution units, processors, RAID-protected disk drives, RAID or ECC-protected RAM, network cards, and so on, in a server cluster configuration with dual-host bus adapters in each server.

- As with standard highly available server configurations, most SAP shops also promote an equivalent “high-availability disk subsystem.” This usually involves a standard frame or disk chassis, standard drive size (in example, 36GB 15,000 RPM 1” drives), standard redundant disk drive controllers, and redundant disk interconnects back to the server infrastructure.
- Next, a standard operating system built for high-availability deployments is typically developed. This would include specific OS release levels, patch or service pack levels, any patches or bug-fixes required, other software drivers and their versions, and so on. Nowadays, standard methodologies for deploying customary server images are often employed, leveraging OS-build approaches ranging from traditional disk imaging to custom scripting, deployment of imaging servers, hardware vendor-specific approaches, and so on.
- Finally, standardized processes regarding managing all of the aforementioned resources help to minimize downtime across the board.

Other standards exist, of course, but the preceding list should prove useful in identifying the key areas within each layer of the SAP Solutions Stack that must be addressed before a single data center floor-tile is ever pulled up, or server mounted, or operating system installed.

Data Center Physical Requirements

Physical requirements mean different things to different people. In the context of building your data center, I could go into minute details regarding actual construction of the facility, for example. In such case, physical requirements would exist with regard to the following:

- Physical construction materials and related factors (fire codes, weight-bearing members for roof-mounted AC/Environmental Units, load-rating factors for the raised floor construction, cable risers or trays, and so on).
- Location (avoid the first floor or top floor of buildings, and avoid flood zones, areas where access may be limited in terms of single stairs, elevators, roads, and so on).
- Environmental systems (cooling, heating, humidity levels), including access to published thermal specifications for each component to be housed in the data center.

- Accessibility (loading dock/freight elevator access, as well as double-protected public access points).
- Physical security/access, including monitoring systems (card or other systems for doors and window access, and attention to vertical security above the dropped ceiling and below the raised floors).
- Controlling systems (temperature/fire suppression, smoke, water sensors, and so on).
- Lighting and plumbing, as required.
- A central operations/monitoring station.
- Access to high-bandwidth multi-path data communications circuits or network/Internet connections.
- General dual power infrastructure, in terms of 208 volts AC versus other options, including the availability of generators, battery backup, and so on. This can also include access to two discrete city or state power grids, should high-availability or disaster recovery requirements dictate such a robust power infrastructure.

These details are best left up to the experts who design and build data center facilities. Ensure that at a minimum the preceding points are addressed, however.

Power Requirements

Power problems can plague an otherwise bullet-proof solution architecture and therefore power needs to be planned for the long term, not merely for the demands of Go-Live—your SAP environment will grow, grow, grow. When addressing the power needs of the SAP data center, it is helpful to first analyze each specific server, disk subsystem, network, or other *hardware* component requiring power, and then work “back” to the ultimate power source. For maximum availability, ensure the following:

- Where the highest levels of availability are necessary, each hardware component must support redundant power supplies (otherwise, the remainder of this list is not of any use). Preferably, these power supplies should also be “hot swappable” or “hot pluggable.” In this way, in the event of a power failure, not only would the server remain available and powered up on its second power supply, but the failed power supply could be pulled out and replaced without incurring downtime.
- Each power supply alone should be capable of keeping the hardware component up and running. That is, if the average load being pulled from one of the power supplies in a dual-power supply configuration exceeds 50% of its rated

capacity, you actually don't have protection from failure of the other power supply! The alternatives are clear—lower the capacity requirements by reducing the number of disk drives, CPUs, and so on, or increase the number of power supplies to three or more, or in rare cases simply replace the current power supplies with higher-rated alternatives.

- Each power supply must have its own unique power cable. This is a very common oversight with some of the second-tier server and disk subsystem vendors, where highly available systems might be touted, but reality differs. These high-availability wannabes often provide only a single power cable receptacle even in their “redundant” power supply configurations. A single *anything* represents a single point of failure, and should be therefore avoided. Besides, I have actually seen a couple of power cables fail in the real world. As silly or unlikely as that sounds, it happens. And besides, even more likely is the potential to pull out a single power cable, effectively bringing down the most available server or disk subsystem.
- As I indicated previously, color-coding or otherwise differentiating power cables makes it very clear to everyone when things are cabled correctly. The most common implementation of this involves a black cable cabled to the primary power supply, and a gray or white cable cabled to the redundant power supply.
- Each power cable must be routed to dedicated separate power distribution units (PDUs, or power strips, and so on)—whatever is used by the company to centralize many power feeds into fewer larger-capacity connections. Each PDU needs to be analyzed to ensure that the load placed on this single component, should the other PDU fail, can still be addressed by the remaining PDU. And as I said earlier, the most common implementation of this is black cables to one PDU, and gray or white cables to the redundant PDU.
- Each PDU must in turn be cabled to redundant uninterruptible power supplies, or UPSes. Like the PDUs, these need to be regularly tested and analyzed to ensure that they are indeed “redundant.” Note that a UPS tends to only be equipped to handle short-term power losses, thus necessitating our next power component.
- Primary power for each redundant power supply should culminate in redundancy at the breaker boxes as well. That is, each power supply should ultimately receive its power from a dedicated breaker panel, like that illustrated in Figure 10.1.
- The “back-up generator” is a necessity for mission-critical SAP shops. Whereas the UPS provides short-term relief from blackouts and brownouts, a generator can conceivably provide power for days, as long as fuel is available. Select a generator that runs on whatever is most easily accessible or readily available, including diesel fuel, propane, or natural gas.

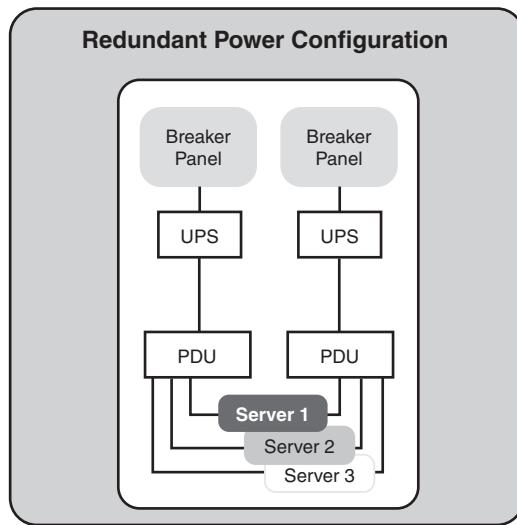


FIGURE 10.1 Here you see a completely redundant power infrastructure, from the servers/hardware components all the way back to the breaker panels.

It is of utmost importance that the generator and the UPS be properly sized to handle the loads placed upon them. Generators must leverage an Automatic Transfer Switch (ATS) to allow them to tie into both the power company (the power utility, or “utility power”) and the SAP data center, as you see in Figure 10.2. Critical systems need to be identified and earmarked for generator backup. These systems typically include emergency lighting, emergency environmental and safety controls, the critical SAP data center gear, and in some cases the contents of the entire data center.

Best practices also dictate that critical computing systems be isolated from main facility/operational power, but that each source of power “back up” the other. In this way, operational failures do not impact the enterprise system, and vice versa, and both are protected from failure by two power sources.

USING KVA FOR ACCURATE UPS SIZING

UPSES are rated by KVA or Kilo Volt-Amps. The formula to calculate KVA is $\text{amps} \times \text{volts} / 1000 = \text{KVA}$. So if your rack is capable of pulling $69.5 \text{ amps} \times 240 \text{ volts} = 16680 / 1000 = 16.6 \text{ KVA}$. Never allow your UPS to run above 80% capacity. 16.8 KVA is 80% of 21 KVA. So at a minimum, the rack should have 21 KVA worth of UPS.

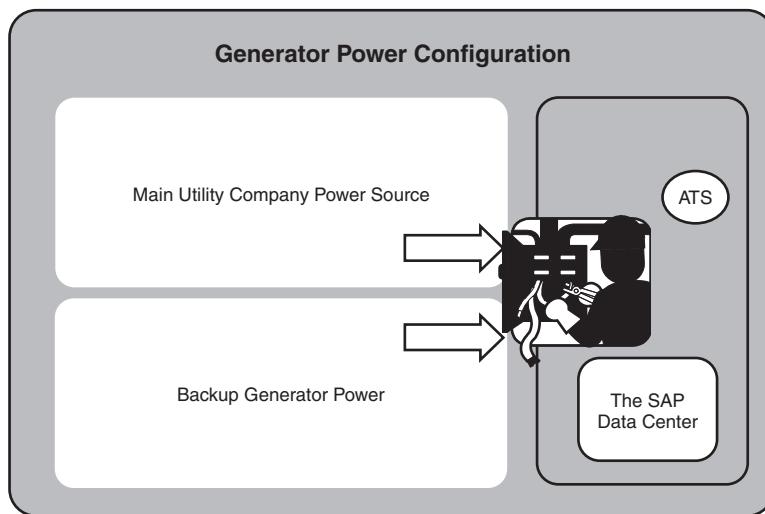


FIGURE 10.2 Without an Automatic Transfer Switch, the actual usefulness of a backup generator is questionable, thereby impacting high availability.

Power Oversight in the Real World

One of my favorite enterprise SAP servers, the HP ProLiant, serves as my next example. A wealth of information is available on the ProLiant, in the form of something HP calls “quickspecs.” These technical specifications have been published and updated for years, and describe in great detail much of the minutia of little interest to anyone but hardcore techies once the SAP system landscape is in place. Prior to that time, though, these quickspecs fulfill a number of critical roles. First, quickly perusing this document reveals that the ProLiant DL760 8-CPU server will draw a maximum of 10 amps on a 208- to 240-volt line, while producing a moderate 5309 BTUs every hour. These little tidbits of information will help ensure that the data center facilities folks understand how many and what type of power circuits to run. The BTUs, on the other hand, should be fed into a simple model that will determine the minimum rating of the air handlers (cooling/heating system).

Another bit of information is provided in the quickspecs as well, the power plug connector. As with most servers, the power connector from the server to the PDU or UPS is ordered as an option with the PDU or UPS. The real challenge then becomes matching the PDU’s power plug with the appropriate power receptacle. Typically, either an L6-20P or L6-30P is called for—in the case of certain PDUs deployed at one of my particular customer’s new SAP data center sites, the L6-30P (a 30-amp circuit) was specified by the quickspecs.

However, the customer got ahead of themselves and in the interest of meeting deadlines for power, laid enough L6-30P power cables for the new Storage Area Network, too, which was to arrive shortly after the servers. When the SAN cabinets showed up, though, they couldn't plug them in. The connectors were different—they required the L6-20P receptacles. To this day my colleagues and I are still not quite sure how our motivated customer managed this, but they actually forced their million-dollar SAN power cables into the wrong receptacles, and ran the system this way for perhaps a week before someone noticed that “something just didn’t look quite right” under the subfloor. Understand that these two connectors are quite different, and this little engineering feat will begin to sink in—they somehow managed to squeeze and twist the male plugs most of the way into the female receptacles. Not only did this pose a potential safety hazard, but they also risked blowing up their 20-amp SAN gear with 30 amps of juice.

They got lucky. The SAN had very few drives actually installed and spinning (and therefore drawing power) that first week. However, this simple oversight caused a one-week delay in the project plan, while the system was effectively shut down awaiting the proper wiring. In the end, the lost time was made up in the operating system and SAP basis installations, and neither gear nor people were any the worse. But the moral of my little story should be clear. Not only should the technical specifications for each piece of equipment be checked and verified, but we should never solve our power problems by brute force. Besides, because all of this information is just a click or quick-spec away, there’s really little excuse for misconfiguring or underallocating power and cooling resources.

Another common mistake illustrates how the redundancy of power-related components can be rendered useless through lack of attention to cabling and overall power architecture. My customer in this instance wanted to factor in redundancy at the physical layer of their SAP deployment. Their high-end servers, disk subsystem, and network equipment all supported redundant power supplies, so they took advantage of them. Each power supply on the back *left* side of each server and disk subsystem drive shelf was the recipient of a black power supply cable. This in turn was carefully routed along the left side of the rack enclosures to a power distribution unit dedicated for this purpose. Similarly, each power supply located on the back *right* side was fitted with a gray power cable, and these gray cables were also carefully routed to their own PDU. So far, so good—no single points of failure existed, in that half of the power supplies, cables, and PDUs could fail, and the system could still remain up and powered.

However, all of the careful preparation and planning that went into this phase of the project was tossed out the window after another few minutes of work. My customer plugged both PDUs into the same UPS, which then was cabled to two large redundant data center UPSes. Like the 1975 Chevrolet Corvette’s exhaust system, a dual-power approach to high availability is just “smoke and mirrors” after everything

merges into a common pipe. The Corvette never realized its peak power potential that year, and my customer lost any hopes of achieving 100% reliability, even though the solution “looked good” from many angles. By plugging the PDUs into the same UPS, they defeated the purpose of redundant power—if their single-point-of-failure UPS were to fail, the entire system would grind to a halt.

Like power, the next layer in the solution stack also represents a basic necessity for supporting your SAP enterprise—cooling.

Cooling and Other Environmental Controls

One of the biggest causes of hardware component failure is heat. Luckily, planning for cooling requirements has become a lot easier with the popularity of the World Wide Web. That is, nearly every hardware vendor out there today publishes BTU/thermal specifications. Your job is then to simply pull down and “add up” these technical specifications on every piece of equipment you plan to deploy. Don’t forget to allow for future growth, either—with server and disk form-factors shrinking every year, the heat generated per cubic foot of data center space just continues to grow and grow. To conservatively address the next three years in your data center planning efforts, determine the average BTU output per cubic foot, and then double that number and apply it to any remaining floor space that could conceivably house incremental SAP gear. Don’t forget to factor in the fact that air might not move uniformly through your data center and any other cooling dynamics inherent to your facility. In doing so, you will be eminently ready for the day the VP of Operations says, “Hey! We’re gonna go ahead with that SAP PLM project, so make room for 20 new servers and a couple of SAN cabinets in the next few weeks.”

As with cooling, air handlers exist that allow for controlling and exhausting heat, monitoring and fine-tuning humidity, and so on. Ensure that new hardware additions to the SAP data center are plugged into the BTU model as soon as possible, so that any new requirements for cooling will be given an opportunity to be addressed. Air handlers and other large environmental gear like this require significant lead times when new components or upgrades/replacements loom in the future.

Don’t forget to load the proper OS-drivers or applets that may be required by your hardware system to shut itself down in the event of overheating or loss of cooling. In the case of the ProLiant, the Compaq System Shutdown Service shuts down the server when the heat exceeds a predefined threshold, acting in response to commands from the integrated management features inherent to the ProLiant server platform.

And consider some of the newer trends in air handling and monitoring. For example, HP recently developed a robot that literally rolls around your data center floor looking for hot spots. Upon finding a hot spot, the robot analyzes the conditions and may, for example, signal your cooling system to increase airflow to the

area. Or it may instead communicate with your hardware systems to relocate workloads from one system to another. Utilizing a combination of these approaches, HP believes that it can reduce cooling costs for its customers in the neighborhood of 25 percent.

Rack Planning for Data Center Resources

With your highly available power infrastructures laying the foundation for your SAP Data Center, and attention to environmental requirements already addressed, you are ready to proceed with the next “physical” data center infrastructure layer—the rack mounting systems for housing all of our enterprise computing gear. In this section, I will cover:

- Rack layout and design considerations
- Optimizing rack “real estate”
- Rack mounting and related best practices
- Cabling and cable management
- Rack placement in the data center

What exactly is a rack? One of my customers describes a *rack* simply as “furniture for computers.” In most cases, racks are somewhere between four and seven feet tall, 19 inches wide, and something like 34–39 inches deep, depending on requirements. Racks allow for computer gear—servers, disk subsystems, network components and more—to be easily mounted, cooled, and serviced in what amounts to a “stacked” configuration. That is, the gear appears to be stacked one on top of the other. In reality, of course, none of it should literally be stacked one on top of the other, as this makes servicing and cooling the gear quite difficult.

Often, each component is mounted on sliding rails (though less advantageous fixed rails are still quite popular with some server and rack vendors). These sliding rails facilitate rapid access to the top and sides of each hardware component, where service areas are easily accessed. And something called *cable management arms* make this serviceability possible, allowing hardware systems to be pulled out for service without requiring cables to be disconnected or rerouted. Cabling is discussed later in this chapter.

Rack Layout and Design Considerations

Before you order a truckload of racks, you need to step back and develop a plan for deploying and laying out these racks. There are a number of best practices to be considered, but nearly as important is achieving some sort of consistency in deployment. This can be accomplished by working with your hardware vendors, and

demanding detailed deployment guidance as it relates to how servers and disk resources should be mounted, how many racks are required, how they should be optioned, and so forth. And there are questions related to rack placement in terms of metrics and standards that must be answered:

- Determine the airflow design for the equipment to be mounted. Then check to ensure that the racks allow for this design. For example, the HP ProLiant server line has always been designed for front-to-back airflow, thus mandating the use of a rack that supports this. In other cases, servers pull air from the data center subfloor and vent it out the top of the rack. It is therefore necessary to ensure that the servers and racks “match” each other—do they?
- It is highly recommended that the racks be arranged in a front-to-front, or back-to-back manner. Picture standing in between two rows of racks that either face toward each other, or face away from each other, and you will understand front-to-front and back-to-back, respectively. For maximum front-to-rear airflow, the *air registers* (the floor tiles with the little holes in them that force out cool air) should be placed in line with the front of the racks. Similarly, air returns located inline with the rear of the racks are a must, too, as you see in Figure 10.3. In the case of bottom-to-top airflow, the air registers are instead placed such that the rack sits on top of them. Are your air registers positioned for the best airflow?
- Racks need space. Is there enough room both in front of and behind each rack (25" in front and 30" in back is suggested, though your specific rack documentation may indicate otherwise) to open and close the doors, and to provide the cooling airflow necessary to keep everything running? If not, then reposition the racks, or move the offending gear/walls.
- Would purchasing a top-mounted fan option or split rear door (both enhance cooling to some degree, regardless of airflow direction) make sense, given the volume of computing gear to be racked? I generally recommend top-mounted fans in the densest of racks, regardless of the direction of airflow otherwise; these fans help draw component-damaging heat out of the rack as quickly as possible.
- Are there overhead fire protection sprinkler devices that must be considered? Check local building codes for acceptable clearances before installing racks underneath such devices.

At this juncture, it makes sense to actually diagram rack layouts for your specific data center needs. Again, I recommend working with your hardware vendor or an experienced systems integrator to get this right the first time.

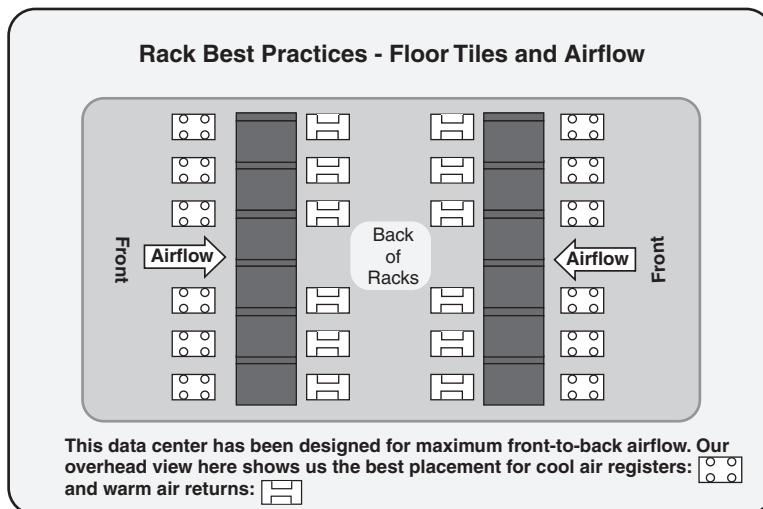


FIGURE 10.3 Placement and layout of racks and floor tile airflow registers/returns are key to maintaining proper air temperature in the SAP Data Center.

Optimizing Rack Real Estate

When loading equipment into the racks, observe the following best practices and observations:

- Put heavier items in the bottom, that is, UPSes and large servers.
- Address floor-loading weight limitations. For example, never load a rack with all UPSes, as the weight alone will almost certainly exceed the data center's raised floor load-bearing factor. Instead, distribute UPSes evenly across all racks.
- Locate monitors and keyboards for maximum ergonomic comfort (based on your particular sitting/standing arrangement). Do NOT locate them on the top of each of your 7-foot racks, like one of my customers did! It's funny reading about this after the fact, but it was no laughing matter at the time.
- Attempt to distribute monitors and keyboards such that access to a particular environment is available in two ways. For example, if an SAP Staging landscape consists of eight application servers, it would make sense to attach four of these to one monitor/keyboard, and the other four to the second monitor/keyboard. In this way, performing maintenance, upgrades, applying patches, and so on is possible from one monitor/keyboard, while the other remains available for business-as-usual.

- Rack blanking panels must be used to “fill in” spaces greater than 1U in size (where a “U” is a standard rack unit of measurement equivalent to 1.75 inches). This is important to ensure that good airflow is maintained within the rack enclosure itself.
- All items of gear housed within the rack—servers, disk subsystems, everything—should be installed with their covers and sides on. Running a server without the top cover or side panels, for example, disturbs the flow of air through the computer such that it is not cooled off as effectively as it otherwise would. In some case, running a server without the cover and sides will actually invalidate the unit’s warranty, too.

Other best practices exist as well. Refer to the Planning CD for a comprehensive “Rack Best Practices” document.

Rack Mounting and Related Best Practices

When connecting the racks to power, fault-tolerant considerations should be heeded. Your servers and disk subsystems already have dual redundant (or N+1) power, and to make the most optimum configuration, multiple (at least two) PDUs should be deployed. I recommend at least one mounted on each side of each rack. All right-side power supplies should be plugged into the right-side PDU, and the left-side power supplies into the left-side PDU. The PDUs should be plugged into separate circuits and breaker panels, as I have discussed previously. These circuits should be adequately sized to meet the potential load demand, which should rarely exceed 80% of their rated capacity if at all possible. In this configuration, if a circuit is lost, only half of the power supplies in each rack will be without power, leaving the other half to continue powering all of the computing equipment. And you will have no downtime.

Another thing to consider in rack planning is how the servers should be grouped. Ask yourself what makes sense. Some of my clients like to group all development resources together, all test resources together, and so on. Others prefer to group all servers together, all disk subsystems together, all network components together, and so on, similar to what you see in Figure 10.4. I suggest that you select either the approach that is used in your data center today, or the one that makes the most sense given the specific SAP system landscape being deployed.

Racking Clustered Servers

Regardless of the grouping approach, though, a best practice exists when it comes to clustering servers. Cluster nodes should always be mounted in separate racks, using separate keyboards, mice, and monitors, connected to completely different PDUs and network infrastructure, and so on. Why? Because if the cluster nodes share anything in common, this becomes a single point of failure. And given that you are clustering in the first place, avoiding single points of failure is key for these systems.

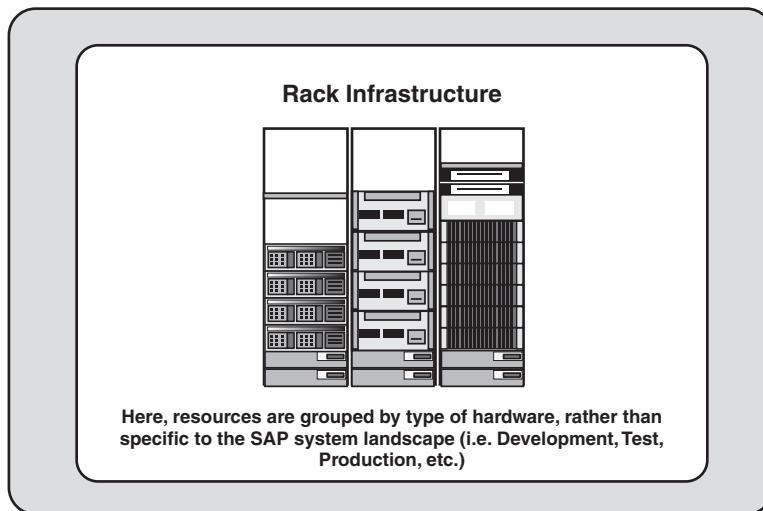


FIGURE 10.4 This grouping approach is hardware-centric, rather than SAP landscape-centric. Either approach is valid.

Racking Production

When it comes to mounting the Production servers in their racks, it is critical to maintain a separation of all infrastructure. That is, Production should ideally be isolated from all other SAP environments, from a power, network, rack, SAN, and server perspective. The Production racks therefore require their own power infrastructure, network infrastructure, and so on.

In the real world, as SANs continue to enable the highest levels of availability and scalability for mySAP customers, the line between production and other resources is beginning to blur. Expensive switched-fabric switches, SAN management appliances, SAN-enabled tape backup solutions, and the fibre cabling that ties everything together is often shared between production and development systems, for example. The important thing to address in these cases where the cost of a sophisticated SAN outweighs the risk of maintaining separate production resources is to provide for some kind of change management support. That is, you still need a test environment of sorts, be it a Technical Sandbox, Test/QA system, or whatever, to test the impact of changes to your SAN. In my experience, these changes are often firmware and hardware-related, though topology and design changes also need to be tested as well.

Cabling and Cable Management

If you have ever attempted to squeeze 21 servers into a 7-foot (42U) rack, the concept of cable management should evoke special thoughts of long days and late

nights. Cables have shrunk in diameter over the last five years, but not to the extent or in keeping with the trends we have all seen in hardware platform densities. As a result, one of the biggest immediate challenges facing new SAP implementations is how to best route the various keyboard, mouse, monitor, disk subsystem, network, and other cables required of each piece of gear. More and more, we see fat cables being replaced by smaller ones, and lots of little cables being replaced by fewer centralized cables. But in the end, there's still a whole lot of cabling to consider!

Alternatives to traditional KVM (keyboard, video, and mouse) cable connections abound today. The following represent two common methods of reducing the number and size of cables (a long-time favorite and a newer alternative):

- Installing a dedicated “computer on a board” into a PCI slot in each server, to facilitate out-of-band and in-band management of the server. Such boards typically require a network or modem connection only, and facilitate communications via a browser-enabled user interface. A fine example is HP’s Remote Insight Board. Depending on the particular server, these boards may come standard with the system, and may even be integrated into the server’s motherboard.
- Installing “KVM over IP,” which represents another way to shrink various cables into a small single network cable. This method also requires a small server of its own, not to mention licensing costs for the particular enabling product.

More often than not, I figure we will continue to see the widespread use of cable management arms, tie-wraps, Velcro-wraps, and similar inexpensive approaches. Regardless of the cabling techniques, though, continue to stay focused on *availability*. Single points of failure can exist in this realm like any other area—remember that a server may be effectively unavailable once it loses access to its keyboard, mouse, or video connections. Even in the best case, if the server remains up you may still be operating blind, saved only through your management and monitoring tools if so enabled.

Network Infrastructure for SAP

Given that SAP is architected to support a distributed three-tiered design, a slew of configurations exist that can potentially impact high availability. There are similar performance considerations as well, as the database, application, and Web tier layers are all affected. For example, a back-end network is recommended to interconnect the database and the application servers, another network interconnects the application servers to the Web/Internet layers, and a third public network addresses client requirements. If you plan on pulling backups across the network (rather than via a

disk subsystem that supports direct-attached SCSI or fibre channel tape drives), a separate back-end network subnet is highly recommended in this case, too.

In all cases, 100Mbit switched network segments are warranted, if not Gigabit Ethernet. Backups are very bandwidth-intensive, and if data is transmitted over otherwise crowded network lines, database disconnects between your application and database servers may result, for example. In this case more than any other, therefore, Gigabit is warranted. Beyond network-enabled backups, the second preferred subnet in which to leverage Gigabit includes the network connecting the application and database servers, where traffic can also be quite heavy. This is especially true as more and more enterprises grow their mySAP environment by adding new SAP components but insist on leveraging the same backend network.

The application/Web tier often consists of a single 100Mbit subnet. As the application servers/Web servers are usually not considered critical from a backup perspective, dedicated network backup segments are usually not warranted in this case—these servers usually contain fairly static data that lends itself to weekly or other less-regular backups.

The Web tier is typically split, though. In the case of SAP ITS implementations, for instance, an application gateway (AGATE) connects to the SAP application servers, a Web gateway (WGATE) connects to the AGATE, and the end users connect to the Web gateway. Thus, the WGATE is often housed in a secure DMZ for access via the Internet or an intranet, while the AGATE remains behind a firewall, safe in its own secure subnet, as you see in Figure 10.5. In this way, the network accommodates the performance needs of the enterprise without sacrificing security.

Network Fault Tolerance

As I indicated earlier, there are many ways to architect a network solution for SAP. Simply segregating each layer in the solution stack achieves minimum performance metrics but does not address availability. In fact, it actually increases the chance of a failure, as more and more single-point-of-failure components are introduced. Fault tolerance must therefore be *built* into the design, not looked at afterwards. To this end, I will next discuss the primary method by which network availability is designed into the SAP data center, using an approach I call *availability through redundancy*.

Just as a redundant power infrastructure starts with the servers, disk subsystems, and other hardware components, so too does a redundant network infrastructure start with the networked servers. Here, redundant *network interface cards*, or NICs, are specified, each cabled to redundant network hubs or switches, which are in turn connected to redundant routers. Figure 10.6 illustrates this relationship between the servers and a highly available network infrastructure.

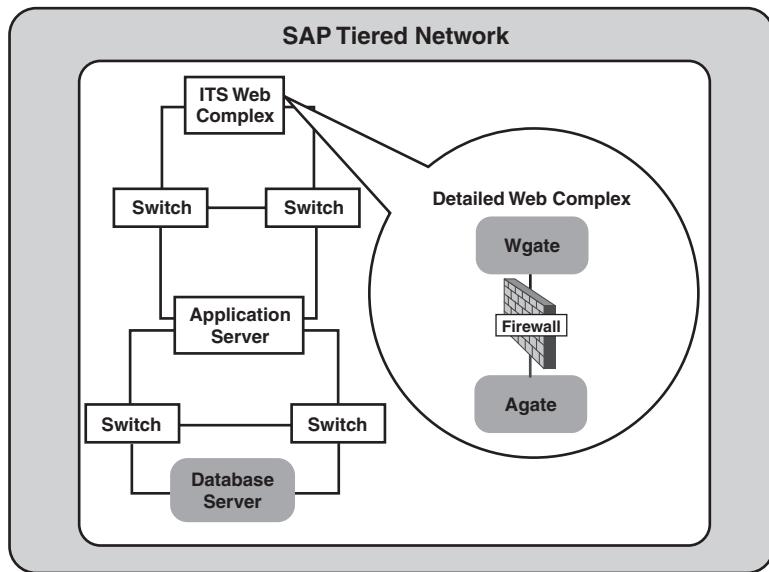


FIGURE 10.5 Network architecture involves every layer of the SAP Solution Stack, from the database server up to the application and Web layers.

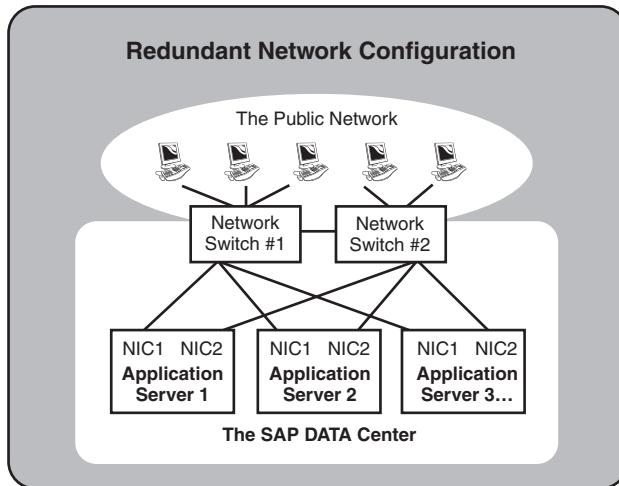


FIGURE 10.6 This example shows what a properly architected highly available network infrastructure may look like for SAP.

Network fault tolerance starts not only with two (or otherwise redundant) NIC cards, but also with OS-specific drivers capable of pairing two or more physical network

cards into one “virtual” card. Hardware vendors often refer to this as *teaming* or *pooling*, but other labels have been used for this important piece of the network puzzle. Regardless of the label, the idea is simple: Pair two or more network cards together, and use both concurrently for sending and receiving the same exact packets of data. Then, use the data on one network card/network segment (referred to as the “primary” NIC), and discard or ignore the (exact same) data on the other NIC as long as the network is up and things are going well. If a failure occurs with one of the NICs, network segments, or hubs/switches on the “redundant” path, continue processing packets as always on the primary path—business as usual. However, if a failure occurs with one of the NICs, network segments, or hubs/switches on the “primary” path, immediately turn to the redundant path and start processing the data moving through this path instead.

Today, most every operating system supports NIC teaming for high availability. Many OSes also support variations of NIC teaming geared toward improving performance, including network bonding, adaptive load balancing, FastEtherChannel, Gigabit EtherChannel, and more. Remember, though, that these latter variations do not necessarily improve availability; teaming must either be inherently supported by or used in conjunction with these different performance-enhancing techniques or approaches.

NOT ALL SERVER CONFIGURATIONS SUPPORT NIC TEAMING

Some operating systems simply do not support specific network cards when it comes to NIC teaming. Further, Microsoft’s cluster service specifically prohibits clustering the private high-availability server interconnect, or heartbeat connection, regardless of the type of NIC—Microsoft simply does not support teaming the heartbeat. Network availability in this case is gained by configuring the cluster service such that both the private and public networks can send “still alive” messages between the cluster nodes. As an aside, best practices suggest that the private network be preferred for this activity, and that the public network only be leveraged when the private network fails or is otherwise unavailable.

Let us return to our primary discussion on designing and implementing highly available networks for SAP. Remember, each NIC in the team should be connected to a different switch that in turn is connected to a separate router or separate card in a highly available router configuration (as required). Each redundant switch, router, and any other network gear must then be serviced by redundant power to indeed create an NSPOF, or no single point of failure, configuration. Insofar as best practices are concerned, I recommend that you also adhere to the following guidelines:

- All network interfaces must be hard-coded or set to a specific speed, for example, 100Mbit Full Duplex. Refrain from using the “auto configure” function, regardless of how tempting it is to avoid three or four mouse clicks per server to hard-code each NIC setting specifically. This is especially a problem if ignored in clusters or in network environments characterized by switches and

hubs servicing different network segments running at different speeds. Using the “auto configure” setting can easily mask network problems, including intermittently failing network cards, and ultimately add hours or even days to troubleshooting cluster issues.

- Keep in mind that NIC teaming is implemented via a software/driver-level function. Thus, anytime one or more NICs participating in a team are replaced, swapped out, or reconfigured, the team should be dissolved and reconfigured again. Failure to do so could create issues difficult to troubleshoot, including intermittent or unusual operation of the NIC team.
- Most servers manufactured in the last 10 years take advantage of multiple system busses. Each bus is normally capable of achieving a certain maximum throughput number. For a 64-bit 66MHz PCI bus, this number is something approaching 528 MB/second. To achieve even greater throughput, then, NICs need to reside in different busses. Taken one step further, the idea of multiple busses should also get you thinking about high availability. Remember, an NSPOF solution theoretically relies on redundant components everywhere, even inside the servers. So, multiple PCI busses fit the bill—as an added level of fault tolerance, place the NICs in different PCI busses.
- Finally, if multiple NIC teams are configured (for example, in the case of SAP Application servers, where redundant connections are preferred to both the database server and the public network or Internet/Web intranet layer), it is important to ensure that the MAC address of the first team represents the primary address of the server. The primary address of the server thus maps back to its name on the public network. This sounds a bit complicated, but hopefully makes sense in Figure 10.7. Here, Network Fault Tolerant Team #6 represents the public team. Further, the MAC address of NIC #1 of this team is the MAC address “seen” by other servers. In this way, both name resolution and failover work as expected.

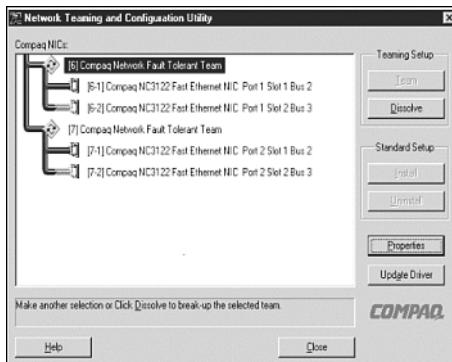


FIGURE 10.7 A unambiguous graphical user interface can help clear up otherwise complicated multi-NIC teaming configurations.

Central Systems and Optimal Network Configuration

Network configurations for SAP are also impacted by the type of server deployed in the system landscape. All of our attention thus far has been focused on three-tiered architectures, where the database, application, and client components of SAP are broken out into three different servers/hardware platforms. However, it is also quite common to deploy two-tiered SAP systems, or “Central Systems,” especially when it comes to small training systems or uncomplicated testing environments. In this case, the first question a network engineer new to SAP asks is “Do I cable the system to the public network, or the back-end database network?”

The question is valid. And because each option seems to make sense, there has been debate in the past on this very topic, to the point where SAP finally published an excellent paper on network recommendations for different system landscapes. Let’s assume that our Central System is an R/3 Development server, and take a closer look at each option in an effort to arrive at the best answer.

If our new network engineer cables the Central System to the back-end network (a network segment upon which other three-tiered SAP database servers reside), the following points are true:

- Given that the Central System is also a database server, the database resides where it is “supposed to”—on the back-end network.
- Any traffic driven by the DB server thus inherently remains on the back end as well. For example, if the SAP BW Extractors are loaded on this system, all of the network traffic associated with pulling data from our R/3 system to populate BW cubes stays off the public network. Again, our network traffic remains where it is supposed to.
- Any other DB-to-DB traffic, like transports, mass updates or data loads, and so on, also stays on the back-end.
- However, because our server is not on the public network, it is not as easily accessed by users and administrators. That is, name resolution becomes a challenge.
- Further, because the back-end network tends to be much busier than typical public networks (which from an SAP perspective only service low-bandwidth SAPGUI and print job traffic most of the time), our end-user response times will vary wildly, suffering across the board.

These are interesting points, but imagine how complex the issue becomes if your Central System now grows into an SAP cluster. In this case, because a cluster is implemented as a three-tiered configuration, you might be inclined to hang one

node (the database node) off the private network, and the Central Instance/Application Server Node off the public network. But what happens when the second cluster node fails over and the DB node becomes the CI/Application server as well as retaining its DB Server role? Or vice versa, and the CI/Application server takes on the role of the database server, too?

By now, you should understand that the best network solution in this case is not an either/or answer. Rather, the best solution is predicated on accessibility to both the private and public networks. This is accomplished by adding another network card into the server, NIC #2, and setting it up for connectivity to the back-end network. Meanwhile, the original NIC (now labeled NIC #1) is moved to the public network, where it is assigned a public IP address. Name resolution and other public network-related services regarding accessibility are now easily addressed. And by using static routing, all DB-to-DB traffic is retained on the back-end network segment as desired. Best of all, the cost to achieve this solution is minimal—a second NIC, another IP address, and a few hours of time are really all that is required.

To verify that your static routing indeed operates as it should after SAP has been installed, launch a SAPGUI session, log in with typical SAP Basis rights and privileges, and follow these steps:

1. Execute transaction **/nSMLG**.
2. Press F6 to go to the Message Server Status Area.
3. Verify that your PUBLIC subnet (under “MSG SERVER”) is listed under the Logon Group Name. If this is not the case, return to the previous screen and under the Instance column, double-click each application server, and then click the Attributes tab.
4. In the IP address field, enter the associated PUBLIC subnet for each server, and then click the Copy button.
5. Verify these settings again, by highlighting the application server and pressing the F6 key—the logon group will list the PUBLIC subnet.

By following these steps, you can in essence verify that the traffic associated with end users stays on the *public* network, thus ensuring that their response times will be as fast as possible—certainly faster than if the end-user traffic was routed over the back-end network.

With your highly available and optimized network infrastructure installed, you can now turn to the next few layers in the SAP Solution Stack—the network server and the server operating system.

Network Server Preparation

In the next few pages, I'll take a look at how to best set up and configure servers for their roles in the SAP system landscape. This includes optimum server hardware configuration best practices, details related to deployment and installation, and keys to configuring highly available and well-performing server-based systems.

Optimum Server Configuration Best Practices for SAP

When it comes to server configuration best practices, consistency and standardization are the keys. Consistent server configurations and standard server "builds" or software installations will eliminate a lot of future headaches. Your goals are simple:

- Minimize variety, which really equates to minimizing variables should you ever need to troubleshoot an issue
- Allow for rapid server replacement
- Support your systems management approach (discussed in detail throughout Chapter 14)
- Support the lowest total-cost-of-ownership model possible

I cannot say enough about minimizing the different server platforms in your SAP landscape. Two or three specific models of servers should be adequate for even the most complex installations or most cost-conscious customers. The benefits are great—fewer platforms for the SAP TSO to learn to support, less variety in the kind of hardware spare parts that might be stocked onsite, a simpler operating system stack (that is, from a software driver perspective), and so on. I recommend the following, considering these best practices regardless of which server platform is deployed:

- Only servers specifically certified by the hardware vendor to support SAP should be used.
- Maintain PCI slot consistency—whenever possible, keep all PCI cards in the same slots on all servers across the landscape. For example, if all servers house a public network card in slot 4, and a back-end network card in slot 6, the SAP TSO support staff will be less inclined to make false assumptions or mistakes later should a network issue arise.
- Standardize on a particular model of network card, disk controller, host bus adapter, and so on. In this way, not only do you minimize variables from a hardware perspective, you also minimize the variety of software drivers that need to be supported, and the variety of spares that needs to be maintained. In this way, you can even go so far as to swap NICs or HBAs from one server to another, to support troubleshooting without worrying about driver differences, for example. And this approach to standardization simplifies change control as well.

- Identify any PCI slot constraints, and ensure that these constraints are documented and followed. For example, particular HP servers only support the Remote Insight Board in a particular slot.
- Disk drives housed internal to each server should contain the operating system and Pagefile or Swap files. We'll discuss this in more detail later. The point here is that all other data types—database executables, SAP executables, database log and data files, and so on (unless otherwise required) should be housed externally.
- All internal disk drives should be protected in terms of availability. In some environments, this means attaching these drives to a hardware-based RAID Array controller (a high-availability and high-performing controller capable of withstanding the loss of one or more disk drives, depending on the specific configuration), and mirroring the drives. RAID stands for *redundant array of inexpensive disks*, and can be implemented via hardware or software (though in the latter case only if the operating system supports software-based RAID). Note that RAID implemented via software can be performed on standard disk controllers—no special controllers are needed beyond those supported by the OS. For the best availability, the controller and any necessary cables should be redundant as well. I will discuss RAID and RAID Array controllers in detail later in this chapter.
- All external disk drives should be attached to a separate RAID Array controller (or host bus adapter in the case of Storage Area Networks or other fibre-based storage systems, also discussed later in this chapter).
- If a disk controller does not support mirrored or otherwise protected cache (which is really just RAM that resides on the controller, to speed up reads and writes), or that cache is not backed up by a battery in case of loss of power, the controller should not be used for database or log drives. The reason is as follows. The cache allows writes to be “posted” or acknowledged by the controller, but not actually written to the physical disk drive. In this way, performance is greatly enhanced—the disk controller tells the operating system “I have the data” even when it's so busy that it does not have the time to actually write the data to the drive. Meanwhile, the operating system is then free to do more work. Later, when the disk subsystem has the time, it commits or writes the data in cache to the physical drives, and the operating system and database are none the wiser. However, if the controller loses power (usually the result of controller failure or losing power to the entire server), all of the writes sitting in the controller's cache never actually make it to the physical disk, and our database becomes inconsistent or corrupted. A battery backup found on the better controllers gets around this problem, though, by “holding” the data until the server comes back up. At that time, the controller then commits the writes, and again the operating system and database are none the wiser.

- Standardize on specific firmware revisions for all hardware—servers, disk controllers, disk drives, tape drives, and so on. Like software, firmware tells the hardware how to react or what to do, but it resides on a tiny read-only silicon chip on the hardware component itself. Also like software, firmware is subject to changes, bug-fixes, and so on. I recommend that a customer maintain a specific revision of firmware on each hardware component consistent with the vendor’s SAP Competence Center’s advice, testing planned changes in a Technical Sandbox first. Note that the “latest and greatest” firmware is not necessarily the best, and that a change even at a firmware layer requires the change to be tested and carefully promoted throughout the SAP system landscape (consistent with change control best practices discussed in detail in Chapter 13).
- If it ain’t broke, don’t fix it—After a stable platform is assembled and supported by the vendor’s SAP Competence Center, best practices dictate making no changes until forced to. This holds true for hardware components, firmware revisions, software drivers and patches, and so on. Think of it in this way—a lot of work went into assembling a collection of different SAP Solution Stack layers that actually work well together. Stay away from change *for the sake of change*, and your monthly reports reflecting unplanned downtime numbers will look that much better.
 - ▶ To review the different levels of RAID, and learn more about configuration options, advantages, and features, see “Disk RAID Configurations and Features,” p. 243 in Chapter 7.

With standardized hardware platforms behind us, let’s embark on the journey to installing and optimizing an operating system for SAP.

Operating System Best Practices for SAP

Although many operating systems are supported for SAP, the following discussion focuses on the most popular OS underpinning new installations over the last three years, the Microsoft Windows family of OSes. In order to install an OS, you need to partition the hard disks upon which the OS will reside. Notice I said hard *disks*—I highly recommend mirroring the OS partitions or complete drives, such that if one physical drive fails, you can still continue to run from the second drive. With regard to installing the OS, you have two choices:

- If you are using a RAID Array Controller, you must run an array configuration utility to partition the drives. This readies them to be recognized by an operating system.

- If a standard SCSI or other controller is used (recommended in cases where software mirroring is warranted), nothing special needs to (or can) be done from a hardware perspective.

Create the OS Partitions and Logical Drives

In the case of a hardware array, create a mirrored pair (also called a RAID 1 set or mirror set). If possible, for maximum availability place the mirrors on drives that reside on separate SCSI busses. This is possible in the case of dual-channel controllers, or with servers where dual SCSI busses reside on the motherboard. In any case, by doing this, an outage can be avoided if a SCSI bus or cable fails. Not only that, but mirroring increases read performance as well, as both SCSI channels and both disk drives can later conduct work simultaneously.

After the disk drives are configured for a RAID set (as appropriate), create a logical drive. This is what will be “seen” by the operating system.

NOTE

If more than four partitions need to be seen by Windows on a single basic drive, you will need to create additional logical drives. This is because Windows is limited to creating four partitions on the primary bootable drive.

If you have started with a pair of 18GB drives, and assume you need only four or fewer disk partitions to be recognized by Windows, you are done—save the array configuration, and reboot the server to the Windows 2000 installation CD. Install the operating system, selecting to make the bootable partition 4GB or 8GB (or your standard boot drive size), and select the option to format it for NTFS. Finally, after the OS installation, carve up the remaining disks via Windows’s Disk Administrator utility into something just over 4 GB each—I recommend using these additional partitions for the Windows 2000 Pagefiles, discussed next.

Optimal Pagefile Configuration

Windows 2000 is still limited as to how large a physical pagefile can be made—4095 MB. And unless you take advantage of a published and fairly simple registry hack, you can still only place one pagefile on a disk partition (that is, one pagefile of 4095 Megabytes on the E: drive). As of SAP’s most recent Basis releases, total pagefile size should still equal about four times the amount of physical RAM in the server. Thus, a server with 3 GB of physical RAM should be configured with 12 GB worth of pagefiles. SAP further states, though, that a total pagefile exceeding something in the neighborhood of 10 GB offers little to no performance enhancements.

Keep in mind that Microsoft recommends a pagefile on the bootable drive (usually C:) equal to or greater in size than the amount of physical RAM, to support capturing full core dumps should the server succumb to a memory failure. Of course,

another simple registry hack allows us to get around this as well. Bottom line, though—if you have the disk space, create the pagefile.

If we return to my server example with 3 GB of RAM, the simplest supported manner of configuring pagefiles would look like this (accomplished by using Control Panel, System):

- The C: drive should be set up for 3 GB of pagefile.
- An additional D: drive should be formatted and configured for a 3.5GB pagefile.
- A final E: drive should be configured for a 3.5GB pagefile, too.

In this way, the server is configured for 10GB of pagefile total. And we have not had to do any registry hacks, nor sacrifice the ability to capture memory dumps. In addition, because we are not completely filling up the drive with a pagefile (we are leaving 500 MB of free space, assuming 4 GB partitions were created), we will avoid those pesky “disk at or near capacity” messages in the Windows Event Viewer. Finally, rather than wasting 2 GB of unnecessary space by building a 12GB pagefile, we instead took advantage of SAP’s concession in pagefile sizing, and created 10 GB total instead.

Another good example of pagefile sizing can be seen in Figure 10.8; note the partition sizes of just over 5GB each, allowing for maximum-sized pagefiles of 4095MB, with room to spare.

Pagefile Sizing Best Practices					
Disk 0		OS/Pagefile (C:)	Utils/Pagefile (D:)	Pagefile (E:)	
Disk 0 Basic 16.95 GB Online	44 MB FAT Healthy (E)	5.86 GB NTFS Healthy (System)	5.86 GB NTFS Healthy (Page File)	5.19 GB NTFS Healthy (Page File)	
Disk 1				Pagefile (H:)	
Disk 1 Basic 16.95 GB Online	PageFile (F:) 5.86 GB NTFS Healthy (Page File)	PageFile (G:) 5.86 GB NTFS Healthy (Page File)		5.23 GB NTFS Healthy (Page File)	

• Multiple pagefiles have been configured for best performance.
 • Each resides on a disk partition with plenty of room to spare.
 • Thus, each may be configured for the maximum size of 4095 MB.
 • Additionally, the OS partition houses a pagefile large enough to address core memory dumps, if required.

FIGURE 10.8 Pagefile sizing should adhere to the best practices noted here.

Other OS Configuration Requirements or Best Practices for SAP

The following configuration changes need to be made or considered prior to installing the database and mySAP component:

- Work with the SAP Basis team to select an SAP system name, and rename the server with this name. In the past, the server name could not exceed eight characters (releases prior to SAP Basis release 4.6C). Now, the limitation varies depending upon the specific basis release. However, given that SAP management utilities and applications have not all been updated to accept these new limitations, it is still a good idea to limit the name to eight characters until your own specific testing verifies that everything works properly with longer names.
- Create the admin/installation user (that is, P11adm or something similar) and give it domain administrator (or admin) rights—all SAP Basis installations, updates, and administration will be performed with this user ID.
- Ensure that all drives were indeed formatted for NTFS (Start Windows Explorer, select each disk, right-click, click Properties, and then select the General tab). Further, format each disk for 64KB blocksizes (or allocation units), and label each appropriately (that is, Pagefile1, Pagefile2, and so on).
- If not already accomplished, label all remaining disk drive partitions via the Windows 2000 Disk Administrator. For example, label the C: drive *OS*, the D: drive *Pagefile1*, the E: drive *Pagefile2*, the F: drive *EXES*, the G: drive *LOGS*, the H: drive *DATA1*, the I: drive *DATA2*, and the J: drive *DATA3*. I also like changing the CD-ROM to drive letter Z: (or another drive letter “out of the way”).
- Set both the TEMP and TMP environment variables to point to C:\TEMP (by using Control Panel, System).
- Create the C:\TEMP directory if it does not exist already.
- Under Network Settings, ensure that the Maximize Throughput for Network Applications option is selected on the File and Printer Sharing tab. This impacts how memory/cache is allocated, and makes a significant performance difference.
- For each disk partition, grant “everyone” permissions to the root and all subdirectories.
- Load the SNMP service, in preparation for any systems management agents that will be installed later.
- Ensure that the appropriate Windows 2000 Service Pack is applied. Note that SAP and Microsoft will oftentimes indicate that a particular version of an OS fix, or patch, or Service Pack is supported. However, I really recommend verifying with your hardware vendor that the specific update in question is

recommended and supported on *their* specific server and disk subsystem platform. Work with the hardware vendor's SAP Competence Center to verify this information.

- Ensure that the vendor-specific software kit is approved by the vendor's SAP Competence Center, and then apply these updates on top of the latest service pack (unless instructed otherwise).
- Install the systems management agents (that is, Insight Manager, HP Openview, BMC Patrol, and so on per your standard). Refer to the appropriate management agent installation guide for any other details or prerequisites.
- Set up an alias for SAPTRANSHOST in the HOSTS file—edit C:\WINNT\SYSTEM32\DRIVERS\ETC\hosts and add an entry for SAPTRANSHOST (make it equivalent to the public TCP/IP address, and ensure that the SAP Basis team knows this has been set).
- Install Internet Explorer (via the SAP Presentation CD).
- Install the Active Dir Svcs Interface (ADSI), via the SAP Kernel CD (search the CD for ads.exe), if not already installed.
- Return to Control Panel, System, Advanced, and click the Performance Options button. Verify that performance has been optimized for background services. This setting ensures that SAP processes are given priority over locally logged-on users.
- Verify via the Windows 2000 Event Viewer that no issues exist from a hardware or OS perspective.
- Verify via your Systems Management Console (once installed) that no issues exist from a hardware or OS perspective.
- After the entire configuration process up to this point has been completed, I highly recommend that an automated or unattended installation script be created to easily set up a server that reflects all of the activities completed so far in this list. A disk imaging utility like Ghost is another good way to go. Other options include Microsoft answer files, and HP SmartStart scripts that load HP specific drivers as well as the OS. I suggest leaving out some of the specifics regarding pagefile sizing and partition labeling, so that the script is that much more useful in a generic “SAP server installation” manner. Barring all of these automated approaches, a good old-fashioned checklist will serve the same purpose, as long as it is followed precisely.
- Finally, to enable remote administrative access to each server, I suggest that a remote control application be tested and deployed. A number of very good solutions exist in the market today—Terminal Services for Windows 2000 is an excellent place to start.

SAP Server Configurations in the Real World

My colleagues and I have seen some pretty amazing SAP server configurations in our day. Think of these as more “Lessons Learned.”

More than once I have seen SAP production systems that are clustered in the name of high availability but forgo basic availability “in the box.” For example, one site chose not to mirror the internal disks located in each SAP cluster node, nor duplicate the requisite controllers and cables. Although this is not absolutely horrifying, it does mean that if a disk drive, controller, or controller cable were to fail, the server would die and the cluster would be forced to fail over to stay up. It has always been my view that it is preferable to never exercise your cluster failover capabilities unless absolutely necessary.

I ran into another cluster issue recently, too. It wouldn’t fail over. As it turned out, it was a one-node cluster.

In another case many years ago, my customer wondered why each server took a performance hit after 15 minutes of uptime. In their eyes, it was definitely a hardware issue. As soon as I walked up to the servers, the real reason was apparent, though—a very nice, and very processor-intensive, screensaver was set to start after 15 minutes of keyboard and mouse inactivity.

One of the biggest problems I see with server configurations regards the use (or misuse) of hosts files. As any network technician knows, the hosts file allows for host name resolution back into an IP address. However, if the contents of the hosts file is incorrect, or the host file itself is absent (that is, perhaps renamed to *host*, rather than *hosts*—another real customer issue of mine), and DNS is not being used, name resolution will simply not work. This problem can manifest itself in a “host” of ways—the SAP Basis installation will fail, the server will seem to be unavailable across the network, and so on. Best practices and experience tell us that the hosts file, if used, should reference all SAP servers in the system landscape, and their IP addresses. Each hostname should be listed twice, once in uppercase and once in lowercase letters. If applicable, cluster alias names should also be defined here.

Enough about configuring servers for SAP. Now, let’s move into one of the most challenging SAP Solution Stack layers, the disk subsystem.

General Storage Considerations

In my experience, perhaps 85% of SAP performance problems are considered to be storage-related. This is not to say that 85% of *all* SAP problems can be traced back to the storage system—but by the time I am called in, the easy fixes are typically taken care of. So I rarely run into simple profile parameter problems, or issues with other hardware subsystems, or problems with the network, either. Instead, after a quick

review of the entire solution stack I usually find myself drilling down into the following:

- The model and features of the disk subsystem servicing the systems that exhibit the slow performance
- Details as to how the disk subsystem has been configured
- Database statistics, to quantify how well the database itself performs on the given disk subsystem platform
- Specific transaction loads on the system (that is, batch processes or especially heavy online user transactions), especially the top 40 or so transactions identified by transaction ST03 as consuming the most database request time
- Finally, the ABAP programs (or other code) that is actually being executed by these top 40 transactions

Although a variety of disk subsystems are deployed today in support of SAP solutions, including direct-attached SCSI, Channel-Channel Arbitrated Loop-based storage, and a few NAS (Network Attached Storage) systems, the discussions that follow focus on the predominant disk subsystems being deployed today—Storage Area Networks, or SANs.

- ▶ To review the most common types of disk storage deployed for SAP environments over the last five years, see “Commonly Deployed Disk Subsystems for SAP,” p. 244 in Chapter 7.

Special Considerations for Storage Area Networks

Deploying Storage Area Networks, or SANs, for current and new SAP system landscapes has kept many SAP professionals quite busy for the last two years. The latest iterations represent not only some of the fastest disk subsystems ever manufactured, but also represent everything that customers need in a highly available and scalable disk solution. Not only does a SAN give us the ability to expand quickly, but it also allows us to move data around with ease. Snapshots, disk clones, data replication, and more satisfy high-availability requirements, and help us to address disaster recovery, too.

- ▶ To learn more about the role SANs can fill when it comes to disaster recovery, see “Disk Subsystem Single Points of Failure,” p. 180 in Chapter 6.

With these new capabilities comes new complexity, of course, and certainly new paradigms. In some cases, we also have a new set of issues that must be addressed when deploying the latest in SAN technology—switched fabric designs, planning for different data access models within the same SAN, providing connectivity to new

shared resources like tape drives and tape libraries, and all of the complexity that comes with deploying new software solution sets that interoperate seamlessly (or nearly so!) within the SAN environment. Moving on, let's sneak a quick look at some basic best practices for implementing SAP in a SAN environment, with the understanding that the newer “virtual” SANs will be covered in detail later.

Deploying SAP on a SAN—General Best Practices and Observations

To approach this in an organized manner, we will start with some general SAN observations, and then cover the servers that play a role in the SAN, move to the SAN infrastructure, and finally wrap up with the disk subsystem itself. The following list has been assembled as a result of more than a hundred SAP-on-SAN design or deployment engagements:

- As the SAN is a special network encompassing all components from the HBA to the disk, it is incorrect to call each cabinet of controllers and disks a SAN. Rather, the entire connected solution is the SAN. If the development environment is off by itself and is not connected in any way to production, for example, we have in place a Development SAN and a Production SAN. After these SANs are connected (that is, via a fibre cable connecting one SAN’s switch to the other SAN’s switch), it then becomes a single SAN.
- Each server needs at least a single Host Bus Adapter, or HBA. Because this represents a glaring single point of failure, it is always recommended to install and configure two where high availability is important. Both HBAs are cabled to the same SAN, but to redundant switches (discussed later).
- A server should not connect to two different SANs—connecting to multiple SANs concurrently is not supported. In other words, a single server with two HBAs should not be connected to two different models or implementations of a SAN.
- If a server contains two HBAs, it actually now has two paths in which to access data on the SAN. To manage this condition, an OS or OS-supported software utility is typically employed on each server connected to the SAN. Hewlett-Packard’s line of StorageWorks SANs uses a utility called SecurePath, for example. SecurePath require a license for each server, but ships with all StorageWorks cluster kits—SecurePath can help to ensure that the HBAs and data access paths are balanced, too, from a performance perspective.
- Each HBA requires what used to be called a GBIC, or gigabit interface connector, as might each port in your fibre switches. Today, GBICs are more commonly called transceivers.

- Fibre cables then run from each GBIC in each HBA to a GBIC (or otherwise pre-enabled port) in each redundant set of fibre switches.
- In regards to fibre switches, different switch vendors may have different rules regarding cascading their switches (to increase the port count available to the SAN, for example). I highly recommend that the specific vendor's documentation is referenced for details, with the understanding that many SAN fabrics in use today are limited to very few levels of cascading, or *hop counts*.
- Switched fabric switches should be configured with dual power supplies. Two switches are required at minimum to address high availability. A fully redundant SAN/server configuration consists of four fibre connections—two to the server, and two to the disk subsystem. These fibre connections should be carefully mapped to the switches so that each controller is connected to two switches, not one.
- Either something referred to as "zoning" or something referred to as "selective storage presentation" is used to ensure that a specific server can only access a specific set of disk drives or virtual drives. The implementation of this varies with the vendor as well as the product set. Regardless, though, be sure to implement one of these access protocols. Failure to do so risks corrupting data.
- Now that we have established connectivity from each server to the SAN, we need to carve up disk drives on the SAN. Until November of 2001, some of the highest-performing SANs on the market leveraged up to six separate SCSI controllers across six different SCSI busses, or drive shelves. Each drive shelf was capable of housing 14 drives, for 84 disk drives total. It is best to view the disks vertically, in that we stripe data "up and down" the disk subsystem. A maximum of six drives (when six shelves are deployed) can be configured per logical drive in this manner. By addressing our SAN disk configuration in this way, we mitigate risk inherent in a bus or shelf or shelf-power failure. This in turn prevents us from ever losing data in the event of one of these failures.
- Whenever possible, I promote the idea of using a graphical user interface to address day-to-day SAN management, and a scripted command-line approach for actually configuring a SAN (to allow for rapid cookie-cutter standard SAN installations).
- For SAP we need several sets of disks, depending on which database vendor has been selected. All data and log files (including redo logs, archive logs, SQL transaction logs, TempDB, and so on) must reside on the SAN. Further, as long as we are *not* using a clustering technology that requires local access to database executables, the Oracle or SQL Server (or Informix, DB2, SAPDB, and so on) executables can be located on the SAN as well.

- If you are clustering, you need to add a Quorum drive or similar such drive as well (actually, a mirrored disk pair distributed across two different busses is recommended). The quorum *must* be located on the SAN. Also, clustering SQL Server or Oracle means that these database executables must be installed on each node in the cluster, not out on the SAN.
- For the production database, I nearly always recommend that the production datafiles reside on LUNs (drive partitions) set up for hardware-based mirroring and striping (RAID 1+0 or 0+1, depending on the storage vendor's implementation). For simplicity, I like to create LUNs where the physical drives are vertically situated one on top of the other.
- SAP BW represents a potential exception to this general configuration rule of thumb for production systems. In the fast disk subsystems available today, the trade-off between performance and disk space between the different RAID levels is not as significant as it has been in the past. I discuss these RAID levels in greater detail in Chapter 7.
- If all of the preceding items are in place, you are well-prepared to address database growth. As the database grows you can add another RAID 1+0 set, for example. The OS will immediately recognize this additional disk space (if not, go to Disk Administrator and execute the option to "scan for new devices"), and allow us to format this space and create a new drive letter. Then, in order to increase your effective database size, the database administrator simply needs to create more table spaces or data files on the new disk partitions.

A good example of what a traditional SAN deployment might look like from a disk-layout perspective can be seen in Figure 10.9.

With your new understanding of how to take advantage of the features and capabilities a classic SAN environment can provide in regards to our SAP environment, let's move ahead and discuss the latest in SAN technology—the virtual SAN.

The Latest in SAN Technology—Leveraging Storage Virtualization

In review, *Storage Virtualization*, or SV, is defined as "the transparent abstraction of storage at the block level." Regarding SV, the idea is to minimize the need for parameters that allow fine-tuning, low-level optimization, and "tweaking," in favor of allowing the intelligent virtual disk subsystem to take care of everything. It's a paradigm shift in the biggest sense of the word—hands-off!

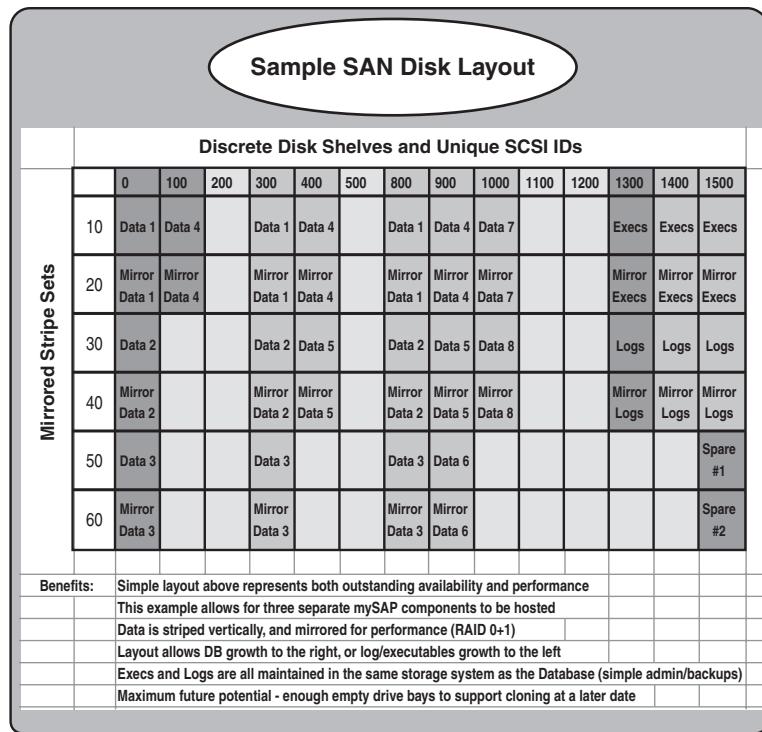


FIGURE 10.9 A well-performing and highly available disk subsystem requires that no two drives share the same storage shelf, power cable, or SCSI bus.

The ultimate in high-performance SV equates to striping and then mirroring all disk partitions across all 84–168 or more disk drives in a virtual storage disk subsystem. The ultimate in storage capacity equates to creating a RAID 5 partition across all of these drives. In either case, multiple partitions, or drives, can be created and presented to the operating system, or a single enormous drive can be created and presented instead. And many storage virtualization technologies automatically create optimally sized and configured RAID drives based on the space available in conjunction with historical data. It's that simple. The various block size parameters, read/write caching algorithms, read-ahead logic, and so on are all under the covers, so to speak. The best SV disk subsystems allow the few remaining configuration choices to be made via a browser-based graphical user interface, too.

- To read more about designing and sizing a Virtual Storage Array, see “Storage Virtualization—The Latest Paradigm in Enterprise Computing for SAP,” p. 246 in Chapter 7.

Virtual storage eliminates the physical one-to-one relationship between servers and storage devices, or databases and physical drives. Some important, albeit limited, configuration options still exist, however, and it is these options that concern us next.

Storage Virtualization Best Practices

When it comes to implementing and configuring a Virtual Array, you have control over the following options:

- As with a traditional SAN, you can still design and implement high-availability SAN switch architectures, including the ability to create zones for effectively segregating specific servers and their storage across multiple storage systems.
- You can create one or more groups within a physical Virtual Array storage system. Groups represent another way to segregate storage and servers, though this capability is limited to disks that are physically attached to the same disk controller subsystem (normally in one or more “cabinets” or “disk enclosures”). Thus, in an 84-drive Virtual Array, you might choose to create two groups of 42 physical drives each, three groups of varying sizing, or just a single group containing all 84 drives.
- You can manually create LUNs, or disk partitions, and select the group (and therefore disk drives) over which this LUN will be created. You can create many LUNs, or one LUN, whatever is optimal for the need at hand. For example, you might create three 100GB LUNs in a single group consisting of 42 drives, and place your SAPDATA files on these.
- Upon creating the LUNs, you can typically specify what RAID level each LUN is to use, and whether caching should be enabled or disabled (although in some virtualization product sets, this is simply not possible). In this way, if required of your solution, you can mix high-density RAID 5 LUNs (for database disk dumps, for example) with high-performance RAID 1+0 LUNs (for database data files and logs).

Some of the preceding points defy the preaching of many a database administrator or basis consultant. I can hear them now. “What, one giant virtual disk chopped up into pieces? You can’t do that, we’re supposed to keep our logs separate from our data! What happened to best practices?” Let them know that best practices differ now, based on the storage solution employed. And recommend that they read the PDF files found on the Planning CD that relate to configuring and optimizing, for example, HP’s Enterprise Virtual Array Storage System.

Finally, let the facts speak for themselves, as presented in Figure 10.10—the Virtual Array easily handles a variety of workloads, while other traditional high-end storage systems struggle.

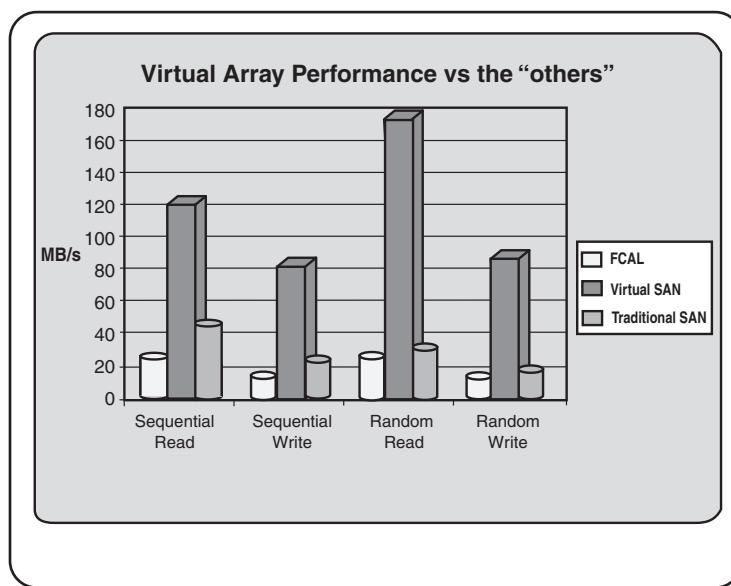


FIGURE 10.10 This chart shows the results of actual I/O throughput tests (measured in Megabytes per second), comparing a Virtual SAN Array to both a legacy FCAL-based SAP Production disk subsystem and a traditional high-performance SAN.

- ▶ For more hard data comparing the Virtual Array with legacy storage systems, see “Additional Stress-Testing Goals,” p. 613 in Chapter 16.

On the Road to Implementation

Before we close out discussions on designing and implementing the SAP Data Center, it’s important to look at how everything up to this point needs to be sustained. SAP deployments progress in phases or “waves”—each wave brings with it another collection of data center resources that must be planned for, installed, configured, and managed. For example, most new deployments start with bringing in the Technical Sandbox and Development systems. Later, the Test, QA, Integration, Staging, Training, and other environments are introduced to the Data Center. Eventually, the Production and Disaster Recovery systems are phased in as well.

Planning for SAP Data Center Operations

Before most of the SAP Technical Support Organization staff involved with setting up the SAP Data Center walk away, I need to cover some details related to managing the SAP Data Center. At this stage in the SAP project, we are more concerned with day-to-day and similar “tactical” operations than anything else. The strategic operational needs of the deployment will be addressed fully later.

I assume that an SAP-aware Enterprise Management Application has been deployed. That is, I assume that software agents are loaded on each SAP and SAP-related infrastructure server, and that a Management Console has been put in place to manage all SAP Data Center resources. I also assume that your Operations and SAP Basis teams have been trained on how to fundamentally use this all-important management tool, with the understanding that the learning curve is still pretty sharp over the next month or two.

In my experience, good SAP computer operations start with not only the use of an enterprise application, but also with *paper-based daily and weekly checklists*. Why the reliance on paper and pencil in this age of automated computing, and after spending tens of thousands of dollars on an SAP monitoring tool? Three reasons:

- These paper-based checklists will force SAP Operations to *learn how to effectively monitor an SAP environment*, using SAP-native tools like SAP CCMS—the same source of monitoring and management data leveraged by the popular SAP enterprise management suites.
- These checklists provide *backup*, should the Enterprise Management tool be unavailable (or the rolled-up or alert data unclear to the marginally experienced SAP support team).
- After a completed checklist becomes mandatory during shift-turnover, these checklists encourage accountability; they force the operations team to actually do the job of monitoring your mySAP solution.

Eventually, the paper-based checklists tend to lose their effectiveness from a training/knowledge perspective. At that point, the SAP Operations team often puts them back on the shelf, to be dusted off again later by new-hires and other new members of the support staff as they come on board. Some of my customers, though, continue to use their checklists long after their Enterprise Management tool is in place and functioning correctly, and the operations team is ramped up. They simply modify the checklists to better reflect the operational needs of the organization, or add the discrete tasks from the SAP checklists to the tasks and responsibilities of a larger-scope document maintained by the Data Center staff. And other accounts continue to use these checklists simply because they are effective; they still force a certain level of accountability, as long as the SAP Data Center Manager requests that they be used.

- ▶ To learn much more about operational and systems management activities in regards to SAP Data Center Operations, see “Back to SAP Operations,” p. 503 in Chapter 14.

Special Considerations When Deploying the Technical Sandbox

The first time SAP is installed in the Data Center will likely be in the Technical Sandbox (“sandbox” or “crash and burn system”) or similar testing environment. Oftentimes, the sandbox will look nothing like the Production system is expected to look. In the haste to get *something* running quickly, many SAP implementations introduce what can only be described as a stripped-down version of what will ultimately grow to be the actual Technical Sandbox.

The sandbox provides valuable hands-on experience to the SAP Basis, DBA team, and the SAP Operations team. It allows each of these technical groups the opportunity to see an SAP installation in action, and begin to realize the fruits of everyone’s labor up to this point. Further, it allows the entire SAP Technical Support Organization the chance to verify what it already should suspect as “holes” in the organization (these holes are addressed later in Chapter 12). Specific SAP components like Enterprise Buyer Pro and Enterprise Portal require knowledge and expertise in areas outside of simple SAP Basis installations, for example.

As the SAP Data Center grows and matures, the Technical Sandbox will grow with it. As a result, the following needs arise:

- Every SAP component needs to be installed and tested in the sandbox prior to installing it elsewhere. Thus, because it’s unlikely that anyone wants to maintain a sandbox with multiple SAP instances, robust tape backup/restore capabilities need to be introduced in the Technical Sandbox eventually.
- Every layer in the SAP Solution Stack must be represented within the sandbox. For instance, if a Web component like ITS (Internet Transaction Server) or Web Application Server will play a part in the Production system, it also needs to be represented in the Technical Sandbox.
- Finally, the SAP Operations team should begin to grow comfortable with managing and maintaining SAP, as I discussed earlier. This can be accomplished in a number of ways. For example, Operations should begin their “career” in SAP monitoring by starting with the Technical Sandbox. I’m talking about more than the SAP-aware Enterprise Management console. Other hardware/software-specific management consoles, like Insight Manager, HP Openview, Windows 2000 Performance Monitor, UNIX command-line performance utilities, and so on, provide must-have snapshot-in-time and historical trend-analysis value.

With the knowledge gained in performing the preceding tasks in support of the Technical Sandbox, the SAP TSO will be better positioned to address the needs of the next truly critical system within the SAP landscape—the SAP Development environment.

Special Considerations When Deploying the Development System

When the development system is installed and in place, for all intents and purposes it *becomes* a productive system—not for end users in the traditional sense, but for the expensive Functional experts and SAP ABAP programmers who will customize this system for many months to come. Therefore, the Development system needs to be treated like a Production system with respect to the following:

- Downtime—Bringing down the development system is unacceptable without plenty of warning. That is, a schedule for planned downtime must be developed, to address various operations and administrative tasks. These might include performing backups, executing client copies and refreshes, and so on. Why? First, because the SAP Functional experts and ABAP programmers are not cheap, and their time is therefore valuable—without a development environment, these folks will only sit idly by, still billing hourly fees that can easily add up to tens of thousands of dollars for the entire development team. Second, because downtime results in implementation schedule slips, and costs even more down the road.
- Monitoring—A proactive approach to monitoring the development system needs to be put in place immediately, both to minimize unplanned downtime and to quantify that the development system is performing well.
- Backup/Restore—It is absolutely critical that the work being performed by the development team can be safely backed up to tape, and preserved offsite in the case of a disaster. Why the seriousness? Because the work that goes into the development system to prepare for a productive system can easily equate to many millions of dollars in productivity.

By now, you should be well into your SAP implementation. In the next chapter, I will step back and discuss in detail the SAP Basis installations particular to most of the popular SAP components—R/3, BW, APO, CRM, EP, and more.

Tools and Techniques

For this chapter, the Planning CD contains the following files:

- A “Rack Best Practices” checklist, with tips and hints on how to best rack and stack your enterprise computing gear.
- Two documents in PDF format—these detail some of the configuration and performance options related to deploying HP’s Enterprise Virtual Array storage system.

- Daily and weekly SAP Operations checklists.
- Each of the figures found in this chapter, in Microsoft PowerPoint format, which include highly available network solutions, disk subsystem solutions, a SAN design template, and more.

Summary

Chapter 10 covered nearly everything that needs to take place or be addressed when it comes to developing your SAP Data Center. Here, we worked our way up the SAP Solution Stack, stopping along our implementation roadmap to discuss data center physical requirements, power and cooling considerations, the need for a robust network infrastructure, setting up racks and servers, configuring traditional disk subsystems and even the newer Virtual SAN Arrays, and more. And we also touched upon some of the basic operational activities that need to be addressed after all of these data center computing resources are in place.

11

Performing mySAP.com Component Installations

Implementation Methodologies for SAP Installations

In this chapter, I will begin by identifying various implementation methodologies and how each relates to SAP component or technology layer installations. I will then walk through installing common underlying or integration products, such as SAP's Web Application Server and Internet Transaction Server, and then move into detailed mySAP.com installations. The chapter concludes by outlining the tasks and activities required to elevate a vanilla installation to a state ready for business process configuration.

SAP's first real implementation methodology was *ASAP*, or *Accelerated SAP*, launched in 1997. The tools and methods introduced were primarily template-based, but were a big improvement over the "one-off" approaches leveraged for implementations until that time. In terms of actual product installation support, however, ASAP was pretty weak; its accelerators focused almost exclusively on functional configuration and other tasks associated with what SAP AG termed *Realization*. Only in one of the work packages associated with ASAP's Blueprinting phase was installation briefly mentioned. During this time, customers turned to SAP technology partners and tools outside of ASAP (like SAP's SAPNET resource) to fill in the holes in the "installation gap." SAP technology partners in particular spent much of their time creating custom technology stack checklists and recipes, leveraging SAP's R/3 product documentation as a basic starting point.

IN THIS CHAPTER

- Implementation Methodologies for SAP Installations
- Rounding Out Your SAP System Landscape
- Leveraging Installation Documentation, Tools, and Approaches
- Installing Your mySAP-Enabling Foundation—NetWeaver
- Installation Details for mySAP Components
- Addressing General mySAP Post-Installation Tasks
- Tools and Techniques

ValueSAP and the Global ASAP Roadmap

With the advent of SAP Basis release 4.6C in 2000, Global ASAP was introduced under the ValueSAP framework, which was divided into the three core phases of an SAP product's life cycle. The middle phase, *Implementation*, was front-ended by *Discovery and Evaluation* and backed up by ongoing *Operations and Continuous Improvement*. It was here in the Implementation phase that more attention was given to SAP installation processes. ValueSAP's tagline of "reducing time to benefit" fit well with the still-new mySAP.com initiative, too, as more and more global implementations sought to leverage best practices and installation approaches to speed up implementations.

With regard to implementation support, the *Implementation Assistant and Roadmap*, backed up by a Q&A database and the SAP *IMG*, served as the primary tools for enabling faster mySAP solution installations.

NOTE

The IMG is a tool for configuring your system; for each business area, IMG walks you through all the steps in the implementation process. In doing so, it communicates the SAP "standard" or factory settings, and describes system configuration activities. The hierarchical structure of the IMG makes it valuable, as it reflects the structure of each mySAP application component.

Included in these tools were SAP's published best practices and approaches to implementation, and something called the "SAP Reference Model," which identified processes and roles that supported the delivery of mySAP solutions. Global ASAP was still largely template-based when it came to installation methodologies, though, updated and enabled to support global SAP implementations (compared to the original ASAP methodology and its smaller scope). Still, attention to the following issues helped to speed up the pre-installation tasks necessary before actual SAP product installations could commence:

- Implementation strategy
- Global system topology
- Organizational change management
- Global business process standardization
- Development of *Global Elements*, such as documentation templates, standard IMG structures and settings, master data formats and documents, and so on

Proper attention to these issues allowed systems to be rolled in more of a "cookie cutter" fashion, based on global reusable templates. By doing so, consistent implementation standards and lower related costs were realized, and the overall quality of the implementation increased.

Additional new or improved tools were also introduced during this time frame, and also helped to save time in installation and post-installation processes. For example, the much-improved Transport Management System (TMS) replaced the less-capable Change and Transport system, and SAP's Computer Aided Test Tool finally facilitated load testing as well as functional testing. Site preparation got more attention, too, especially with regard to providing training to the teams tasked with designing and installing the mySAP system. Building a standard site-specific training curriculum was promoted, and the idea of developing repeatable local implementation processes finally took hold at the SAP Basis level. Comprehensive installation and master implementation guides were developed. All of this attention to speeding up an implementation and therefore rapidly achieving ROI set the stage for the next wave of implementation support to take hold of SAP—the SAP Solution Manager.

SAP Solution Manager for Implementation

According to SAP AG, by the end of June 2003, ValueSAP/ASAP training will no longer be available. With the ability to order ValueSAP CD-based content already a thing of the past, and SAP Solution Manager (SSM) training related to implementation ramping up throughout the first half of 2003, the writing on the wall is clear—SSM will be *the* key service and support enabler going forward (along with SAP Service Marketplace). SAP Solution Manager is touted as both the implementation and operations platform for mySAP solutions. The former is the topic of discussion here, and I cover the latter in Chapter 14. In terms of implementation, it is the technical implementation aspects that we are actually most interested in; functional and operations-enabling aspects are extremely valuable but do us little good when it comes to the actual installation of mySAP.com components. Key benefits of using SAP Solution Manager for Implementation include

- SSM acts as a central point of access and support for key implementation activities, and finally (!) includes an “Enhanced Solution Management Roadmap” targeting the Technical Implementation Team (SAP TSO, and its various technical consultants).
- SSM provides a process-driven approach to blueprinting, technical configuration, and testing. Combined with standard implementation scenarios, this streamlines everything from sizing through installation.
- SSM includes built-in project monitoring and reporting capabilities, and a central repository for storing project documentation and issues.
- SSM also identifies services and processes aligned to ensure both an uneventful Go-Live and excellent ongoing operations.

One way that SSM for Implementation delivers is through *Ramp-Up Knowledge Transfer* (RKT). With RKT, initial competence in delivering a mySAP solution is

provided through Web-based and CD-based content. For instance, Solution Manager Learning Maps can be downloaded from <http://service.sap.com/rkt-solman>. And a formal SAP-sponsored training class, SMI310, is available as well, covering SSM's implementation tools in detail. Another class, TSML10, even addresses Solution Manager infrastructure and installation. Other resources offered by the SAP Solution Manager for Implementation are directly accessible from SSM:

- The Enhanced Solution Management Roadmap and other Roadmaps describe how to organize and run your mySAP implementation project.
- A Business Process Repository outlines and describes your specific mySAP.com solution (this is part of the built-in monitoring and reporting capabilities).
- Integrated Business Content is available, offering scenario-based access to generic technical descriptions and collaborative Business Maps.
- Access and integrated use of customizing tools, along with best practices for implementing mySAP.com components, is available.

Both version 2.2 and 3.1 of SSM sit atop Web AS 6.20. Support for RKT and Implementation was introduced in SSM version 2.2 (the previous version 2.1 only supported Operations), but with release 3.1 of SSM, Implementation and Operations are finally wrapped up in one Web AS-enabled product. Given this, I look at SAP Solution Manager now as just another core product that must be maintained throughout a customer's mySAP system landscape—a “development/testing” environment is recommended (for testing updates/enhancements and upgrades), along with a “production” system to be used by the SAP TSO to actually support implementation and operations processes. In terms of hardware needed to support this basic two-system landscape, though, requirements are minimal. SAP recommends a dual-processor server with 1GB of RAM and 50GB of disk space at minimum, and in my limited experience, this seems more than adequate.

Rounding Out Your SAP System Landscape

In Chapter 10, we concluded our SAP Data Center activities by installing the Technical Sandbox and Development systems. Actual database and mySAP component installation details were not covered, though (they can be found later in this chapter). Instead, the assumption was that we would finish installing the rest of the SAP system landscape as dictated by milestones outlined in the SIPP, our master project plan, spread throughout the remainder of the project.

In my experience, a number of training systems or instances are often installed fairly quickly after Development is installed. These include the developer's training system, and perhaps an SAP Knowledge Warehouse or similar instance. Next, a month or two later the Test/QA system is often installed, followed by another system or instance

earmarked for end-user training. Finally, a Staging system and/or the final production system(s) are installed after another few months. If a staging system is put in place, it usually precedes production by a couple of months; stress testing often takes place on the staging system in this case (rather than production), the results of which tend to change the production configuration. That is, after the stress testing is concluded, the production system's design is fine-tuned accordingly, and the system is finally ordered from the hardware partner.

Speeding Up Your OS Installation Process

Installing each successive SAP system within the system landscape takes time. Because of the repetitive nature of the installation process, a cookie-cutter approach to installation makes sense, as do tools and utilities that help speed up the process. Scripted OS installations using Microsoft's Setup Manager Wizard (`setupmgr.exe`, found on the W2K Server CD under the Support folder) are popular for Windows-based servers, as are disk imaging products such as Symantec's Norton Ghost and PowerQuest's DriveImage. To connect your server to an Installation Server containing your OS installation folder or image files, though, you'll need a bootable floppy or CD with the requisite network, memory, disk management, and other drivers. These are easily created, but can be time-consuming in and of themselves, which is contrary to the mission at hand. So to speed up this process, I recommend *Bart's Boot Disk* Web site, accessible at <http://www.nu2.nu>.

- ▶ To review Operating System considerations with regard to installing a mySAP component, see "Operating System Best Practices for SAP," p. 364 in Chapter 10.

Preparing for an SAP/Database Installation

After the hardware and operating system are installed, the real fun begins, for now we can finally begin preparing for an actual mySAP installation. Novices take note: an SAP installation includes (and is very much tied to) a database load. Thus, when I speak of installing a mySAP component, I am also including the database layer. SAP's products are database-specific; you order or download the installation kit for R/3 Enterprise, for example, only after specifying a particular database platform. As R/3 Enterprise is available for IBM DB2, Informix, Oracle, SAPDB, SQL Server, and a few other database platforms, you must specify R/3 Enterprise for SQL Server, for instance, to receive the correct SAP and database-specific installation media.

Further, with very few exceptions, you must actually use the database CD or media that shipped with your SAP product. Doing otherwise (for example, using a vanilla off-the-shelf copy of SQL Server 2000) will not work—SAP updates the database CD that they provide to you, writing hooks into it to facilitate the installation process. By the same token, even your SAP installation CDs are also tweaked specifically for SQL Server. Thus, trying to install an Oracle database with SAP R/3 installation media tweaked for SQL Server will never work, either.

When you have the correct installation media, exactly how do you go about performing an SAP installation, though? The answer lies in SAP's installation tool, SAPinst.

System Landscape Implementation Manager—SAPinst

The SAP System Landscape Implementation Manager, or SAPinst, is SAP's installation tool of choice for new products and mySAP components. Do not be tempted to simply execute the tool, though, until you address all of the installation planning and preparation required for a successful outcome. You might be loading SAP in a Windows environment, but the process is a bit more complicated than double-clicking a Setup icon and walking through a couple of "Next" screens! To that end, I've included a section later in this chapter dedicated solely to planning and preparation.

SAPinst appears similar to some of SAP's older installation tools in that it's GUI-based and has much of the same familiar look and feel. However, it offers quite a few advantages over previous installation utilities and approaches, such as R3Setup (used to install SAP Basis 4.0B through 4.6D systems) and R3INST (4.0A and prior releases through 3x). For example:

- SAPinst does not abort due to errors; instead, the installation stops and you are given the opportunity to fix any issues. Afterwards, the system installation is simply restarted where it ended (the point of failure).
- You can also step "backward" at any time during the installation to correct or change your entries, without having to restart the installation.
- The installation is logged in a single log file (`sapinst.log`), rather than multiple log files as has been SAP AG's custom. In this way, it's easier to track and analyze installations after-the-fact.
- Because the graphical user interface (GUI) portion of the installation tool is Java-based, SAPinst has a very consistent look and feel across multiple components and even operating systems.

Because the GUI is Java-based, a Java Runtime Environment (JRE) must be installed before the actual component installation can be started. This is accomplished by installing a Java Development Kit (Java 2 SDK, Standard Edition). A JDK is unfortunately not generally included with your installation CDs or media. I have successfully used the JDK that ships with SuSe Linux Enterprise Server 7 (Blackdown JDK 1.1.8) for Intel-based installs, and heard good things about 1.1.6 for Sun Solaris environments.

If you have a JDK upon which your SAPTSO or general IT organization has standardized, I suggest trying it—it will probably work just fine. To be sure, check with

<http://service.sap.com/platforms> and drill down into Availability for SAP Components in Detail > SAP Web AS/Basis/Kernel > Planned OS/DB/JDK Releases. Note that the specific location of and access to this Web site have changed a number of times, and will probably continue to change with every release of Web AS; worst case, execute a search for JDK from the /platforms site.

Leveraging Installation Documentation, Tools, and Approaches

SAP has provided us with an enormous amount of product- and component-specific documentation. Much of it is quite good. Even in the best of cases, though, to install an end-to-end mySAP solution you will still require product documentation (or very deep experience) relevant to your entire SAP Solution Stack. I find this to be especially true at the OS level—even the best SAP Basis consultants will run into a brick wall the first time they try installing a mySAP component on an operating system with which they are unfamiliar. With a background primarily in Windows-based SAP installations, I speak from experience, as I remember well my first installations on Red Hat Linux, SuSe Linux, Tru64, and HP-UX.

There is less of an implementation challenge at the database layer, surprisingly. SAP has integrated the mySAP/database installation process quite well, leaving little to be deduced during the actual installation. That explains why my first Oracle, DB2, Informix, and SAPDB installations for SAP were much less traumatic than the Linux one.

In all cases, though, to successfully prepare yourself for an SAP installation means doing your homework. There is a lot of documentation to read; doing otherwise means there will only be even *more* to read, though, as you try to dig your way out of a failed install. In the pages that follow, I will share with you all of the background reading, preparation, and planning you need to do, to get through the installation process quickly and accurately.

Do Your Homework

I cannot tell you the number of otherwise intelligent SAP consultants I run into who still feel compelled to learn “the hard way” and try installing an SAP product without the benefit of doing their homework. If one thing is true of an SAP installation, it’s that it is fraught with potential pitfalls and late hours. In the best case, an “easy” end-to-end manual installation still takes a full 24–32 hours, or three to four business days. This can easily stretch to more than a week, and even longer, though, if too many incorrect assumptions are made or shortcuts are taken. Consider the following points:

- Not all installation processes are the same, or even similar. In some cases, you might be asked to install the database layer first, for example, whereas in other cases you might need to start by installing the SAP Central Instance. It all depends on the component, the operating system upon which you are installing it, the specific RDBMS being implemented, and whether the system is a Unicode or non-Unicode version (see the Note that follows this list).
- Preparation and planning differs between different releases and products, and is also OS- and database-specific.
- Prerequisites will differ as well. With each new release of a mySAP.com component come not only new OS and database minimum requirements, for example, but also minimum releases related to other mySAP solutions, enabling technologies, and so on.
- Post-installation tasks are not only critical, but differ between releases, too. This underscores the importance of fully *completing* an installation such that it may actually be useful to the functional configurators and ABAP/Java developers who need to get to work themselves. Leaving a new mySAP installation without setting up the transport system, or tying it into another required system, for example, is akin to wasting time—the end result is simply not useful to anyone.

Thus, doing your homework equates to reading the various installation guides and product documentation relevant to your mySAP solution. I cover these resources in detail next. If you are not fond of reading, I suggest that you find someone to do it for you, or maybe find another line of work.

NOTE

Unicode refers to the encoding system for characters used in a computer. Unicode unifies all characters of all character sets (languages) into a single encoding scheme. Thus, all letters, numbers, and other characters in a Unicode system are uniquely represented by the computer; each maintains its own unique numeric representation. Doing otherwise requires transforming between different encoding schemes running on different computers, and therefore represents a potential risk in terms of data loss or corruption. In SAP, non-Unicode systems use characters that are represented in binary with only one byte, whereas Unicode systems represent characters in binary with two or four bytes. See SAP Note 79991 or <http://service.sap.com/unicode> for details.

SAP Master Guides

For complex mySAP offerings like SRM and CRM, SAP publishes a “Master Guide,” which provides general information related to architecting and installing these

multiple-component solutions. Because of this, SAP describes these documents as “Central Overview Documents.” And they are some of my favorites, especially when I am tasked with installing a solution with which I have had little or no previous experience. For example, the CRM Master Guide for 3.1A includes the following valuable information:

- *Getting Started*, which includes prerequisites, information on tools and topics relevant across the solution, a history of changes to the mySAP offering, relevant SAP Notes, and so on.
- A CRM *Software Component Matrix*, of which I am a big fan. SAP’s software and technical matrices help you identify the specific versions of products and other technology layers that interoperate well with the mySAP product (CRM in this case). The SAP-required CDs necessary to install the mySAP solution are also noted in this section.
- *Technical Information*, covering individual components such as E-Selling, Field Sales, Interaction Center, Customer Service and Support, Field Service and Dispatch, Marketing Management, and so on.
- Detailed set of *Appendices*, which address complementary products and also point you in the right direction to obtain more detailed documentation.

I have found the master guides to be more than simply valuable—they help me to build a rapid knowledge foundation in a solution. That is, they provide the high-level overview information necessary to understand how a solution or product fits into the mySAP “big-picture” scheme of things. But at the same time, these guides are “deep” enough to keep me from having to waste a lot of time surfing the Net when time is at a premium. Thus, I enthusiastically recommend them as a precursor to the more detailed information that is only found in SAP’s product-specific Installation Guides, discussed next.

SAP InstGuides—Traditional Installation Guides

For each mySAP component or technology solution, SAP AG publishes a standard installation guide, referred to as an *InstGuide*, and makes these guides available over the Web at <http://service.sap.com/instguides> or via CD-ROM-based media. Master guides are also made available through this same URL, as are other guides, installation documentation, customer letters for new-product introductions, and more.

Generally speaking, InstGuides come in two varieties:

- Detailed New Installation Guides. These installation guides are specific for each OS, database, mySAP component, and release. For example, you can download an installation guide for installing SAP Business Information Warehouse 3.1A on Windows 2000 Server, for SQL Server 2000.

- Upgrade Guides. These are specific to a certain set of releases of an SAP product and database. Common guides include those that walk you through an upgrade from R/3 4.5x to 4.6x, or 4.6x to Web AS 6x, for instance, leveraging an Oracle database platform.

Installation guides are built around best practices for installing mySAP.com components. And they are detailed! Really detailed, in fact, to the point where nearly all of the pre-installation, installation, and post-installation tasks necessary to make a mySAP.com component ready for developers are included in a single document. Anything not explicitly explained in the InstGuide usually references a link to the SAP Service Marketplace, or points to a specific location within an online help resource.

Addressing Pre-Installation Tasks

Perhaps fully a quarter to a third of an SAP InstGuide is dedicated to the tasks and activities that must be performed *before* the SAPinst tool is ever actually executed. Pre-installation tasks cover:

- *Pre-reading*, including the need to look through other documentation published by SAP or in some cases other vendors, SAP Notes that need to be reviewed, and more.
- *Installation Planning*, including minimum or recommended specific hardware and software requirements that support the particular mySAP component to be loaded, and considerations related to the database (like physical disk layout), SAP directory layout, use of the Multiple Components One Database (MCOD) approach to installation, and more.
- *Installation Preparations*, such as checking your file system, validating network/domain infrastructure, optimizing the OS (reducing the size of the file cache in Windows environments, for example), checking/modifying the kernel (UNIX only), creating OS users and addressing user rights/permissions, sizing swap space or pagefile, selecting an SAP System ID (SID) and host name, copying specific installation CDs to disk, installing a Java Development Kit (JDK) to support SAPinst, and more.

After pre-installation tasks are performed, you are finally ready to run SAPinst and install your SAP system. I cover installing the most common mySAP components later in this chapter.

Post-Installation Tasks

Just like pre-installation tasks, post-installation tasks are plentiful and take up a lot of space in each InstGuide, too. After a mySAP component instance is installed, you

still have much to do prior to turning the new system over to developers and programmers. The InstGuide is useful in this respect, covering post-installation tasks such as obtaining and applying the software license, setting up single sign-on, installing and configuring the transport management system, setting up printers, planning for background jobs, applying support packages, and more. And each InstGuide also includes a number of handy how-to's, such as how to start and stop the SAP system, how to log in to the system, and how to install online documentation and additional language support. It also details how to perform a client copy, and even addresses some basic operational tasks, like performing a backup. Some of these are covered in depth at the end of this chapter. For other post-installation tasks, refer to the InstGuide, its pointers, supporting SAP-provided documentation, and Chapter 17.

Custom Checklists and Recipes

As I have mentioned a number of times elsewhere, custom installation checklists, or "recipes," prove invaluable to a Technical Support Organization throughout the life of an SAP system. A good recipe will cover the solution-specific characteristics unique to your particular installation, and therefore provide value far beyond that provided simply by InstGuides and Master Guides.

A good recipe also promotes my goal of creating repeatable processes, which ultimately saves time when you have little to spare, as in the aftermath of a disaster. And recipes prove themselves valuable in supporting day-to-day activities as well, for example, when you need to restore the production environment in your technical sandbox to re-create a problem or troubleshoot an issue.

- ▶ To learn more about the role recipes and checklists play in supporting Disaster Recovery, see "The Disaster Recovery Crash Kit," p. 210 in Chapter 6.

In the past, I have created end-to-end custom installation checklists that detail hardware, OS, Database and mySAP component specifics. Rather than re-creating the wheel, however, I used an umbrella approach to documentation, and called the resulting Microsoft Word document a "Delta Guide" (because it brought together the processes discussed in multiple other documents, but only detailed what was missing in these other documents—the deltas—to pull off an end-to-end installation).

A Delta Guide is my own version of a master guide, and may include embedded attachments, though I prefer to reference documents published elsewhere (like a URL on a Web site hosted by SAP, Microsoft, Oracle, HP, and so on). The Delta Guide therefore brings everything together in one place, one master document. The core of the Delta Guide is a Microsoft Word document. It's the master umbrella, if you will. I provide a short overview of the purpose of the document and illustrate the flow or sequence of events, and then into this document I embed or link to objects like those that follow:

- A “big picture” document, like SAP’s Master Guide for the particular solution, or a custom-developed guide for a complex mySAP environment. I embed these PDF or Word documents directly into the Delta Guide. Note that a Web link works well, too, but you risk the Web site owner updating or removing the document. To mitigate this risk, I suggest housing the documents on your company’s Web site—the site used by the SAP TSO to support Disaster Recovery often makes the most sense all the way around.
 - A hardware configuration document, detailing the server configuration—server model, number and type of processors, memory configuration, details regarding the I/O busses and what resided in each, and so on. In the case of HP/Compaq ProLiant servers, for example, I might simply embed a “survey.txt” file (which includes all of this data and more, easily created via the HP Survey software utility) into the Delta Guide.
 - A disk subsystem configuration document, detailing the layout of drives, type of drives, RAID levels employed, type and number of disk controller(s), controller configuration settings, cache settings, and so on. Usually it’s possible to “dump” a disk subsystem’s hardware settings into an output file or log. I embed this output file into the Delta Guide, edited to reflect any high-level data that might also prove useful.
 - Operating system installation details, including version and patch levels of the OS, logical disk layout, network configuration details (IP addresses, DNS/DHCP settings, and more), system-wide settings, and so on. Most often, I build a 20–30 page document of screen shots and similar images, and embed this into the Master Guide.
 - Database configuration document, which captures the size and layout of data, log, and executable files, database block, cache, buffer, and other settings, and more. Like the OS details, this is often best captured through screen shots.
 - mySAP Component InstGuide, which walks you through a product installation from an SAP perspective end-to-end. I like to download these PDF documents and embed them into the Delta Guide, along with any supporting technical documents that might be referenced by the InstGuide (like SAP Notes).
 - Other documents that help document the solution, like anything related to a specific solution component or integration vehicle. In the past, I have included details relevant to an SAP ITS installation, EBP details related to installing and configuring Requisite BugsEye and eMerge, and so on.
- To read more about Delta Guides and similar approaches to documenting installation processes and other repeatable procedures, see “Documenting Daily Operations and Installation Procedures,” p. 507 in Chapter 14. An actual Delta Guide sample is also included on the Planning CD.

Smart Implementations and the SAP Configuration Assistant

I believe that one of the biggest things to come out of SAP AG that addresses technical implementations is the idea of a *Smart Implementation*. Smart Implementations support unattended installation and basic configuration of your mySAP technical solution—not the application customizing that takes so many months or even longer to complete, but the technical installation and customizing that often takes many *days per component* otherwise.

Smart Implementations can facilitate both planning and installing an end-to-end mySAP technical infrastructure, handling the installation of your unique mySAP components and even configuring and integrating these components into your larger existing SAP system landscape. And Smart Implementations aid you in managing all of this infrastructure as well.

The power of Smart Implementations lies in the following:

- Connectivity to the Internet gives Smart Implementations access to the information it needs to ensure that your SAP components are mutually compatible, noting different configuration requirements and platform dependencies.
- The *Configuration Assistant*, an Internet-enabled tool, facilitates gathering the information necessary to later install and configure your SAP system landscape; your system landscape's technical infrastructure is set up according to the specifications outlined by the Configuration Assistant.
- A single implementation tool, the System Landscape Implementation Manager, rather than multiple tools and approaches, is used to install all applicable mySAP solutions. The actual installation process is performed with the SAP System Landscape Implementation Manager, which supports many current and all forthcoming mySAP components and platforms. And this tool also supports installing multiple components in a one-step process, as illustrated in Figure 11.1. By leveraging the Configuration Assistant's output file and the latest InstGuides, Master Guides, and other tools, SAPinst can automatically perform complex mySAP-specific and cross-component scenario installation processes, ultimately making for a smooth and repeatable technical implementation.
- Each mySAP component is automatically configured with “default” values based on experience gained through SAP's many successful implementations. Unless you override them, each installed system is set up using default values proven over time to be reliable and stable.

It is the SAP Configuration Assistant specifically that enables you to save days of research, installation, and configuration effort. With the Configuration Assistant, you can easily begin planning the design of your specific SAP system landscape and then capture all of this data in an XML-based configuration file. This configuration

file feeds the next phase of your Smart Implementation, where each of your identified mySAP components is installed and configured automatically—saving a lot of time later on down the road, too, as common configuration problems are avoided altogether.

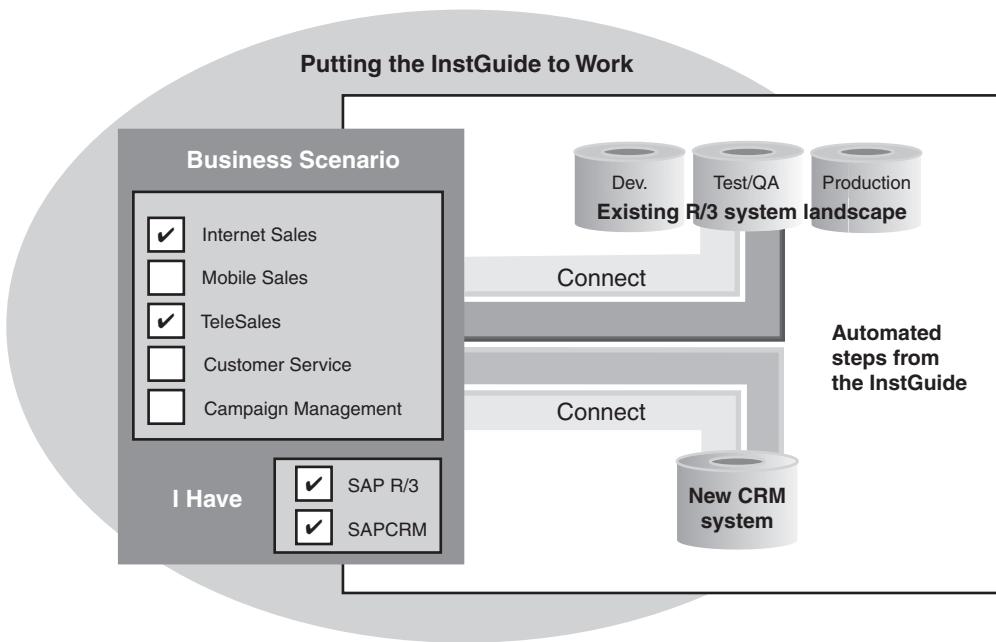


FIGURE 11.1 The InstGuide for CRM is put to work, preparing for an automated mySAP.com CRM installation.

The SAP Configuration Assistant is very easy to use, but very powerful as well. Its graphical user interface makes this system landscape planning tool perfect for designing and configuring an SAP system landscape unique to your needs. At a high level, you simply select the applications and components to be installed (limited to Workplace and BBP/CRM), define how these products are distributed across different hosts, and as necessary integrate additional components into the system landscape, as shown in Figure 11.2. The specific steps are as follows:

- Define each component, host, and/or scenario to be installed.
- Specify the database and OS combinations upon which each system will be installed.

- Provide parameters relevant to each component, such as the SID to be used by the system, languages to be installed, port numbers over which to communicate, and so on (standard installation questions that do not lend themselves to “default” values).
- Drag and drop software on to your hosts, thus distributing the software.
- Define integration requirements to back-end systems, which may also include the development of specialized installation scripts for systems outside the realm of mySAP solutions.
- Define any front-end settings, such as the name and characteristics of logon groups, how load balancing will be employed, and so on.

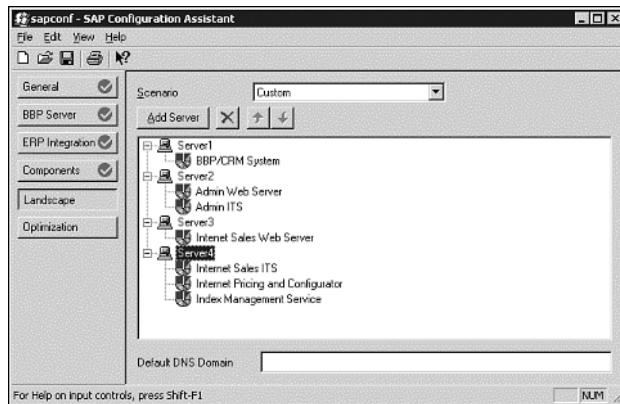


FIGURE 11.2 The SAP Configuration Assistant allows you to allocate mySAP technical components to servers and then save the design in an XML-based configuration file.

The Configuration Assistant is easily installed and executed on any Windows-based computer—I run it on my laptop, in fact. This makes it simple for me to architect, plan for, and discuss what a potential SAP system landscape might look like for one of my customers, and allow me to prepare a conceptual design on an airplane, or change it later from my hotel room, without actually being connected to a physical installation. Alongside my custom SAP sizing tools and lessons-learned from several hundred implementations, it has proven itself in the past to be one of the most valuable “customer-facing” tools I carry.

Using SAPinst for Unattended Installations

The System Landscape Implementation Manager is truly amazing, enabling an integrated and even unattended approach to planning, installing, and configuring each component. If you consider the technical challenges inherent to installing a single

product, much less a suite of SAP products—the communication interfaces, inter-component protocols and network settings, technical configuration of SAP itself, security settings, user accounts requirements, and so on—you will begin to understand the significance of this tool. It's huge!

The System Landscape Implementation Manager supports two different operation modes—Smart and Manual. Recommended for standard installations, “smart” mode executes a comprehensive set of configuration steps during the installation process in a cookie-cutter, repeatable manner. As a result, the Smart Implementation itself reflects a high-quality, repeatable implementation, and the time required to plan, install, and configure your SAP system landscape is reduced simply because many error-prone installation and configuration data-entry tasks are avoided. This is accomplished by using the previously discussed InstGuides and the Configuration Assistant’s XML-based output file, as illustrated in Figure 11.3.

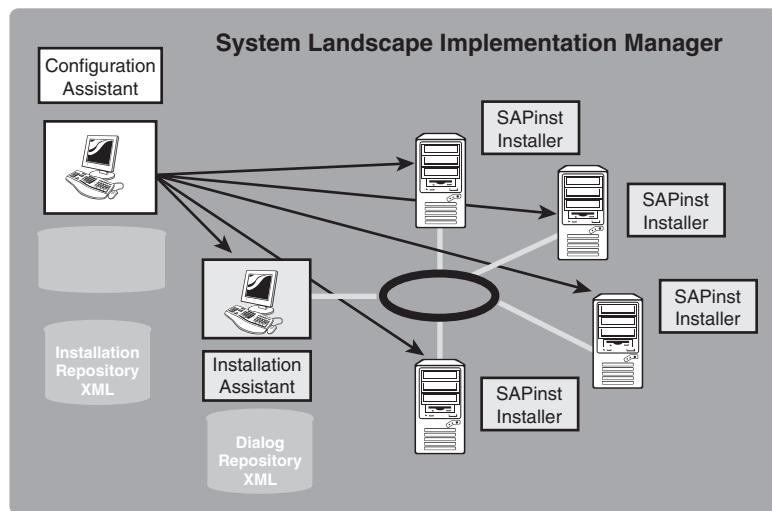


FIGURE 11.3 Leveraging the Configuration Assistant’s XML output file and other InstGuide data, SAPinst now has the input information necessary to perform an unattended installation.

In “manual” mode, on the other hand, you are granted greater flexibility in making changes to installation and configuration settings, but have to sacrifice speed and probably repeatability of your install because most of this flexibility requires manual entries. And because of this, manual mode does not support an unattended installation or configuration of your SAP components, either.

Installing Your mySAP-Enabling Foundation—NetWeaver

SAP AG has made it clear in the last year or so that certain technology solutions and approaches will constitute the foundation or enabling technology for mySAP.com components going forward. In January 2003, SAP AG went so far as to introduce SAP NetWeaver, the next generation of mySAP technology that underpins SAP's Enterprise Services Architecture (ESA), which in turn enables SAP's customers to build and deploy Web-services based business solutions. NetWeaver solves the dilemma of Microsoft .NET versus Java that many companies face today—it embraces both. In doing so, SAP AG provides both a platform and a middleware solution that truly facilitates heterogeneous integration. Check out

<http://www.sap.com/solutions/netweaver> for more details, and refer specifically to the Planning CD for a link to SAP's white paper entitled SAP NetWeaver Platform Interoperability with IBM WebSphere and Microsoft .NET.

Though the label may have changed recently, the fundamental technology enabling mySAP solutions include

- Web Application Server (Web AS)
- SAP Exchange Infrastructure
- Enterprise Portal

I include Internet Transaction Server (ITS) in this list, too, at least until ITS is completely replaced by Web AS-enabled solutions. Each of these is discussed in terms of installation details and considerations next.

Web Application Server 6.x

The underlying technology layer of nearly all mySAP solutions, and the technology foundation beyond SAP Basis release 4.6D, is referred to as Web Application Server, or Web AS; everything prior to Web AS is simply known as "SAP Basis." Given that Web AS is a superset of the former SAP Basis, experienced SAP Basis consultants should be quite comfortable installing and configuring Web AS—the core application-enabling technology layer is merely broader and extended, not completely revamped. Note the following:

- Most of the mySAP.com components on the market today were released "on top of" Web AS 6.20. The newest components are being released for version 6.30.
- Web AS supports Unicode and non-Unicode systems. However, you must specify the desired version of Web AS when you download it or order the CD media; they are not interchangeable.

- Like support for Unicode, Web AS is also database-specific. Thus, you must order the SQL Server version of Web AS, for example, if you want to install Web AS on a Microsoft SQL Server platform.

The installation process for Web AS is straightforward after the pre-installation planning and preparation tasks are completed. If we assume a SQL Server-based installation of a central system, the high-level installation tasks for Web AS 6.20 include the following steps:

1. Install the Microsoft SQL Server database server software.
2. Run SAPinst to install the Central Instance.
3. Answer all of the installation questions posed by SAPinst during the first few minutes of the SAP installation. SAP recommends that you obtain the required information needed to complete an installation before you actually run SAPinst.

Installation questions include selecting the installation type (that is, central instance, dialog instance, or database server), entering the three-character SID, determining whether the system should be integrated with Microsoft's Active Directory, and designating LDAP/SAP MMC support. Then parameters like the database server's host name, port number of the CI (message server), configuration of the database's TempDB and Data files, and the password for the SQL Server login must be provided. The number of parallel jobs/load processes is requested next (this should be equal to the number of CPUs in your server). Then you can select whether MNLS (Multi National Language Support) is desired.

Configuration data for the J2EE engine is requested next (if you are installing this), including the maximum memory to be used, J2EE client password, and location of the Java Development Kit on your host. After this, the actual installation takes place, ranging in time from an hour to perhaps three or four hours, depending on the number of processors in your host and the configuration of your disk subsystem.

The SAP Exchange Infrastructure 2.1

SAP XI enables collaborative business processes, using XML and a message-based architecture to create an open standards-based Web-centric solution. From an installation perspective, five major components can be installed, all of these natively supporting the SAPDB database (though Oracle is supported on all but the Integration Builder). And all of these components are based on a Unicode Web AS installation:

- *Integration Builder* (IB) and the *Integration Server* (IS), the former of which is only supported on the central instance of a system running Unicode Web Application Server 6.20 on top of SAPDB.

- *SAP XI Add-On*, which must be installed on the IB and IS.
- *MarketSet Adapter*, which can only be installed on the IS and further must only be used with SAP SRM solutions greater than version 2.0.
- *System Landscape Directory (SLD)*. Note that other databases are supported for this particular component, too.
- *SAP XI Connectivity*, which is automatically installed with the Integration Server. Adapters can be installed on the various business systems to which you will be connected. To connect systems separated by a firewall or running mixed operating systems, the adapters *must* be installed on the business systems.

As indicated earlier, only Unicode versions of Web AS 6.20 are supported for XI 2.0. Further, SAP only supports a dedicated instance of Web AS—you can *not* share an instance created for BW or APO, for example, with XI. From a high-level perspective, prior to actually running the SAPinst installation process, perform the following tasks:

1. Address “Installation Planning”; work with the XI project team to understand the scenarios to be implemented, components to be installed, and layout/architecture of the system hardware upon which XI will be installed.
2. Next, read through prerequisite documentation and installation guides, including SAP Note 557414 (<http://service.sap.com/notes>).
3. Prepare for the installation by installing a separate SAP Web AS 6.20 as a Unicode system running the SAP J2EE Engine (do not use an existing Web AS running another component).
4. Do not install the SAP Online Documentation CD. You will install an XI-specific version of this documentation later in the installation process.
5. Update Web AS up to and including Support Package 11. For the latest Support Packages and binary patches (executable updates), go to <http://service.sap.com/patches>, and then navigate to SAP Exchange Infrastructure - SAP Exchange Infrastructure 2.0.
6. Set up user SAPJSF in client 000, if it does not exist already (verify that it exists by trying to display it, using transaction SU01, for example). Note that the password for SAPJSF must not exceed eight characters, either.
7. Install the XI add-on. This *must* be done before you execute the next step and copy client 000.
8. Create a specific client for XI.
9. Apply the Delivery manager update, as explained in SAP Note 532892.

10. Install the Support Packages for Java Libraries.
11. Apply patches to SAPDB and the Java connector as required. The reasons are specific, and are explained in SAP's InstGuide for XI 2.0, published January 21, 2003.
12. Configure the SAP J2EE engine; disable FastRFC by changing the entry in the "Properties" file from "RFC_Default_Destination= " to "RFC_Default_Destination=false".
13. Configure the SAP J2EE engine; specify the amount of RAM to be used by the engine (768MB is the *minimum* that should be configured, per SAP's recommendations).
14. If you are installing the MarketSet Adapter, you also need to adapt the ServiceManager.properties file; set the parameter "AdditionalLoadTimeout=80".
15. For Windows-based systems only, add the SAP J2EE entries to the Start Menu for *all* users (as opposed to only being displayed by the installation user).
16. After all changes are made to the SAP J2EE engine, reboot the server (or simply stop and restart SAP). Note that unless you made password changes to the Web AS after it was installed, the default user ID for the J2EE engine is administrator, with no password (blank).

SAPinst can now be executed. Afterwards, SAP XI connectivity to non-SAP systems (and any SAP solutions not running on the Web Application Server platform) must be established. Then, a whole host of post-installation tasks must be performed, most of which are more time-consuming than difficult, and can be performed by user DDIC. Note that the password for DDIC remains unchanged; it's still 19920706 after all these years.

Enterprise Information Portal 5.0

Called simply Enterprise Portal for short, the latest release of Enterprise Information Portal (EIP) combines the benefits of older similar products such as mySAP Workplace 2.11, EIP 4.5, and Enterprise Unification Portal 4.5 into one feature-rich enterprise portal. Benefits like the following impact installation:

- Web browser access to applications, content, and services simplifies the installation from a client-side perspective.
- However, simply preparing for EP 5.0 means installing Microsoft's IIS and a JDK on a dedicated Web Server (other Web Server options will be available in the future).

- A Java Server (SAP J2EE Engine and JRun Server) must be installed and configured; see SAP Note 526760 and work with SAP and your other technology partners to determine the best configuration for your environment. Note that the SAP J2EE engine was formerly named *In-Q-My*. The Java Virtual Machine's default memory allocation is only 64MB; this needs to be changed to half of the physical RAM of the server upon which Enterprise Portal is being installed (and perhaps more if you are directed by your hardware or SAP technology partner to do so).
- Extended portal platform functionality includes support for LDAP-based systems management; LDAP support must be installed, however, on a corporate Directory Server (which may be installed on the portal server machine or another server).
- Unification technology and predefined Unifiers for database and legacy systems, SAP, and other applications require installation of Unification components.
- Given EP's robust knowledge management capabilities, it's likely that you will also need to interface your EP implementation with other repositories of unstructured data including Windows file servers, Web servers, Lotus Notes, Microsoft Exchange, Documentum, and so on, on top of configuring connections to R/3 and other systems.
- A number of databases are supported; if SQL Server is selected, ensure that the authentication method is set to *Mixed Authentication*. If Oracle is selected, you will need to obtain the SAP document entitled "How to Install Oracle for Enterprise Portal" from the SAP Service Marketplace,
<http://service.sap.com/epinstall>. Regardless, a supported database must be completely installed before the Enterprise Portal itself is installed.

Given Enterprise Portal's roots outside of SAP AG (EP originated from TopTier Software, which was acquired by SAP in May 2001), it's not surprising that the installation documentation for version 5.0 looks nothing like an InstGuide. The installation document is very good, however, and is structured similarly to an InstGuide in that it provides both an introduction and a pre-installation section prior to jumping into the actual portal installation process. And a pointer to
<http://service.sap.com/epinstall> within the installation guide provides access to additional relevant SAP Notes, how-to guides, and version-specific installation documentation. In fact, before you get started trying to install EP, SAP recommends that you read the following documents:

- Architecture of the Enterprise Portal, which contains information on the various portal components and their roles
- SAP Portal Technical Infrastructure for EP 5.0

- Sizing SAP Portals - Enterprise Portal 5.0, useful in determining hardware requirements
- Platform Availability Matrix, which documents the products, brands, and versions/editions of software used in an EP installation

Prior to installing EP5.0 SP3, read SAP Note 510074 (newer versions of EP require that you read through different notes). The actual Enterprise Portal installation is executed through a setup wizard available through the Enterprise Portal Welcome screen (which is in turn started by executing start.hta from the root of your EP 5.0 SP3 CD). From the Portal Installation column, click the *EP 5.0 SP3* option, and the setup/installation wizard will be launched. Do not try to install any other components or options until the portal installation is completed. And if you plan on clustering Enterprise Portal, read through SAP Note 596221 (which actually covers EP5.0 SP4) first.

From the File Download screen, select the option entitled Run This Program from Its Current Location, click OK, and then click Yes. The installation process is refreshingly straightforward. After the installation, a hotfix must be installed, the SAP J2EE engine must be adapted, a proxy server must be configured (requiring PROXYCFG.exe from Microsoft's Web site), the license key must be obtained and applied (refer to <http://help.sap.com/portals>), and the portal must then be configured for specific users, roles, content, and so on. You can also set up Secure Sockets Layer (SSL), to encrypt traffic between your Enterprise Portal and your Web clients. And finally, you can configure the portal to allow SAP transactions to be launched from portal iViews or the toolbar. All in all, EP is one of the smoothest mySAP products to install today.

By the end of October 2003, general availability for Enterprise Portal 6.0 is expected, however. This will broaden the number of solution component alternatives, and therefore impact the portal installation process significantly. For example, Apache Web Server 1.3x will be supported alongside IIS. And with regard to LDAP directories, both Novell eDirectory 8.6.1 and the Sun ONE Directory Server version 5.0 will be supported. Sun Solaris 8.0 and HP-UX 11i will join the list of supported operating systems (with AIX joining the party shortly thereafter), running 64-bit database versions of Oracle 9.2. AIX will also (not surprisingly) eventually support IBM's DB2/UDB database. Finally, the Netscape UNIX browser (version 6.2) will be made available alongside the already supported Internet Explorer and Netscape offerings on Windows-based clients. With Enterprise Portal 6.0 will come expanded support for Windows 2000 and .NET, too, including .NET support for both SQL Server 2000 and Oracle databases, plus the latest version of IIS bundled with Windows 2003 Server (at one time referred to as Windows .NET Server).

Installation Details for ITS 6.20

ITS is SAP's original and still quite popular method of making older (or not yet Web AS-enabled) SAP solutions available over the Web, accessible through the use of a standard Web browser. In a nutshell, Internet Transaction Server maps screen elements in SAP transactions to HTML, and dynamically generates HTML versions of SAPGUI screens. As of version 6.20, ITS is installed via *SAPinst*, the System Landscape Implementation Manager discussed previously. Thus, it is quite different to install than older releases of ITS. As a reminder, don't forget to load a Java Runtime Environment on each ITS server to be installed; a Java Development Kit is not included with your SAP distribution CDs, however.

Before performing an ITS installation, see SAP Note 526835 491781 for the latest updates to the ITS 6.20 installation process. An overview of the installation process follows:

- Confirm that ITS installation requirements match your ITS sizing. Preparations include installing the OS, setting up the file system, creating the proper ITS administrator accounts, verifying the correct version of Apache HTTP Server (if performing the installation on Linux or Solaris, which was finally introduced with version 6.20) or other Web Servers, and so on.
- If multiple servers are to be configured (for dual-host systems), ensure that you have good network connectivity—test for this by issuing the ping command from each host.
- Address ITS security. For example, productive ITS implementations should *always* include separate WGate and AGate servers separated by a firewall. Further, ITS files and passwords need to be protected from unauthorized user access; this is accomplished by following specific dialogs in the installation process. Finally, you need to determine whether global or specific service-related user accounts are appropriate—the latter approach is recommended for production systems.
- Determine the type of installation to perform. Four options are common, including the installation of a single host (SH), dual-host AGate instance (AI), dual-host WGate instance (WI), and an ITS Administration Instance (ADM).
- Determine the type of Internet Application Components (IACs) needed. A tool called the Package Manager allows you to apply incremental IACs to an existing ITS instance. You will need to enter the names of the specific IACs. The system package is loaded by default.
- Install the AGate first. Information you need to know beforehand includes the URL of your home page (which is where end users are “left” after they exit ITS), SAP system name, client number, SAP User ID, and password (for the user ID in the specific client to be used). Then the SAP message server name, its port

number, and login group information are required. An SAP system number and SAProuter string may also be required. For Windows 2000 systems, the name and password of user ITSAdm and the names of groups ITSAdministrators and ITSUsers must also be supplied. And finally, you will be asked to define the memory configuration of the system, where Configuration for Productive Environments requires a system configured with 512MB of RAM and the Minimize Memory Usage option only requires 128MB of RAM.

- Install the WGate. This requires the name or IP address of the AGate host, and the number of ports on the AGate (obtained by viewing the services file on the AGate)—select a number greater than 1 if you expect more than 400 concurrent ITS users, or if more than 2GB of physical RAM is installed in the AGate. When the installation is complete, the system should be rebooted.

After the core ITS installation, post-install tasks like loading IAC packages are performed. IACs beyond the default system package include the WebGUI (needed if you are installing the SAP GUI for HTML), Flowdbg (for debugging ITS applications on development systems), its_adm (installed by default in ADM installations), and Watchdog, an optional IAC specific to Windows 2000 and IIS. Watchdog allows you to monitor all of your ITS instances running on the local host via a dual DCOM interface. It also lets you register these ITS instances in LDAP directory-accessible tools such as the SAP MMC; support for the Lightweight Directory Access protocol has been available since ITS version 4.6C. Most importantly from an HA perspective, though, Watchdog supports the Windows Load Balancing Service (WLBS). With WLBS, high availability for the WGate component of an ITS implementation can be achieved. Combined with support for multiple Agates (achieved first in ITS 4.6D), an end-to-end high availability solution for ITS is made possible.

Installation Details for mySAP Components

I simply do not have the space to completely cover the installation and configuration of each mySAP component and enabling products. However, I thought it would be beneficial to address what I believe are either the core challenges or problem areas related to installing key mySAP.com components. In some cases, simply knowing up front what is needed to perform an installation—the prerequisites or planning details—represents the biggest stumbling block to a successful installation. In others, understanding what is happening behind the scenes of an installation, or a heads-up into the quirkiness of an install process taken for granted by veteran installers, will get you through the process.

In the next few pages, I cover the following components:

- R/3 Enterprise
- mySAP BI

- mySAP CRM
- mySAP KW
- mySAP PLM
- mySAP SCM
- mySAP SEM
- mySAP SRM

Because the core products within each of these mySAP solutions reside “on top of” SAP’s Web Application Server, you may want to refer to the Web AS installation discussions earlier in this chapter prior to looking at a specific component. And before you begin, note that each product ships with Web AS; fortunately, there is no need to hunt down the specific Web AS version supported by the component you are installing.

Installing R/3 Enterprise

After all of the planning and preparation tasks outlined in the R/3 Enterprise InstGuide have been performed, read through the following SAP Notes, available at <http://service.sap.com/notes>:

- 534334—Composite SAP Note for installing SAP R/3 Enterprise
- 45619—Installing R/3 with several languages or typefaces

You also need to search the SAP Notes Website for notes related to R/3 Enterprise running on your specific database and operating system platform. For example, SAP Note 529151 covers installing R/3 Enterprise in Windows environments. Similarly, SAP Note 529150 addresses known installation challenges with R/3 Enterprise and SQL Server.

After you have finished your required reading, you’re ready to begin the installation (which, not surprisingly, looks a lot like Web AS 6.20). Special considerations or conditions include the following:

- If you want to use LDAP (for the Microsoft Management Console’s SAP plug-in, for example), you need to specify the type of LDAP integration to be set up during the R/3 Enterprise installation. Select from either the Windows 2000 Active Directory Service or the Generic LDAP (applicable to UNIX and Windows machines) choices. Otherwise, select No LDAP.
- For MCOD installations, you need to specify the password for the SAPMssXPUser user. As with all passwords set during an installation, I recommend going with something simple and standard, so that no one has to think

twice about what to enter. This practice alone has probably saved my colleagues and myself countless hours of time and productivity.

- MCOD installations have only recently been supported in a cluster. Read through SAP Note 421112 for details, keeping in mind that clustering may only be performed after the last MCOD system has been installed in the MCOD landscape.
- Unless you are performing a homogenous SAP system copy, you will want to specify the Load Packages from EXPORT CD option, rather than Load Packages from MIGRATION CD.
- For the number of parallel jobs, enter the number of CPUs in your host system. For newer Intel processors supporting hyperthreading (where one processor appears as two to the OS), I suggest sticking with the number of *physical* CPUs. I had an installation hang after I entered “8” for a 4-CPU system; after rebooting and restarting the installation, I changed this number to 4 and everything worked as expected.
- The installation will take quite a few hours, and according to the installation status bar, will appear to hang at 65% and again at 75% or so (the numbers vary depending upon the specific release of Web AS). Rest assured that the system is being installed. To verify this, however, check your CPU and database activity levels with an operating system tool like Task Manager or PerfMon (Windows), or top, iostat, vmstat, w, or a similar utility (UNIX).

It's common to install a number of application servers in an R/3 environment—this is called “installing a dialog instance.” The installation process is similar (though much shorter) to installing the DB/CI; preparations need to be made, a SQL Server *database client component* (rather than a full-fledged database) needs to be loaded on the application server, and then SAPinst needs to be run. When asked for the dialog instance number to enter, I use the *same* number as the Central Instance for dialog installs performed on a dedicated application server (a server physically separate and different from the CI). Conversely, I specify a *different* number when the application server instance is loaded on a machine with multiple instances. In this way, in the future it will always be easy to differentiate one instance from another, as either the host name or the instance number will be different.

Primary mySAP BI Installation Considerations

The primary Business Intelligence offering from SAP continues to be its very successful Business Information Warehouse (BW) product. Installation preparation and planning is very similar to that of R/3 Enterprise or Web Application Server 6.20, which is not surprising considering that SAP BW 3.1 is based on Web AS 6.20. I

could only find one major difference—if installing BW on SQL Server 2000, you can optionally install a “bridge” between SAP MOLAP—Multidimensional OLAP—and Microsoft’s MS Analysis Services. This also requires installing an SAP Gateway instance along with the OLEDB/ODBC data sources corresponding to the SAP BW 3.1 database, all of which is performed *after* the core BW installation has been completed.

Before you begin installation, obtain the updated relevant SAP Note. For example, for BW 3.1 on SQL Server 2000, read SAP Note 552911, entitled “SAP BW 3.1 Content Server Installation on Windows.”

NOTE

The official minimum requirements for 3.1 are 512MB of physical RAM (another 50% on top of your sized RAM requirement is needed to support Unicode systems), three times the physical RAM plus 1GB for pagefile sizing, and Service Pack 2 for the Windows 2000 OS. SQL Server 2000 Enterprise Edition is required for an SAP installation, and in this case Service Pack 2 is required as well.

System installation will be significantly faster than an R/3 installation performed on similar hardware, as SAP BW’s default database start size is significantly smaller. After the system is installed, nearly all the same Basis transaction codes available in R/3 will be available for BW. Finally, remember that only two clients are installed by default—client 000 and client 066. Like other components, a client copy must be executed as part of the post-installation procedures.

Primary mySAP CRM Installation Considerations

CRM version 3.1 is a complicated product, not only in terms of the specific components and servers that may be required to support your CRM vision, but the connectivity to additional mySAP and other enterprise solutions. Because of this, I recommend obtaining the latest installation and configuration guides available at <http://service.sap.com/crm-inst>. And to help you understand at minimum what must be installed for a mySAP CRM 3.1 system, refer to the following list and install the products and component in this order:

1. *CRM 3.1* component.
2. *SAP BW 3.1* component.
3. *WP-PI 5.0 plug-ins* on all back-end systems (like CRM and BW), which will enable communication with Enterprise Portal, installed next.
4. *Enterprise Portal 5.0 SP4 patch level 2*, including the Knowledge Management platform.

5. *Business Package for CRM 3.1*, which is available at <http://www.iviewstudio.com/>. An installation guide for this particular product is available from the `crm-inst` site on the SAP Service Marketplace.
6. *Business Package for Campaign Management*.
7. *Business Package for Portal User*.
8. *Business Package for Communication*.
9. *CRM Designer 3.1 SP2*.

The aforementioned list may also be found in the CRM Master Guide, which should be checked for more recent information prior to performing an installation.

Additionally, critical SAP Notes that need to be read beforehand include 525457 (SAP CRM 3.1: Information on Installation), 543841 (CRM: Support Packages Collective Notes), and 515499 (Release Restrictions for R/3 Plug-In 2002). In addition, look through 497413, 497414, or 497415 for installation hints and troubleshooting advice specific to CRM on Windows, UNIX, or the IBM eServer iSeries, respectively.

Installing the SAP Knowledge Warehouse 6.0

SAP Knowledge Warehouse 6.0 consists of a number of components, central of which is SAP Web AS 6.20 (including the first 16 Support Packages). The SAP Knowledge Provider (KPro) and the SAP Content Server/SAP DB Server (which is where the contents of the SAP Knowledge Warehouse are stored) are the core components. A TREX server enables full-text searching against the SAP DB, and a Performance Assessment Workbench (PAW) Communication server provides a way to push feedback from SAP KW learning units back into the Knowledge Warehouse; both of these are necessary components as well.

To facilitate communication back to R/3 systems that do not reside on Web AS, a SAP ITS server is required. And finally, to actually use the Knowledge Warehouse, front-end clients require IE 5.0 or newer, Microsoft Office 2000 or newer, the Knowledge Workbench (a PC authoring tool integrated with Microsoft Office), the SAP KW Viewer (for display only), the SAP KW Translator (for converting HTML and multi-media objects), and Arbortext Epic Editor 4.2.1, which allows the KW user to create and edit XML documents.

Installing mySAP PLM

SAP's role-based mySAP Enterprise Portal is the delivery vehicle for mySAP PLM, specifically *mySAP Product Lifecycle Management Using cProject Suite 2.0*, which consists of a number of functional areas like

- Lifecycle data management
- Lifecycle collaboration
- Program and project management
- Quality management
- Asset lifecycle management
- EH&S: Environment, health, and safety

To enable this functionality, the SAP Solution Manager and the cProject Suite version 2.0 are necessary. Obtain the SAP Solution Manager via the SAP Service Marketplace (it is not included with your mySAP PLM media or download kit). The cProject Suite 2.0 ships with PLM, and consists of two applications, cFolders 2.0 and cProjects 2.0. The first enables Internet-based collaboration within PLM, and the latter supports end-to-end project management (from conception to planning, implementing quality processes/checks, and finally production or completion of a project). Before starting your installation, be sure to read through the most current technical documents regarding PLM implementation, available at <http://service.sap.com/plm-inst>. And then download SAP Notes 536926 (cProject Suite 2.00 Installation) and if necessary 539024 (Migration from cFolders 1.1 to cProjects 2.00). And of course, you'll want to review the latest notes and installation tips for installing Web Application Server; SAP Notes 407565 or 492208 are helpful for Windows-based and UNIX-based Web AS installations, respectively.

SAP labels collaboration *projects* (projects enabled with cProjects) as *CPR* and cFolder-enabled collaboration as *CFO*. With these labels in mind, it is quite easy to work your way through the Master Guide for PLM and determine minimum versions, mandatory components, and optional components of a mySAP PLM solution that fits your needs, as designed by your SAP Solution Architects and SAP technology/sizing partners. This is critical, because mySAP PLM normally interfaces with a number of other mySAP components like BW and R/3; successful integration requires the correct versions not only of SAP R/3 and BW, but also of the R/3 plug-ins (PI 2002.2 or higher), ITS 6.10 (if non-Web AS-enabled R/3 systems are part of your solution), and so on. Remember, the R/3 plug-ins need to be installed on the SAP Web AS server as well as any back-end systems with which PLM will communicate. And Web Application Server 6.20 requires Support Packages 1 through 8 prior to installing the cProject Suite.

Other solution components may come into play for your particular PLM solution, too. SAP Content Server and a product called ECL Viewer 4.0 (to view documents stored in cFolders) are sometimes deployed, for example. More details on these and other optional components can be found at <http://service.sap.com/PLM>.

As for a high-level technical design, SAP AG generally recommends that mySAP PLM be installed in the following manner:

- cFolders should be installed outside your intranet, to allow for collaboration with your external partners without the need for allowing access into your company's secured resources.
- cProjects, on the other hand, should be installed within the intranet, and therefore protected by a firewall. It is assumed that cProjects enables intercompany project management.

Of course, other layouts are possible. But SAP AG's experience in this regard should be considered as a de facto best practice, and heeded wherever possible.

Primary mySAP SCM Installation Considerations

The three primary components of mySAP Supply Chain Management currently include SAP Advanced Planner and Optimizer 3.1, SAP Event Manager 1.1, and SAP R/3 4.6C, all of which can be enabled in terms of installation and operations support through the SAP Solution Manager. Four primary business scenarios are supported by mySAP SCM today, each consisting of many low-level business processes:

- Demand and Supply Planning
- Procurement
- Fulfillment, including Global Available-to-Promise, Transportation Management, Outbound Order Fulfillment, and more
- Supply Chain Performance Management

SAP AG recommends reading the following SAP Notes prior to beginning an SCM installation, as appropriate in your particular case:

- 523886, SCEM 1.1 Add-On Installation on Basis 6.20, which covers the installation steps for the SAP Event Manager (EM) add-on.
- 523885 Collective note for Add-On SCEMSRV, which covers all subsequent SAP Notes related to SAP EM.
- 523883, detailing the release strategy for the SAP EM Add-On.
- 515537 or 436231, which address the plug-ins for versions 2002.1 and 2001.2, respectively.
- 431502, which provides updates to the mySAP SCM Master Guide. Information related to mySAP SCM that was not available when the Master Guide was last published/updated, and all corrections to the mySAP SCM Master Guide, are reflected here.

- 429400, detailing recommendations for Planning Versions for SAP APO Demand Planning.
- 196998, which addresses SAPGUI hardware and software platform support and system requirements for the SAPGUI for SAP APO.
- 66971, which provides more information on supported front-end platforms (supported Microsoft Windows SAPGUI releases).

Unlike other mySAP “3.1” releases, APO 3.1 is founded on SAP Basis 4.6D rather than Web Application Server 6x. And embedded into APO is SAP BW 2.1C, used to build info cubes related to planning support. Other core mySAP SCM components that are normally installed include

- SAP liveCache 7.4.
- SAP APO Optimizer 3.1.
- SAP EM 1.1, which resides on Web Application Server 6.20 (though it is recommended that it be loaded on a dedicated server, and not a Web AS 6.20-enabled R/3, CRM, BW, or APO server that you might already have in your landscape).
- SAP R/3, the version of which depends on your particular business scenarios. For example, R/3 version 3.1I and higher is supported for Demand Planning and ATP, but Outbound Order Fulfillment requires version 4.6C or higher.
- SAP R/3 plug-ins (the version of which depends on the scenario[s] being implemented).
- SAPGUI for Windows 6.10 or greater (6.20 required at minimum to support Outbound Order Fulfillment, however).
- SAP ITS 6.10 or greater, for SAP APO 3.1, if you are using the WebGUI (SAPGUI for HTML) or still using SAP Workplace.
- SAP ITS 6.20 or greater for SAP EM 1.1 support (if used).
- Optional—if an external SAP BW data warehouse for reporting is desired, it must be SAP BW 3.0A.
- Optional—if a Web AS system is desired in addition to the aforementioned Web servers, it must be Web AS 6.10 or higher.
- Optional—Any currently supported version of SAP Workplace can be optionally deployed as well.

In my experience with APO, the actual APO server installation goes fairly smoothly. However, I have had problems in the past with older versions of liveCache. Be sure to download the product-specific installation PDF for liveCache from the SAP Service

Marketplace. While you're at it, obtain the same installation document for the Optimizer server, too. I also suggest reading through SAP Notes 431498 (liveCache Installation note for APO 3.1) and 431497 (Optimizer Installation note for APO 3.1) prior to installing either product.

SAP liveCache leverages a different installation tool and process than SAPinst or R3Setup, called LCSETUP. To begin a liveCache installation, you actually must first install the tool on the server to become the liveCache server (not your APO or other server). The following process assumes a Windows 2000 platform:

1. Execute `lcsetup.bat` from the correct OS-specific folder on the liveCache installation CD. For example, for Windows environments, execute `\NT\COMMON\lcsetup.bat`.
2. After answering a question related to the name of your SAP APO server (the SID), and providing the directory path where LCSETUP will copy its files, you are then asked to log off. Best practices dictate rebooting this liveCache server as well, to ensure a fresh environment in preparation for the actual liveCache installation.
3. After the liveCache server is backed up, log in using the same installation ID used to run the `lcsetup.bat` file, and choose Start, Programs, SAP liveCache Setup, Install liveCache Instance.
4. Enter the name of the liveCache server (for example, LCA).
5. Select the drive letter that will contain your liveCache executables. I recommend keeping these off the C: drive.
6. Enter the domain of the liveCache administration user.
7. At the point in the installation where memory can be configured for use by liveCache, I recommend using it all. Note that the number that appears by default during the installation process is the maximum amount of RAM that the system recognizes—this is an excellent way of ensuring that you can actually access the RAM you have installed in your server.
8. Later, you will be asked how many devspaces should be used by liveCache. At least one is required; the total devspace allocated should be two times the size of physical RAM installed in the server, and no less than 3GB regardless.
9. Finally, after some additional devspace-related and other interview questions are answered, you will be asked to confirm starting the actual liveCache installation. After the installation process has completed, be sure to restart the server.

SAP liveCache pagefile sizing differs from typical mySAP solutions. Given liveCache's memory-hungry role, it's not surprising that some versions of the liveCache installation guides suggest the total size of all pagefiles should equal at least five times the

size of physical RAM. However, SAP Note 337445 is much clearer with regard to Windows-based liveCache systems—the pagefile is automatically set to the size of physical memory available, rather than a multiple. Further, you should refrain from setting the pagefile too large, as known issues with starting liveCache exist in this case. So the 10GB max pagefile size rule-of-thumb is not applicable in this case, either, regardless of how much physical RAM is installed.

Windows memory management can further complicate your liveCache installation. For example, access to RAM above 4GB is explained in the installation document for liveCache 7.4. Note that enabling this access must be addressed *prior* to installing liveCache; otherwise, the memory not seen by SAP during installation can not be used.

Beyond pagefile sizing, because liveCache and Optimizer servers represent glaring single points of failure in any SCM solution, it's becoming more common to cluster these products. This might hamper a speedy installation, however—although it is actually pretty easy to cluster these products, clustering still adds to the complexity of your SCM solution, and therefore the time required to install and later support it.

After the liveCache server is installed, a number of post-installation activities need to be performed. The liveCache instance must be initialized by logging in to your SAP APO server and running transaction LC10. Then, report /SAPAPO/OM_LCCHECK must be run via transaction SE38 to check both the liveCache instance and the SAP APO COM routines. Optionally, you can install the administration tools for SAPDB and SAP liveCache on Windows-based liveCache servers. Finally, to periodically reorg the liveCache COM objects, set up job /SAPAPO/OM_REORG_DAILY to run daily.

Installing mySAP SEM

Installation of the SEM-Java 3.1B component is predicated on SAP Web Application Server 6.20 technology, and valuable primarily for its Management Cockpit function. Using the Cockpit, both real-time and reporting mySAP data can be formatted and viewed, indicating the health or position of your company. Strategic Enterprise Management accomplishes this lofty goal by combining historical SAP BW reporting data with data gleaned from various functional areas within your core R/3 environment, including financials, logistics, and human resources modules.

To install SEM-Java 3.1B, perform the following high-level installation tasks:

1. Install SAP Web Application Server 6.20 or higher (if it is not already installed).
2. Install the SAP J2EE Engine.
3. Install the Software Delivery Manager (SDM) on the central instance of the Web AS server.
4. Install SAP Internet Graphics Server 6.10 or higher.

5. Install SAP BW Version 3.0B or higher.
6. On top of BW, install the SAP BW Financial Basis 1.0 add-on.
7. Install the SAP BW SEM-BW 3.1B add-on.

SAP Note 511461 details the installation of SEM 3.1B, and Note 320641 addresses configuring the Internet Graphics server.

To actually install SEM, the Software Delivery Manager is used. SEM is delivered in the form of a Software Delivery Archive (SDA) file, which is ZIP-compatible. The SDA for the SEM Management Cockpit 3.1B is loaded by the SDM into its Repository and then installed on the SAP J2EE Engine. This is done by copying the `mc.ear` file from the SEM installation CD to the SDA's inbox, and then using the SDA to "deploy" the SEM Management Cockpit product.

Primary mySAP SRM Installation Considerations

mySAP Supplier Relationship Management (mySAP SRM) brings together two solutions, SRM E-Procurement (extended existing components of the former mySAP Procurement and Business-to-Business products) as well as new components and services delivered through the SRM Supplier Collaboration Engine (SRM SCE). As you would expect, the E-Procurement solution includes Enterprise Buyer, Requisite BugsEye, and eMerge. But it also includes SAP BW, the SAP Integrated Catalog, SAP User Management Engine, SAP MarketSet Adapter, and SAP Enterprise Portal. The SAP Collaboration Engine (SCE), on the other hand, consists of the SAP Bidding Engine and SAP Supplier Self Services. Note that SRM SCE needs to be licensed for using the new mySAP SRM functionality (functionality beyond what is available by default via SRM E-Procurement).

All of these products can be managed by SAP Solution Manager, which is installed as an add-on to SAP Web AS 6.10 or higher, an SAP R/3 4.6C instance, or mySAP Workplace 2.11. For large SRM system landscapes, it's recommended to install SSM on a standalone server, however.

Two key SAP Notes should be reviewed prior to attempting an SRM installation, including:

- 492536, the SAP Note that provides updates to the SRM Master Guide
- 503196, a composite Note for Enterprise Buyer 3.5 installation tips and issues

Additionally, two Integration Components are associated with mySAP SRM, the SAP Exchange Infrastructure and SAP Content Integrator. Depending upon the business scenarios you are implementing, the SAP Exchange Infrastructure may be mandatory, optional, or not even required. The SAP Content Integrator is an optional component in all cases, though it only applies to a limited number of business scenarios.

Addressing General mySAP Post-Installation Tasks

In many ways, completing a mySAP component installation is only the beginning of the installation process; to actually prepare a system to be configured by programmers for use by end users, a wealth of additional post-installation tasks must be addressed. These core tasks include

- Setting up a client and transport strategy
- Addressing security and authorizations
- Setting up printing and faxing
- Implementing a backup/restore solution
- Addressing archiving

The last two points in the preceding list occur outside SAP, whereas the first three are directly related to tasks performed by explicitly logging into the SAP system itself. I take a look at each of these broad areas in the next few sections, and address them in detail through documents found on the Planning CD. For a more complete list of tasks and activities that must occur after a mySAP component installation, refer instead to the SIPP master project plan, or a published SAP InstGuide.

SAP Client and Transport Strategy

Don't confuse a SAP "client strategy" with the GUI strategy employed for accessing your mySAP solutions. Although the latter is certainly important, attention to the legal entities created within an SAP instance is critical. Your SAP client strategy directly impacts change management, and facilitates cost-effective training as well. For example, you might employ a client strategy where client 010 is the golden or "destined for production" client, and client 800 is a copy of 010 used for training end users or developers or your SAP technical team. And your change management process might be to promote changes from client 010 to client 800 on a regular basis, say weekly, while refreshing clients *between* instances on another regular basis—a transport strategy. Refer to the "Sample Client Strategy" PowerPoint document on the Planning CD for more detail, including a sample six-system landscape client strategy layout.

With regard to your transport strategy, you will probably initiate all development activity in your DEV environment; alternatively, some customers choose to initiate this activity from a special client residing on a Business Sandbox. Regardless of origination, all changes to any objects will also be initiated here. As development progresses, objects will eventually need to be tested; this is accomplished by transporting the updated object(s) to your Test/QA environment. Beyond pure development efforts, changes can be initiated and submitted by developers, ABAP/Java

programmers, Basis/Technical Implementation team members, and other technical and functional SAP TSO resources. These changes are typically managed by a dedicated Change Management Organization, discussed in detail in Chapter 13.

Besides ABAP and Java code, transports can include enhancement packages, support packages, SAP kernel upgrades, other patches, and so on—anything within the realm of your SAP system. See Chapter 17 for more details, and reference the Planning CD for a “How To” document relevant to performing transports. But keep in mind that SAP transports represent only a fraction of the content of your regular monthly or quarterly “Change Wave,” because changes to your mySAP solution can include hardware and firmware updates, OS updates, database patches, and so much more.

At the end of the day, your transport strategy facilitates sound integration and Quality Assurance testing. Often this is accomplished by moving bulk changes in waves, as I just mentioned. In other cases, however, changes might need to be transported quickly (as in the case of an emergency, where poor code is impacting profitability or customer satisfaction). The key to successfully allowing only authorized members of your SAP TSO to circumvent change management brings us to our next topic, security.

Security, Authorizations, and Trust Relationship Management

Security in general, even just the concept of SAP Authorizations, consumes entire books in and of itself. SAP dedicates a great deal of training classes and consulting time to the topic of security, too. Why? Because SAP security is much more than managing risks to information assets, or protecting business processes through the use of security solutions, such as firewalls and the like. Underneath the wide-ranging topic of SAP security lies equally broad *Secure Systems Management*, which further includes areas like Single Sign On (SSO), Secure Network Communication (SNC), secure ABAP Programming, and more. Add to this *Identity Management* and *Trust Relationship Management*, and you will begin to understand the magnitude of SAP security.

In the world of mySAP solutions, the ability to manage users’ roles and authorizations outside the boundaries of a single SAP component is critical. SAP calls this Identity Management or central user administration, and it is made possible through a *directory service* that enables central security management of both mySAP and non-mySAP systems. But that’s only the beginning; nestled underneath the broad area of Identity Management lie three matters of importance in their own right:

- Users/Roles/Authorizations
- Centralized Administration
- Directory Service Integration

Authorization management consumes a great deal of time before Go-Live. Getting authorizations worked out and roles nailed down is time-consuming in the best of cases—simply locking down users so that they cannot use transaction SE38, for example, to pull practically any data from the database, is only the beginning. You need to determine who should have read-only access to particular functional areas and components, and who can create new objects or edit existing ones.

Authorizations apply to different roles (or classes of users), too, not just end users. Authorizations apply to systems administrators, for example—you don't want just anyone to have the ability to add or modify the supported languages in your system, nor can you afford to allow just anyone to use the Transport and Management system; only a select few members of the SAP TSO merit these abilities. Similarly, authorizations apply to developers—you need to lock down systems (like production!) such that developers are not even tempted to make code corrections outside of development systems. Others, like Help Desk personnel, senior computer operators tasked with managing a mySAP enterprise, and so on also have job-specific roles and authorizations that need to be adhered to. I recommend SAP Labs' "Authorizations Made Easy" book to help you get a foundation in this area, followed by targeted technical consulting to get your own implementation started on the right foot.

In the past, I have found it useful to create a Profile Matrix or Role Matrix to help me manage users, roles, and so on, too. This can be taken a step further, and adapted for specific components, cross-application roles, and authorizations as well. I suggest considering such an approach to document not only the roles or authorizations inherent to your organizations, but also who has the authority to assign a role/authorization, and who has authority to make changes to them.

Authorization management alone will not ensure a secure, auditable environment, however. Trust Relationship Management takes managing authorizations to the next level by attaching trust relationship information to the data itself, regardless of how it might be distributed or shared through collaboration or other means. By attaching this trust relationship metadata to data, records can be maintained that prove that the data has been used correctly. I believe that this will only grow in importance as distributed environments grow to include more partners and vendors, allowing companies to collaborate and share their data and business processes. To learn more about Trust Relationship Management, including Authentication, Pluggable Authentication Service (PAS), Logon Tickets, and X.509 Certificates, see SAP's Service Marketplace.

Printing and Faxing

SAP uses spool work processes, which execute on one or more application servers (or the central instance) to process print requests. In this way, SAP supports printing in quite a few different ways:

- Print requests can be routed from an application server to its local OS print spooler, and printed from an application-server-attached printer.
- Using TCP port number 515, print requests can be routed to a standard “line printer” service from the target host.
- Using a printer daemon (SAPIpd) and port 515, output can be printed.
- Printing via the dialog work process connection is maintained by a SAPGUI or WebGUI front end; output is routed to printers (usually) defined at an OS level and set up in SAP.

The last printing method discussed—using the SAPGUI or WebGUI—is the most popular approach for users who do not need to print giant reports best handled by industrial-strength data center printers. The first step in setting up this type of printing is simple—verify that you can access the printer from your desktop or laptop at an *operating system* level. Or make sure that you can print from a different application, such as Microsoft Word. This will save you troubleshooting time if you run into problems. Next, pull up the online documentation that ships with the SAP product you are configuring for printing. For our purposes here, let’s assume we want to set up a printer for Web AS. Drill down into mySAP Technology Components—SAP Web Application Server—Computing Center Management System—SAP Printing Guide. And then follow the procedure outlined here. Through this procedure, you identify the printer (by UNC or IP naming conventions), indicate whether you want to print immediately upon receiving a print request, provide a cover sheet for each print job, and define the output in terms of lines, columns, and general format—all of which is clearly illustrated in Figure 11.4. Later, you can use transaction SP01 to manage your print requests.

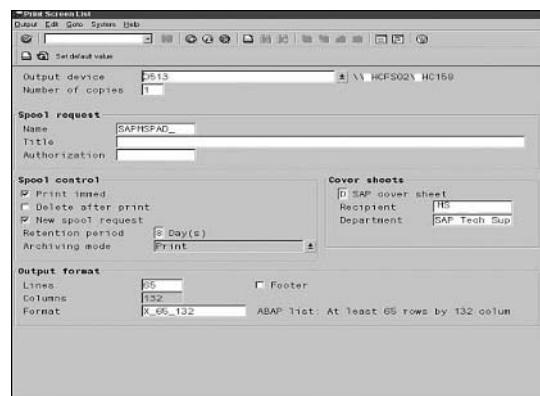


FIGURE 11.4 Setting up a printer within SAP allows for customizing exactly when and how you receive your print request.

In some cases, it's also necessary to set up faxing capabilities. In the scope of my travels, I have come across many different faxing applications. But Facsys and Faxination are the only ones that left an impression on me, and after talking with my customers who still use these products, they seem worth mentioning.

Faxing is set up within the SAPGUI; afterwards, go to Tools, Business Communication, Communication, SAPConnect, Goto, Error handling to check the error handling status of a fax, or use SAPOffice for fax status (depending on the faxing application). SAP notifies the sending user's Inbox or Outbox of fax attempts and related status, depending on whether the fax was sent/received successfully or unsuccessfully. Successful fax notifications are dropped into the sending user's Outbox, whereas unsuccessful fax notifications are dropped into the sending user's Inbox with a description of the kind of problem that occurred.

To troubleshoot basic faxing issues, use transactions SM59 and SM37. The first verifies that the TCP/IP connection is good, and the latter can be used to see if a fax job is running. Transaction SCOT (SAPConnect) can be used to reprocess fax jobs if necessary. Additionally, SCOT allows you to view fax information, process waiting faxes individually or by group, and look at errors and other issues.

Backup/Restore Considerations

Surprisingly, backup and restore (B/R) solutions for SAP represent one of the more troublesome areas in an implementation. Why? I'm really not sure, other than to say that it's more complicated to back up a large enterprise system than it appears.

Although most people tend to lump B/R solutions into the "tape drive/tape backup software" category, B/R solutions are more accurately characterized in the following ways:

- Backups can be integrated directly with SAP (through SAP's BR tools interface, typical of Oracle database solutions) or might require their own proprietary backup/restore application interface (such as SQL Server, because by design it does not support SAP BR).
- Backups can be offline (where the database is shut down and all files are therefore closed) or online (where the database is up and running).
- Backups can take advantage of a number of different tape drive and tape media technologies, which vary in terms of price, speed, and capacity.
- Backups can be centralized (where multiple databases residing in a SAN all get backed up to an enterprise tape library) or decentralized (where each database server has its own locally attached dedicated tape solution).
- Backups can be configured such that the data is streamed to multiple tape drives and protected with parity (much as RAID 5 arrays work to protect disk

data), dumped to multiple tapes (for speed), or simply dumped to a single tape (for simplest administration).

- Backups can be performed in other ways prior to dumping bytes to tape (which ultimately needs to take place, consistent with best practices)—for example, backups can be dumped to an array of disk drives (for speed and as a “backup” to the backup), or they can be created from a clone set of drives configured specifically for backup purposes.

One of my SAP colleagues shared the following with me recently. I believe this sums up the attitude that companies should have regarding *what* to back up:

- If you don't back it up, you cannot recover it.
- If you don't back it up *and* ship it offsite, you will not survive a disaster.
- If you don't test your backups by recovering from them, you really don't have a backup.
- If you don't maintain your backup/restore system, it will at best perform poorly for you.

In the end, each of these nuggets of wisdom tells us the same thing—back up your data, or find another job.

Archive Considerations in the Real World

From R/3 sales orders to CRM customer data and APO demand planning versions, an enormous amount of data will be created annually in your mySAP environment. It's not too early to begin thinking about how you will manage this data in terms of database growth and size while ensuring excellent database response time—remember, it's the database performance that drags down average response time over the life of your SAP system more than any other single component. I completely understand the desire to kick back a bit after Go-Live and focus on things like ironing out operational issues and getting to know your family again. In fact, I must admit that most of my customers tend to put off discussions of archiving until a year or two after Go-Live, when they are fighting yet another unforeseen disk space issue and facing potentially expensive disk subsystem upgrades or replacement strategies. But if they were able to do it all over again, believe me, many of them would gladly make room in their project plan to implement at least a basic archiving solution *before* Go-Live.

So let's step back and talk about why an archive solution is needed in the first place. If we look at typical R/3 systems, they tend to grow at a pretty regular rate when things settle down a few months after Go-Live. Most of my customers' production

R/3 databases grow between 5 and 10GB a month. The growth comes mainly from transactional data—doing business day-to-day, like cutting purchase requisitions, creating sales orders, and so on. With every extra GB of data, the R/3 database tables grow longer and longer. SELECT statements take longer to run as more rows of data are added to each table. Database “reorgs” and other administrative tasks take longer, too. Ultimately, a transaction that used to consume 500ms of database request time will consume 600ms, and then 700ms. The system will even begin to *seem* slower to its end users; their perception will be grounded in truth, unfortunately.

An archive solution works by slimming down the database to something resembling its former self, by removing old or little-used data from the database—transactions run faster because there is less data to sift through. SAP offers a host of built-in tools and interfaces to help you archive your data, and a slew of third-party archiving solutions are available as well. Thus, archiving simply means dumping your unwanted data somewhere else, somewhere other than your production database. My customers have taken a pretty liberal view of where precisely “somewhere else” is, however:

- In the most classic sense of the word, archived data is actually dumped to a *near-line storage facility*, such as a magneto-optical jukebox. These hardware devices are slow compared to disk subsystems, but provide a good tradeoff between online data and completely offline data (like data that resides on tape). And they exhibit phenomenal reliability, making them perfect candidates for sensitive or critical data that absolutely must be safeguarded for a long time.
- Other customers have dumped archived data into more readily *available and cheaper* media, such as inexpensive RAID 5 storage systems. With the cost per GB of storage space dropping every year, this can be a pretty attractive prospect over time. However, disk drives fail, and therefore must be protected from failure.
- Still others look to their *data warehousing systems* as archive “systems of record.” Indeed, one of the most common reasons I heard for implementing SAP BW the first two years after it was released was to archive old production data out of R/3 and other transactional systems.

In a nutshell, before you go to the trouble of understanding all of your requirements and needs, ask yourself the following questions:

- Do you expect your database to hit 200GB in the next two years?
- Do you expect your database to grow at more than 10GB per month?
- Will you have a hard time getting budget money or time later to implement archiving, or will it be easier on everyone to earmark the dollars and time now during the implementation?

If your answer to any of these three questions is affirmative, read on—there might be a lot of ways to implement archiving, but the keys to a successful archiving project are few:

- You need to understand your data. This means understanding more than simply what may or may not be archived. It also means understanding how different mySAP.com components and even the discrete business areas making up these components can benefit from archiving. And it means understanding how both your transactional and master data grow over time.
- Understand the technologies, tools, products, and consulting expertise available and necessary to pull off an archive project.
- Understand priorities. With multiple mySAP components and hundreds of business objects available to be potentially archived, you must understand what to tackle first. There's plenty to choose from—production orders, sales orders, material and financial documents, and so on.
- In the same manner, focus on OLTP environments like R/3 and CRM, where absolute response time is *always* more important than in OLAP or reporting environments like BW. In other words, go after archive projects that focus on customer-facing systems that ultimately provide better ROI.
- Understand your access needs—can you tolerate 3–7 seconds of wait time to access archived data, or do you need it faster? This will dictate the hardware technology underpinning your archive solution.

Certainly, other post-installation tasks outside of developing a client strategy, setting up TMS, and addressing security, printing, backup/restore, and archiving need to be considered. An SAP license needs to be installed, as should a SAProuter (for additional security), the SAP online documentation needs to be installed if this hasn't already been done, and RFC destinations need to be verified. Basic operational tasks, such as setting up logon groups, operations modes (OpModes), background house-keeping jobs, applying Support Packages, performing the initial client copy of client 000, and so on, also need to be done. In most cases, the relevant InstGuide walks you through much of this; worst case, the InstGuide points you to SAP's Service Marketplace, where you can obtain further detailed information.

Tools and Techniques

The Planning CD contains a sample SAP component installation checklist, a post-installation checklist/procedure documents, and each of the figures found in this chapter, in Microsoft PowerPoint or PCX format. Other specific documents of interest include a “Sample Client Strategy” PowerPoint (essentially an illustrated sample

client strategy layout for a fictional six-system landscape), a “Faxing Architecture” PowerPoint, and a number of documents related to the SAP Configuration Assistant. Additional “how-to” documents that could potentially prove useful at this stage of your implementation can also be found on the Planning CD, though they reside more appropriately with other resources related to Chapter 17 and preparing for Go-Live.

Summary

In Chapter 11, I covered SAP’s methodologies over the last few years, and how each successive methodology better addressed the tasks surrounding technical installation of SAP’s products. I then picked up where Chapter 10 left off in terms of completing planning and installation of the rest of the SAP Data Center’s SAP system landscape. This naturally led into a comprehensive discussion surrounding SAP’s installation documentation, tools, and approaches. I discussed how Smart Implementations and the SAP Configuration Assistant have proven useful in automatically adding new mySAP.com components into your SAP system landscape. I then began the long process of detailing the installation of mySAP’s technical foundation (recently renamed to NetWeaver), from Web Application Server to the SAP Exchange Infrastructure, Enterprise Portal, and even SAP’s old Internet-enabling workhorse, ITS. Specific mySAP.com component installation details or challenges, followed by attention to post-installation tasks and considerations, rounded out the chapter.

In concluding this chapter, we have finally accomplished the activities usually associated with the core SAP implementation, from sizing and blueprinting through installation. Part III takes us into SAP Functional Development and all of the associated tasks relevant to preparing for Go-Live. Thus, in the next few chapters, we’ll look at filling in the remaining holes in your SAP Technical Support Organization, addressing change management, implementing systems and operations management, and performing all of the testing necessary before Go-Live.

12

Rounding Out Support for SAP

Identifying and Staffing the Remaining TSO Roles

Consistent with our project management approach to writing this book, and based on the foundations laid in Chapters 4 and 8, you by now should have a well-oiled SAP Technical Support Organization in place. Key positions have been staffed for some time, your mix of consultants to internal resources tends to favor the consultants, and the assembled team represents an elite group of folks adept at supporting much of your unique SAP Solution Stack in place today. In fact, your team members are by this time experts in ensuring that the SAP solution is indeed available to be used by the various developers, business process analysts, functional consultants, and other testing/development-focused individuals, who at this point represent the TSO's primary customers.

The business-focused implementation team members continue to play as key a role as ever. Some members of the technology-focused support team have probably transitioned back to a combination of their former and current roles, though, as their tasks have been completed or future tasks lie "on hold" awaiting the completion of future milestones. Other technology-focused individuals will find themselves in a similar position soon, too, as the project plan dictates.

IN THIS CHAPTER

- Identifying and Staffing the Remaining TSO Roles
- Approaches to "Filling in the Final Holes"
- Let's Give a Big Warm Welcome to the "Specialists"!
- The Most Overlooked Critical SAP Support Function
- The Role of the SAP Help Desk
- Putting It All Together in the Real World
- Tools and Techniques

But the SAP support organization is by no means complete. In fact, based on my recommendations in Chapter 8, the team should only be about 80% staffed. Much of what is missing of the remaining 20% can probably best be described as either subject matter experts in certain layers of the SAP Solution Stack, or long-term operations, help desk, and other such support providers.

Throughout this final chapter on SAP staffing, I will discuss these few remaining positions. In addition, I will note appropriate methods of staffing or otherwise fulfilling the jobs and tasks to be accomplished by these late-comers to the SAP project. My hope is to leave you with a sense of the critical nature of these final positions with regard to the success of your SAP implementation project, including real-world organizational structures, tools, and approaches.

Where Exactly Are the Holes in Your TSO?

Before it makes sense to begin talking about *how* to staff for the remaining technical positions, a brief discussion on the typical “holes” that remain at this point in the project is in order. To start with, the biggest, most gaping, holes center on SAP operations, systems management, and help desk activities. These represent sustaining positions that will be around long after the majority of the SAP functional team disbands, and the other consultants are let go. I will cover these holes in great detail later in this chapter.

Additional staffing holes exist around various specialists, including security, performance, specific SAP components like EBP or Enterprise Portal, and so on. Specialists involved with integrating components or legacy systems within the context of the larger SAP solution, not to mention high-availability and disaster recovery experts, will assist us in filling in critical staffing holes, too. In all of these latter cases, the tasks to be accomplished are typically short in duration but require the advice of SAP Subject Matter Experts (SMEs) to keep the project plan on track.

Holes in the realm of “testing” exist at this point in the project plan, too. Functional or integration testing, along with general systems testing and full-blown SAP stress-testing, will occur soon, and therefore need to be planned for, executed, analyzed in terms of testing results/output, and so on. The tuning and tweaking that comes out of the results of testing (which, after all, is often the goal of the testing in the first place) also eventually needs to be accomplished. This testing provides feedback to the rest of the SAP support team when it comes to configuration parameters that need to be changed, SAP solution stack elements that need to be tweaked, and so on—to ultimately better serve the patiently awaiting user community.

Accomplishing this goal is not trivial, and therefore requires competent people

knowledgeable and experienced in both testing and analyzing those same test results.

Finally, preparation for Go-Live may require additional specialists and analysts simply for the sake of risk mitigation. That is, it is common and very advantageous to plan in advance for SAP Solution Stack experts to be onsite the week *before* as well as the actual week of SAP Go-Live.

Having said all of this, it is imperative to get busy identifying which specific roles are missing in your particular organization, as you continue reading ahead on the subjects of different methods and approaches for staffing the missing “twenty percent.”

How Big Is “Big Enough?” in the Real World

In the past, I have been asked “How big should my SAP technical support organization be?” Although no single answer fits everyone, a couple of thoughts come to mind. Consider the following:

- For every technology-focused group, there tends to be a business-focused group with similar interests. These two groups need to communicate effectively with each other and operate in lockstep with one another.
- The more complex the SAP Solution Stack, the more critical it becomes to have subject matter experts either on staff, or quickly available (that is, through what I call “consulting on demand” arrangements).
- From a cost/value perspective, determine whether an onsite resource is more advantageous, or an offsite “on-call” resource makes better financial sense. Either approach can make a lot of sense, depending upon the situation or environment. That is, if the weighted risk versus cost of downtime exceeds the cost of keeping an incremental resource on hand, it makes good business sense to keep the resource on hand. Otherwise, it probably makes more sense to simply arrange for quick access to consultants and other “pay-as-you-go” assistance.
- Technical support organizations tasked with supporting lots of users will tend to be larger than those tasked with supporting fewer users. The number of users ultimately drives help desk calls, shrinks windows of available downtime (to perform system maintenance, for example), impacts the growth of the SAP databases, impacts the number (and therefore complexity) of application servers and ITS/WebAS servers, and so on. Each organization is challenged to achieve the best degree of balance.

- An SAP TSO responsible for a larger geography or multiple time zones, or in some other way distributed, will tend to be larger simply by virtue of the support required (and therefore the number of persons needed from a pure staffing perspective). Refer to the organizational charts at the end of this chapter for a view into how small, medium, and global entities attack supporting SAP.

In the end, work with your vendors and partners to compare your operation to similar mySAP operations. For example, I often provide what I have coined an *SAP Comparative Analysis* to my SAP customers and prospects. The comparative analysis consists of raw data covering hundreds of my own SAP customers that can be sorted and analyzed by factors like the following:

- Number of average concurrent users
- Hardware, operating system, or database platform
- Version of SAP, that is, R/3 4.6C or BW 3.0B
- Monthly average growth in database size
- Average user/dialog, background, and other response times
- Print volumes
- Operations staffing model
- Size of SAP Basis team
- Much, much more

A sample comparative analysis is shown in Figure 12.1.

Of course, no real customer names are provided here or in my sample on the Planning CD (so as to maintain client confidentiality), but I have found over time that everyone gains value from this kind of information. And in the end, it lets a company “benchmark” its own SAP project against others, helping a company understand similarities and differences between its own implementation and other implementations across the world.

SAP Comparative Analysis, last updated 04-2003			
Configuration	HP-UX/Oracle8	Tru64/Oracle8	W2K/SQL2K
General SS Software Versions:			
SAP Component	R/3 4.0B	R/3 4.6C	R/3 4.5B
SAP Kernel	patch 50	4.6D patch 432	4.5B patch 37, LCP 45
RDBMS	Oracle 8.1.7	Oracle 8.1.x	SQL 2000
OS	HP-UX 11i	Tru64 5.1 DB and W2K Apps	W2K DB and NT4 Apps
Staffing Model:			
Basis/Technical	5	5	7
DBA and Disk Specialists	4	3	3
Other Infra Specialists	8	8	6
Programming Support	8	25	15
Security/Access/ChgMgmt	7	4	4
Enterprise Operations	6	8	6
SAP Help Desk	8	3	2
Database Server:			
Model	HP rp 7410	AlphaServer GS320	Proliant DL760
Processors	8 @ 750 MHz-2MB	24 @ 667 MHz	8 @ 900MHz
RAM (GB)	8	16	6
MAX CPU Load %	21	36	29
Disk Subsystem Details:			
Size of Prod Database (GB)	650	1100	460
Monthly DB Growth (GB)	12	32	8
Model of Disk Subsystem	VA 7410	ESA 12000	EVA
Central Instance:			
Dedicated, or on DB	On the DB Server	On the DB Server	Dedicated Proliant 8500
MAX CPU Load %	-	-	20
Application Servers:			
Total App Servers	5	10	12
Number of dedicated DIA	4	8	10
Number of dedicated BTC	1	2	2
Number of dedicated UPD	0	0	0
Typical Server Models	rp5x class	Proliant DIA and Alpha BTC	Proliant 6500/6400
Number of Processors	2-4 CPU	4	4
RAM (GB)	6 GB each	3-4 GB each	3 GB each
MAX CPU LOAD%	37	20	39
Dialog Steps:			
MAX DS/HR	58,000	79,000	68,000
USERS:			
Active Users (CSD)	780	1020	1730

FIGURE 12.1 Credible partners and vendors will maintain similar data like that found here in what I have coined an "SAP Comparative Analysis."

Approaches to "Filling in the Final Holes"

Don't let the fact that we're staffing again midway through the project fool you into thinking that these final additions to the SAP TSO are anything but critical. They are! As you can see in Figure 12.2, these positions support key solution vendors, tasks, and essentially the fundamental layers of the SAP Solution Stack.

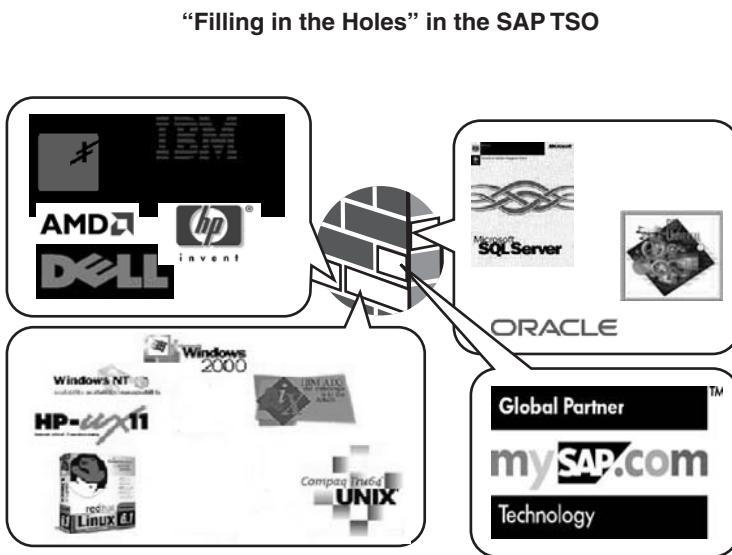


FIGURE 12.2 Like a strong brick wall with a few bricks missing, the SAP project will never be complete without eventually filling in all of the “holes” in the SAP Technical Support Organization.

The skillsets that will be brought to the table during this round of staffing will take us through the remainder of the SAP project, and position us for Go-Live and beyond. Some of the positions will exist from this point on through the life of the SAP implementation, as I said earlier. But perhaps even more importantly, some of the short-term positions actually represent critical must-have skillsets, needed to simply make it to Go-Live. Regardless, both long- and short-term posts will be subject to their own unique challenges in regard to staffing, transitioning into the SAP TSO, and so on.

Before we discuss approaches to filling these positions, let's take a quick look at the keys to successfully expanding the TSO.

Keys to Expanding the SAP TSO

In my experience, five fundamental keys exist to bringing new members into an existing SAP Technical Support Organization. In no particular order of importance, these include the following:

- People skills—the ability to build and maintain professional relationships—represent key #1. The presence or absence in SAP TSO newcomers of other “soft skills” also plays a key role in how the overall TSO will perform during the last half of the project. However, a team not ready for a newcomer, or a newcomer

not ready to work as an integral component of an existing team, will have that much more difficulty to overcome. Often, the key to a good relationship relates directly to simply understanding and working well within the established hierarchy—respecting everyone on the team, and treating everyone well, will go a long way toward cementing relationships that last. Respecting the hierarchy with regard to senior and junior folks, mentors and those being mentored, and where the newcomer fits in, is critical too.

- Attitude toward communication is key #2. No one can afford to be aloof or “go it alone” during these precious last months before Go-Live. That is, the importance of communication becomes crucial and cannot be overstated. Consider the following example. A new member of the team has been brought in as the eCommerce guru. His job is to tie together SAP’s Enterprise Buyer Pro system with a Requisite catalog capable of being securely updated by vendors and partners. He must also establish a process for maintaining the catalog. This newcomer will therefore work closely with the folks in the company’s security group to ensure that all firewall, DMZ, and other such intranet/Internet concerns are addressed. And he will work closely with the business folks and SAP Infrastructure/Basis team as well. Naturally, if our newcomer is not a team player, or not comfortable communicating his decisions and configuration requirements, everyone will fail.
- Key #3 is this—everyone needs a backup. A single person, no matter how good he is at his job, becomes a single point of failure if all knowledge of a particular piece of the solution is known only to him. Take again our SAP EBP newcomer as an example. If this guy keeps everything to himself, if he does not share the knowledge obtained through implementation, the project’s risk grows. For every position, task, or responsibility, a second person needs to be trained and able to “step up” should the need arise. This is especially critical with contract personnel—beware of providing your consulting folks with key knowledge that is not being *regularly* transferred to employees or long-term contractors. For such a no-brainer statement, I have seen this time and time again at client sites. And it works both ways, as I feel that it is the client’s responsibility to ensure that a staff member absorbs a consultant’s knowledge as much as it is a consultant’s responsibility to ensure that he proactively shares his knowledge with the client. In a perfect world, I would require every consultant to “tie off” with an employee on at least a weekly basis, with the goal being to share expertise, knowledge, and insight gained that week. This would be followed up with a weekly written report indicating the same, to be archived in the project’s Knowledge Repository. For short-term projects (that is, a few weeks or months), I would simply pair up each consultant with an employee tasked with knowing how to “do” or support everything the consultant does when the consultant is finally out of the picture.

- Building on key #3 is my next key—documentation. This is without a doubt the most important component of providing “backup” to other team members. Unfortunately, the job of enforcing the creation and maintenance of good documentation is often put on the back burner, becoming important temporarily only when a crisis occurs (like when our SAP EBP newcomer winds up in the Emergency Room after falling asleep at the wheel because he hasn’t slept more than four hours a night for the last week). Key #4 needs to be pushed from the top echelons of the project all the way down to the guys installing software and turning the screws—without documentation (processes, procedures, and practices), your SAP implementation will never achieve the highest levels of availability otherwise possible.
- A balance of first-rate experience in completing the tasks at hand, coupled with knowledge of why and how the business side of the project is affected by performing (or not performing!) these tasks represents my final key, key #5. In this area, being successful is all about achieving balance. The best newcomers to the project will understand how the business is run today, and how their efforts will bring the company up to the next level of integration or capability. Therefore, these folks need to have close to the same understanding of the business processes (that they are supporting via their position in the SAP TSO) as the current team does. And it goes without saying that they need to be up to the technical challenges they will face, too.

With these five keys in mind (reinforced in Figure 12.3), let’s drill down into the two approaches to staffing for these final SAP TSO positions—reallocating internal staff, and bringing in third parties.

Transferring in New Internal Folks

Even though it’s not a likely avenue at this point in the project plan, it’s worth mentioning that perhaps the SAP expert you are seeking is just a “few cubes” down. I say unlikely, though, because by this time you have probably pulled every SAP-literate Information Technology specialist or Business Analyst into the SAP TSO, either on a full-time or shared/part-time basis.

In larger companies, though, it’s certainly possible that the SAP EBP expert you need, for example, is hiding somewhere in an eCommerce or Internet Connectivity Support group. Perhaps he’s a newcomer to your company, or a recently appointed contractor—check with the HR organization to scour through recent new hires and other folks brought on board in the last year or so. A really good HR group (or IT manager) will also maintain something of a skill set or proficiency database of their folks, too—run a search against it, and you may be surprised at the talent cubby-holed away in some not-so-faraway place.

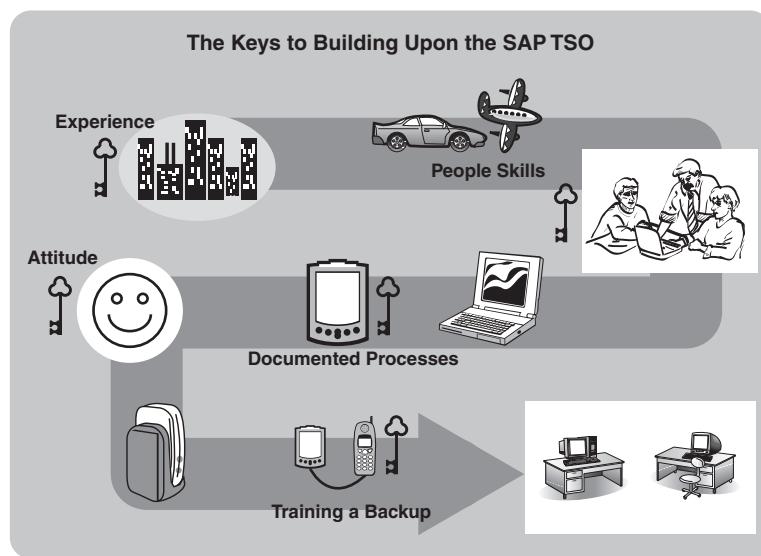


FIGURE 12.3 Here you see the five “keys” to adding new members to an existing SAP Technical Support Organization.

Finally, post the job internally on the company’s intranet or other job-posting system. Communicate open positions via weekly updates to the business and company-at-large, or the company or IT-specific newsletters that may be regularly published about the SAP project. Target your colleagues in the various IT and business support groups, too, searching for anyone who might fit the bulk of the job description.

After all of the prospects have been identified, evaluated, and ranked, pull out any clear candidates. At this time, it might also become apparent that other prospects could rise to the occasion with a bit of formal training or minimal on-the-job training—put these candidates to the side, too. Then I recommend going through the technical phone screen and role-playing interview process described in detail in Chapter 8, keeping in mind that an internal candidate will require less investment in some respects than outside candidates. That is, as you see in Figure 12.4, your internal candidates already have the following advantages:

- They know the company. This means they have contacts and relationships, understand how things get done, have some understanding of some of the business processes and practices germane to the company, and so on.
- They are easy to communicate with—they have a company telephone number, email address, pager/cell phone, and so forth. They are therefore instantly

accessible, and presumably instantly “productive” future members of the SAP TSO. They are ready to go!

- They understand something about the impact that the SAP project will have on *everything*. Therefore, they should appreciate the fact that SAP will change the manner in which the company does business. And if they have sought out the project and you’re talking to them, it’s a given that these changes excite them!

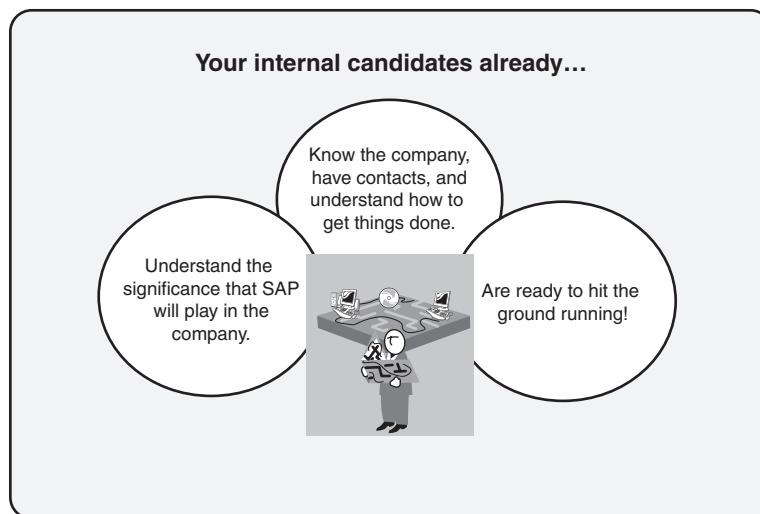


FIGURE 12.4 Internal candidates have a clear advantage over third-party consultants and contractors.

In the end, though, check this step off your Master Check-Off List, and be prepared to go external with your job search. Of course, if the need is critical, and you wait too long, you’ll find yourself in trouble. I recommend that you start the external search as soon as you finish searching for internal folks, but before you start phone screening and so on. In this way, as you begin evaluating your internal “finds,” you will simultaneously be collecting qualified leads from the outside, which brings us to the next section.

Leveraging Third-Party Consultants Again

Assuming that the idea of bringing in an internal candidate is simply not feasible, the only real alternative is to leverage your favorite consulting firms again. I say “firms” because you need to give yourself not only choices in candidates, but also the ability to leverage pricing, terms, and conditions between different consulting organizations.

But should you go to the Big Five, your enterprise hardware/software providers, a mid-size SAP consulting firm, a Mom & Pop "body shop," or the independent contractor route? That's a big question, and like any good consultant, I have to answer it with a resounding "uh, it depends." Each avenue has its own good and not-so-good points. Unlike staffing the bulk of the SAP TSO, where I tend to favor the larger consulting organizations to achieve your 80% staffing goal (to leverage economies in pricing, administration, and to minimize the number of players and so on), there's much more to discuss at this level of staffing. Consider the following:

- The Big Five and mid-size SAP consulting firms probably have the mySAP-specific expertise you're looking for, but do they have the niche expertise you need that is specific only to your unique SAP Solution Stack? Further, can you afford them? Finally, are these skills embodied in a single person, or will you need a team of "micro-specialists" to achieve your goals?
- The enterprise hardware/software providers might be less expensive, but do the individual SAP consultants on their consulting rosters understand more than just their niche areas? And do they understand the ramifications that their niche areas have on your mySAP project holistically?
- The body shops and independent contractors may be less expensive from an hourly perspective or in the short-term, but are you assured of getting "the right person"? Also, if you lose this person to another SAP shop for an incremental few dollars an hour, what is your recourse? Will the body shop be able to backfill in a timely manner with an equally qualified resource?

These are all important questions. The right answer for many companies that have implemented SAP in the recent past amounts to taking a step back, though, and approaching the problem from a broader perspective. Let's examine some of their real-world methods next.

Consulting in the Real World—Fewer Is Better

Two of my newest and very successful SAP implementations took the approach to staffing that fewer partners was better than more partners. In the first case, a combination of the customer's internal people, HP, Microsoft, and SAP accounted for 95% of the project team. In another case, the customer, HP, a third-party mid-size consulting firm, and SAP accounted for 95% of the project team. Figure 12.5 illustrates this approach.

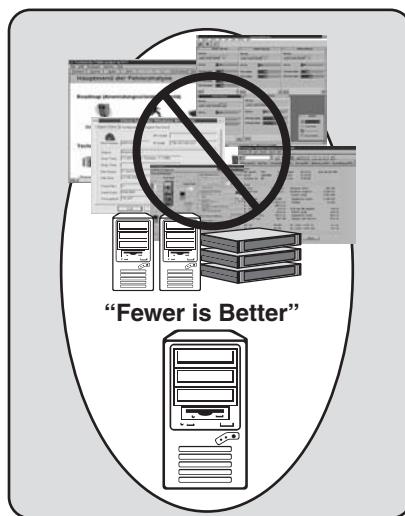


FIGURE 12.5 “Fewer is better” is one approach that companies might consider when it comes to determining the critical solution stack partners at this stage in the project plan.

What did both projects have in common? They kept the number of “players” to a minimum. In effect, they

- Reduced time related to all of the overhead associated with bringing in lots of incremental partners—relationship building, negotiating pricing and terms/conditions, working out how interviewing/evaluating candidates is accomplished, invoicing/billing processes, and so on.
- Minimized finger-pointing. With fewer players comes improved accountability, simply because the lines of accountability and responsibility tend to be drawn more clearly. Like so many other things, I look at this from a solution stack perspective—one partner or vendor is held liable for each layer in the stack.
- Simplified organizational dynamics. Just as fewer players equates to less overhead, fewer players also tends to equate to fewer communications issues, simpler escalation procedures in times of trouble, and so on.

These two companies had another thing in common, too. They each leveraged best-of-breed partnerships to comprehensively address every layer of the solution stack. So, instead of trying to make a “Mom and Pop” peg fit into an “enterprise hardware provider” hole, these two companies pieced together a virtual support organization that worked hand-in-glove. And they didn’t need to engage 20 different consulting, software, and hardware companies to be successful.

Consulting in the Real World—Bigger Than Me

In another real-world example (not related to the two companies highlighted in the preceding section), my SAP customer shared with me a philosophy I now label "Bigger than me." Their approach was simply to bring in organizations that are "bigger" (in terms of number of employees, level of expertise, annual revenue or something similar) than their competitors, for the following reasons:

- Bigger companies will probably be around longer, surviving trying times, downswings in customer purchases or spending, and so on. This includes technology and point-solution companies as well as consulting organizations. Closer to home, the bigger consulting organizations have enough work to keep a stable SAP workforce employed, rather than relying on 15–20 W-2 employees and a roster of "phantom" employees gained through relationships with independent contractors and partnerships.
- Bigger companies by virtue of their size have access to more people. Thus, should attrition take its toll on the project team, the bigger company could more quickly backfill with newcomers who are almost equally qualified. Worse case, the bigger company could leverage its partner base, which is more extensive than that of a smaller company, and satisfy demand in this way, too.
- Bigger companies tend to be mature; they leverage proven processes, utilize sound billing and invoicing systems, carry insurance on their consultants, provide their people with the tools and equipment to do their jobs, and so on. This all adds up to a mature organization, preferable to the alternative even if everything else is equal.

Thus, in this particular customer's eyes, bigger companies were therefore "safer" alternatives to smaller organizations, generally speaking. My customer applied this philosophy to everything, going with the biggest players across the entire SAP Solution Stack arena rather than smaller specialty organizations. Looking back on their decisions and the number of smaller third-tier players that have disappeared or been swallowed up by others in the last year, it's probably hard to argue much with their approach!

Of course, they acknowledged two primary disadvantages to this approach. First, the cost per consultant tended to be higher than what could otherwise have been pieced together through smaller organizations comfortable with less margin (or burdened with less overhead). Second, "bigger" still generally meant longer times in getting things done than perhaps a smaller, more nimble company would be capable of. But with the benefits already noted, I was told that the rewards outweighed the penalties for this particular company. And in the end, they were still happy with their decisions.

Speaking of decisions, now is a good time to take a look at staffing decisions as they relate to bringing in new experts, or specialists, into the SAP TSO.

Let's Give a Big Warm Welcome to the "Specialists"!

Although most any SAP-experienced or trained resource can install an SAP component (in three to five days, usually, starting with uncrating gear all the way to logging in to the new unconfigured empty system), fewer can actually configure the product to be useful to the developers (who will perform their own "configuration" and create a functional SAP environment useful to end users). In Figure 12.6, I identify some of the most common specialist roles that are staffed at this stage of the project plan.

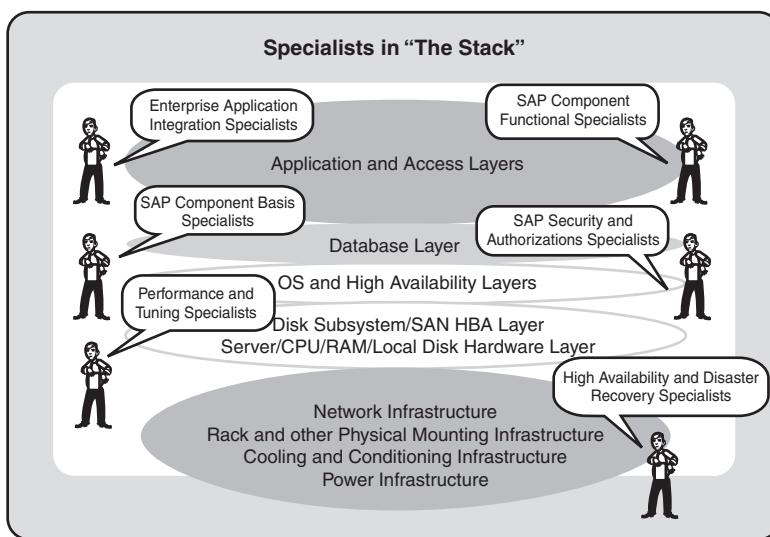


FIGURE 12.6 The need for specialists like those identified here varies based on the SAP implementation at hand; note, however, that they represent all layers of the SAP Solution Stack.

The next few pages detail some of the challenges inherent to identifying the best people in this regard.

SAP Component Basis Specialists

When it comes to expertise in rapidly installing and configuring an SAP product or component like SAP APO, Enterprise Portal, or SAP CRM, nothing replaces experience. Read that last sentence again—unless time is really a luxury, training just

doesn't cut it here—you require real experience. With so many mySAP components, experience makes the difference between doing it once and doing it 15 times. I know. People like me have already done it 15 times in the lab, or during training, or through various hand-holding exercises. It only makes sense for many companies to leverage this experience the first time through, teaming their own in-house SAP TSO junior specialists with seasoned consultants and integrators.

But I've been around and know that probably more than half of all actual SAP software installations are performed by the customer's internal staff themselves, in an effort to not only seek knowledge but retain it in the best way possible in-house. The problem arises when the SAP Component Basis Specialist is new to the team, and new to the particular component, or new to one or more of the SAP Solution Stack layers inherent to the solution at hand (like a particular database, or specific approach to high availability, for example). Not only does the "newbie" have to figure out how to get things done, obtain or get access to everything, and come up with a plan for installation, he also has little to fall back on in terms of "backup." More often than not, this person's backup consists of SAP online support/OSS, a few experienced colleagues who tend to answer their cell phones in the middle of the night, and so on.

In most cases where a customer is dead-set on using an internal person, then, it makes sense to transfer responsibility for new SAP Basis installations to current SAP TSO members, or at least to current IT team members transitioning into the TSO. Get them the training, give them a reasonable timeline to achieve success, and move out of the way.

Worse case, if an organization still needs to go "outside" to bring someone into the SAP Technical Support Organization on a full-time or "project" basis, consider the following qualifications and background of the *best* candidates:

- They have maintained a relevant set of core SAP Basis skills, and have added new or cutting-edge skills to this foundation.
- They tend to have a long list of not only repeat business, but repeat "make it happen" stories of successfully installing new SAP products.
- They possess a background characterized by diversity and adversity. The best folks have proven themselves in different and difficult environments, supporting a variety of solution stack alternatives and approaches.

The best outside Basis candidates have also probably concentrated to some extent on a particular mySAP solution, like supply chain or core R/3 or similar. In doing so, they therefore tend to have knowledge of the "open technologies" that SAP leverages to power these particular mySAP applications, like J2EE, Java, XML, .NET, and so on. Or they might have exposure to SAP's integration technologies like BAPIs, SAP BW

Business Connectors, WebAS, and so on. In either case, the ideal candidate's general background of diversity is adorned by a string of recent successes in the particular product suite at hand.

SAP Component Functional Specialists and Programmers

With functional specialists and programmers, as with their Basis colleagues, top-notch functional and programming expertise equates to actual experience. These positions lend themselves to being filled by consulting organizations because of this fact. However, if an employee must be brought in, and the person in question is not directly experienced with the component or the particular mySAP solution, take a close look at their background. When it comes to newer components like Advanced Planner and Optimizer or Strategic Enterprise Management, even a novice functional specialist should still exhibit a solid foundation in the general relevant functional areas, for example, coupled with relevant training.

Let's consider a candidate being considered for an APO position. If the candidate's background is at minimum not manufacturing- and logistics-oriented (that is, MM/PP, with experience in profitability analysis), look elsewhere. Similarly, the best new CRM candidate will be experienced in sales, service, pricing, and other order management-related functions. And some of the best SEM/BW and SAP Portals people will consist of long-term FI/CO consultants.

In all of these cases, the top-of-the-heap candidates will also add a technical edge to their functional core competencies. As with everyone else in the SAP support team, the better the solution is understood up and down the solution stack, the finer the support that can be provided. This technical edge usually manifests itself in a complementary area, like what is discussed next—the EAI specialist.

Enterprise Application Integration Specialists

One of the most difficult of all staffing holes to fill in the SAP TSO is the role of the *Enterprise Application Integration* (EAI) specialist. These are the people who do the magical quasi-basis quasi-functional quasi-programming stuff, like setting up systems for EDI, or integrating third-party and legacy applications with the new SAP solution, or developing special integration points for faxing, Web communication, eCommerce connectivity, and so on. In all cases, the best candidates tend to have experience with ALE, XML, BAPIs, the SAP Business Connector, Exchange Infrastructure (XI, the replacement to the Business Connector, used to easily maintain interfaces to other systems), and third-party tools like webMethods. Sometimes this set of expertise is loosely referred to as the *SAP-EAI toolkit*. This toolkit, coupled with solid experience in the appropriate SAP components (like adding SAP Business Warehouse or Supplier Relationship Management to an existing R/3 implementation, or simply tying multiple R/3 instances together with ALE) creates a solid SAP technical integration background.

Note, too, that experience and exposure to these integration technologies will greatly enhance a newcomer's ability to integrate SAP Portals and SAP Web Application Servers into new and existing SAP implementations. And given the fact that APO, CRM, and SEM will all be running on or via the foundation provided by SAP Portals and WebAS, the SAP-EAI toolkit will only grow in importance in the next few years. Couple this experience with deep SAP BW experience—as BW represents the primary data repository leveraged by APO, CRM, SEM, and other components—and we will create a veritable SAP integration dynamo.

Other solution sets may prove critical as well. For example, when it comes to SAP Strategic Enterprise Management, experience in pulling data from other systems into SAP is key. SEM relies heavily on messaging or message-bus functionality afforded by software like TIBCO, MQ Series, iWay, and many others. This is because every system in a company's enterprise is ultimately folded in to SEM. Thus, the ideal SEM candidate not only understands SAP and SAP integration, but also general networking, systems integration, and data migration from non-SAP systems.

And contrary to what some folks might think, my prediction is that Microsoft's .NET initiative will color more than a few SAP implementations in time. Not that J2EE (or ABAP!) will fade away anytime soon, or SAP's Business Application Pages will disappear as fast as they came. Rather, the folks in Walldorf, Germany, understand that continued growth in both the mid-market and already crowded enterprise space necessitates integrating well with Microsoft's product lines.

So be prepared to leverage candidates who have gained experience or exposure not only to .NET, but also SAP's more mature EAI offerings. And be prepared to budget and staff small teams of integration specialists, rather than waiting for a single Superman to satisfy all of your integration needs. Bottom line, the arena of enterprise application integration will only continue to grow. And as SAP Portals, SEM, CRM, and other pent-up projects begin kicking off in earnest as the global economy strengthens, I believe that EAI expertise will be as sought-after as any SAP skillset to date.

Performance Specialists

Another highly sought-after staffing position is the Performance Specialist. Why? Because ultimately, the users of the system tend to really just care about how little they see an hourglass on their screen. Sure, the system needs to be configured correctly, and protected appropriately against disasters and so on. But it's system performance that ranks consistently as #1 or #2 in so many post-implementation user surveys.

Performance specialists vary in terms of the specific layer of the SAP Solution Stack in which they specialize, but they can absolutely make the difference in a mediocre

implementation versus a system characterized by blazing-fast user response times. The best of the best

- Understand the entire solution stack and how one layer impacts the performance possible from other layers.
- Possess a background steeped in multiple layers of a company's particular SAP Solution Stack.
- Hold a core competency in a particular layer or two of the stack. That is, in all of my travels I have yet to find more than a handful of true end-to-end SAP performance experts. Instead, a team of two or three people typically tends to be required to address a real customer's environment when performance issues crop up.

The most common performance and tuning specialists are predisposed toward the general solution stack layers. For example, I see a lot of server/disk subsystem hardware specialists, and perhaps just as many OS specialists, and a whole lot of network and data center experts. After this, the field narrows a bit, and fewer database and mySAP performance tuning specialists are uncovered. It is the latter that are most difficult to find, especially with complementary knowledge of your specific hardware and database platforms.

The surest way of finding expertise for new implementations seems tied to the specific vendors. That is, for the best Microsoft SQL Server experts, I tend to lean on Microsoft. For Hewlett Packard Superdome or HP/Compaq ProLiant expertise, see HP. For superior OS support, dig up the OS vendor's professional services organization. And keep in mind what I said before—it's nearly impossible to find SAP employee prospects or paid consultants who embody experience in every technical aspect of a solution.

When an SAP system landscape matures, and the SAP Infrastructure/Basis team matures with it, though, the body of knowledge created and developed onsite becomes quite impressive. Eventually, a few of the support staff become quite adept at maintaining, troubleshooting, and yes, tuning, their own specific solution. It is these folks who you fight to keep on staff! And for new implementations, it's this kind of people you want to somehow locate for the benefit of your own company's deployment of SAP. It's no wonder that probably 50% of these top-knowledge workers historically have entered the consulting ranks after spending time on "their" own company's SAP implementation or upgrade learning the ropes—which then brings you back to the various vendors, partners, systems integrators, and consulting firms.

What about the remaining 50%, the folks who stay loyal or otherwise tied to their own company, supporting the ever-maturing SAP deployment in which they learned so much? How do you make them part of your team? I subscribe to the following:

- You can't have everyone—if they haven't left, there is something keeping them at the company. Call it golden handcuffs, fear of the unknown, lack of the desire to see their families only on weekends, or simply joy in the position they hold today. Whatever it is, most of these people are simply not interested in working elsewhere. You will therefore have to toss a whole lot of money, quality of life, or something similar their way to even get their attention. But all of this is almost moot, in that you will be hard pressed to actually find these folks in the first place.
- Some are not ready yet—Look for the ones on their way up in their respective organizations, people who are "maturing" in their capabilities, who will eventually join their colleagues in both the consulting fields and at other companies. Until then, they probably represent junior people at best, and therefore a risk to some extent. But these folks are certainly worth phone-screening (and interviewing, if warranted). And depending on the capabilities required, or a particularly narrow skillset desired, they could actually turn out to be excellent fits into your organization.
- Some you don't want—the remainder are probably on their way out, possibly because they are not wanted or needed any longer. Beware of this bottom 10–20%, the SAP stragglers! You will happen upon these folks quite easily, as their resumes will be everywhere on the Internet, and in half a dozen recruiter's files. I only hope that you heed the interview process outlined in Chapter 4, check references diligently, and so on—the process, when followed, should weed these folks out nicely.

In retrospect, although the preceding considerations were tailored to Performance Specialists, the breakdown could easily apply to many SAP fields of employment, and perhaps IT as a whole.

SAP Security, Access, and Authorizations Specialists

Security is a whole field in and of itself. But when it comes to SAP, security takes on additional significance in that a good security specialist must have an understanding of each layer of the solution stack, especially in terms of security holes or weaknesses. For example, a good SAP security candidate will possess the following skills:

- Proven and demonstrable experience in the setup, configuration, and maintenance of SAP's authorizations.
- As required, experience with Central User Administration, Global User Manager, or other user management approaches. This could easily include

experience with Enterprise Portal or the older SAP Workplace Server, as well as Internet Sales and Internet-managed users for SAP's procurement systems.

- Security analysis experience using R/3 WorkStream in Role and User Master development.
- A clear understanding of TCP/IP-based networks.
- Solid experience with firewall technologies, including SAProuter and some of the excellent products from companies like Checkpoint, Axent, Cisco Systems, and Nortel.
- Solid experience with the Operating System upon which the SAP landscape is built, from a security perspective, including Internet-access ramifications, directory security, and familiarity with virus protection products from companies like Trend Micro, Norton, and others.
- Some level of experience with the database management system in question, again from a security and access perspective.
- Outstanding analytical skills.
- Outstanding follow-up and follow-through, and detail-orientation, as good security is all about dealing with details while letting nothing fall through the cracks!

Note the focus on experience in the different solution stack layers, including deep SAP security and SAP component/solution expertise. It should also be apparent that the need for detail-oriented individuals with awesome analytical abilities is key—this position is not for everyone.

With the skillsets identified in the preceding list, backed up by requisite experience, the SAP security specialist should be prepared to address the following challenges:

- Protecting the SAP network and hardware infrastructure against internal and external threats
- Working with corporate or headquarters IT to assist in efficiently managing the SAP-relevant subnets in terms of IP address infrastructure
- Implementing an open security solution that enables integration not only with SAP but also with future bolt-ons and other extensions to the enterprise
- Almost instantaneously detecting and responding to attacks or less obvious suspicious activity against the SAP enterprise environment
- Providing secure connectivity for all mobile and remote mySAP end users

- Developing and maintaining SAP user-based roles, authorizations, and associated access-related matters
- Working with vendors, suppliers, and other business partners to provide selective network and SAP application-layer access to them (through a secure extranet or similar approach)
- Adhering to SLAs (service level agreements) derived from the various end-user communities and other business entities to provide a secure network characterized by high performance, reliability, and availability
- Defining, monitoring, and enforcing OS-level, DB-level, and SAP-level security policies and authorizations, both across the entire SAP system landscape and up the solution stack

Will such an individual prove difficult to find? Yes! So start your search early. As an interim solution, an external consultant might be pulled in. Ultimately, though, such a position is best filled by a long-term employee—the knowledge gained in performing this role is exactly the kind of knowledge that most companies should prefer to keep “inside” the organization.

High Availability and Disaster Recovery Specialists

Although High Availability and Disaster Recovery can be quite different from one another, I have chosen to address them together because both are enabled by solution-specific technologies. Further, the best high-availability personnel understand not only that disaster recovery is simply the logical extension to good HA practices, but also that HA and DR needs are driven by the business. With this said, truly top-notch HA Specialists, like DR Specialists, are difficult to find. Therefore, specialized DR consulting houses are often leveraged for deep expertise in this regard while the client builds up general or day-to-day expertise in managing and supporting the HA or DR solution previously architected and installed.

- To learn more about the role that HA and DR specialists play in terms of the actual solutions that they can support, see “The Disaster Recovery Organization,” p. 209 in Chapter 6.

Highly experienced HA/DR folks are also adept at managing changes to your HA/DR solutions, such as those resulting from upgrades due to either business or technology factors. And the best HA/DR people are experts in the products underpinning your specific high-availability approach. For example, if your SAP solution architecture called for implementation of a particular flavor of a UNIX-based active/active cluster, the ideal HA person will already have this expertise. And to take it one step further, the best candidate will also be experienced in the specific database version and

release to be clustered (whether this is Oracle, Informix, DB2, or another), as well as the particular component and version of mySAP being implemented.

With such specific requirements, the best candidates typically come from a polarized background (similar to Performance Specialists), where core expertise is found in the primary solution stack layers of hardware, operating system, database, and SAP itself. Thus, I tend to see companies turning to Oracle for the most experienced Failsafe and 9iRAC consultants, to Microsoft for the best cluster service support, to particular database vendors for their best-practices approach to log shipping or transaction replication, or to specific hardware vendors like HP when it comes to supporting its various stretched cluster products and methods.

Finally, the HA and DR Specialists you prefer to have on your own team understand both your underlying solution architecture and business constraints within the context of maximizing uptime. In other words, High Availability professionals never lose site of the fact that they exist to minimize planned and unplanned downtime. As such, they are accustomed to doing what it takes to make their system available—working unusual hours, constantly balancing technology and business requirements, doing what's best for the team and the customer, and managing themselves under the umbrella of important service level agreements are all integral parts of an HA/DR Specialist's life.

Testing, Data Conversion, and Other Specialists

Based on your unique implementation, additional SAP specialists will be required at some point in time. For example, I have worked with and been a part of teams responsible for stress testing, regression testing, integration testing, data conversion, legacy integration, and more. My team and I assisted one company with developing their complete promote-to-production change management processes. In another case, I trained operations and help desk staff in supporting SAP R/3 on an interim day-to-day basis while a holistic enterprise management approach was still being worked out. I have also taken on numerous team-lead roles on projects, where I was held responsible to complete miscellaneous tasks and support timelines associated with almost every specialty mentioned in the last few pages. My point in describing all of this is just to remind you that not all tasks and roles fit neatly into the areas previously discussed, or map perfectly to the SAP Solution Stack; as the Master Project Plan illustrates, there are a lot of “miscellaneous” activities that simply need to get done by *someone*, on time and on budget.

Regardless of the activities or role, I suggest that you give thought to the following thoughts and practices:

- The more critical the position is to the deployment, the more lead time you require to find the best resource.

- In the case of short-term positions (like stress/load testing, where start-to-finish might only mean a few weeks or months), pay for experience rather than attempting to “build it yourself.”
 - Where long-term positions are merited, seek to hire a qualified resource long-term as well; this is much safer, cheaper, and smarter than hiring a long-term contractor or consultant, with the added benefit of building core experience in-house.
 - Jump-start your own team through the measured use of subject matter experts. Get them in, use them, learn from them, and get them out.
 - Build timelines into your project plan such that they allow you or your team the luxury of doing the work of your SAP implementation yourself, while an experienced consultant stands by ready to assist and add value as requested. In this way, your implementation knowledge is built internally, not passively acquired later.
 - In most cases where third-party resources are utilized, focus on knowledge transfer (learning from the third party) as much as on getting the actual job at hand completed.
- For more detailed information on the various functional/systems integration and stress-testing roles, including what each encompasses, see Chapters 15 and 16, respectively.

Be sure to kick your phone-screen/interview process into high gear again, via the Rapid Deployment Approach to staffing described in Chapter 8. And in the meantime, with the need for SAP specialists identified and behind us, let us turn to a much less obvious need. Surprisingly, it is also one of the most overlooked and underutilized SAP support functions established during and after the implementation.

The Most Overlooked Critical SAP Support Function

Although it is true that many SAP specialist support functions and skillsets can be called critical, I have consistently found one important area that tends to be overlooked until the last minute—Computer Operations for the SAP enterprise, or the SAP Operations staff.

SAP Operations Expectations

Most often, the team tasked with managing and generally supporting the SAP infrastructure when it is beyond special-project status is labeled “Computer Operations”

or simply Operations. For our purposes here, I wish to label all but the most basic outfits “SAP Operations.” This team usually already supports other mission-critical computing systems, is staffed or organized to do so 24x7, and is familiar with processes and procedures geared toward ensuring that the systems under their charge continue to operate smoothly around the clock.

Rather than leveraging this team, what tends to happen is that the core SAP project team and a bunch of consultants gain all of the experience germane to supporting SAP once implemented, but then fail to share this knowledge with the operators. Bad move. Expensive move.

I like to think that one of the best-kept secrets in any SAP implementation is the Computer Operations group. What *should* occur soon after the SAP Data Center is developed or built from scratch, many months before Go-Live, includes the following:

- The SAP Infrastructure Manager or team leader should establish a basis for an ongoing relationship with Computer Operations. This is best accomplished by including operations to some degree in the installation and configuration stages of an SAP project, as well as including them in regular status meetings and the like.
- A smooth transition of all of the services handled by the SAP project team must occur. This equates to clear communication and delineation of tasks that need to be transitioned, including hand-off of day-to-day operations and other similar documentation that *works!*
- Each team must be cognizant of, and focused on, minimizing disruptions to ongoing business. That is, operations must continue to support their current computing environments, and the SAP project team must continue to ensure that the needs of the developers, folks training on the current systems, and other pre-production users are satisfied.

SAP Operations can play a key role in ensuring high availability and as-expected performance, *if they are trained, enabled, and kept in the loop*. In the next few sections, I take a closer look at exactly how this often-underutilized resource can provide huge value to an SAP project.

SAP Monitoring—Ensuring Highly Available Systems

When it comes to monitoring your SAP system, entire volumes can be written on administration activities that must take place on a regular basis. Indeed, quite a few books exist on the topic of SAP Administration. But when it comes to actually

performing various operations tasks, tools, and approaches like those illustrated in Figure 12.7 and described in the following list they are sorely overlooked. Consider the following:

- The SAP Operations Manual, which embodies everything when it comes to ensuring that SAP is up, available, and safeguarded against loss or disaster.
 - Documented daily procedures, the actual checklists that can be used for daily regularly scheduled activities (at least until an automated approach is developed via an SAP-aware enterprise management tool). I like to use CCMS, especially transaction SSAA, to drive much of this.
 - Other documented regularly scheduled procedures, including weekly, monthly, quarterly, semiannual, and annual operations activities.
 - Well-documented procedures and approaches to systems backup/restore and complete rebuilds, such that even a novice could perform a restore (in the event of a disaster, for example).
 - Leveraging CCMS and other manual processes/checklists, including using scripting utilities to take the burden of data collection off the operator.
 - Approaches to defining enterprise management requirements for SAP, and then evaluating and selecting the best Systems Management application for your SAP environment.
 - Various approaches for sharing all of the documented procedures and processes mentioned in the preceding list, including using standard Web sites/Web pages, to full-blown portals/knowledge repositories, to simple Word and Excel documents within a structured file-share.
 - Processes for accurately maintaining your SAP documentation, as well as how to regularly review and update it.
 - Practice drills for improving operations and availability; restoring backups, testing cluster failovers and disaster recovery failovers, and mimicking other failures up and down the entire solution stack.
- To learn more about the role that SAP Operations plays in terms of systems management and monitoring, and exactly how all of the aforementioned approaches can be employed, see “Systems Management Techniques for SAP,” p. 511 in Chapter 14.

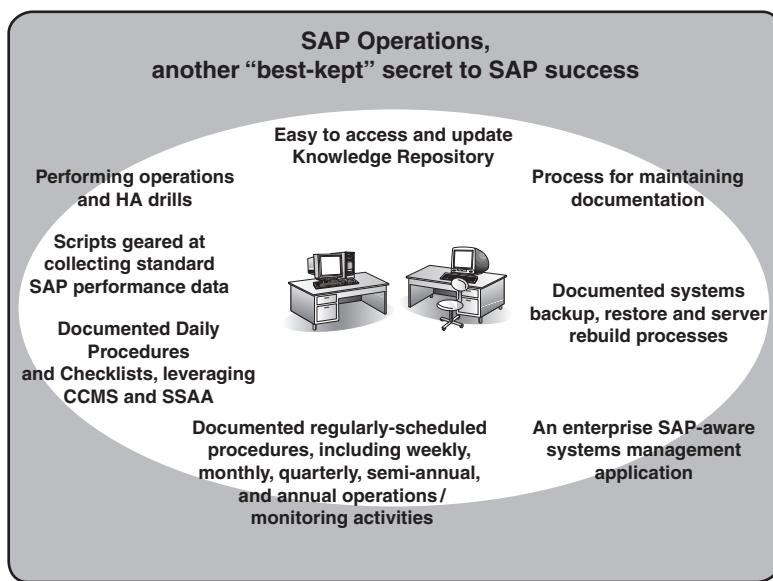


FIGURE 12.7 One of the best-kept secrets to supporting SAP lies in pushing existing computer operations teams to “step up” to an SAP enterprise operations role.

Taking Advantage of SAP Operations

As I stated previously, SAP Operations represents a perfect breeding ground for future SAP TSO professionals. It only makes sense—business-wise, financially, and employee-wise—that the Computer Operations team be pushed to support SAP infrastructure changes as they occur, be engaged in monitoring, assist in implementing changes to the solution stack, and more. Given their knowledge of the foundation, and the company’s overall legacy computing environment, a good operator is well-positioned (with appropriate training, mentoring, or experience) to move into SAP TSO roles like junior-level database administration, SAP basis/infrastructure, and even entry-level enterprise integration roles. Given this, I believe that both line management and your project’s management circle needs to be very aggressive about providing training and education to these folks.

From a day-to-day perspective, SAP Operations can prove highly useful to the organization. Specifically, they can assist in kernel upgrades, perform infrastructure updates, apply support packages, monitor key performance indicators, help with SAP client maintenance, and more. They can run through the general and regularly scheduled activities outlined in transaction SSAA, reporting on the health of your entire SAP system. In essence, the SAP Operations team can easily become the “backup” to the support staff primarily responsible for such activities, and thus serve

a couple of purposes—they help the company not only mitigate risk, but also provide promote-from-within opportunities, thereby attracting additional talented folks naturally drawn to such organizations. And they allow the senior SAP TSO members the luxury of more time to look down the road at strategic issues, freeing them from some of the tactical day-to-day tasks so readily seized by career-minded operators.

Change Control—Where the Rubber Meets the Road

Much of the activity involved with implementing and managing SAP involves managing change. Changes are guaranteed to occur, and will affect every layer of the SAP Solution Stack. Perhaps the most common perception of change as it relates to SAP is geared toward functional changes. That is, the customizers, programmers, and other developers continue to enhance functionality of the SAP system throughout its life. Support packages also add functionality, and more often resolve issues in different functional areas. Legacy systems are tied in and retired. Through all of this, the system evolves, and indeed controlling these changes is paramount to ensuring that the changes are both stable and capable of being “backed out” if necessary.

But change occurs within and between all layers of the SAP Solution Stack, not just inside the application layer. And our SAP Operations team plays a critical role here, too. For it’s here that the “rubber meets the road.” All of the planning, research, and testing inherent to each change in the stack culminates in a change to Production. So unless the SAP Basis team is dead set on supporting every single operating system patch/service pack upgrade, or database update, or hardware/firmware upgrade, it just makes good business sense to leverage a company’s investment in its computer operations staff—they’re already onsite, are anxious to learn, and are almost certainly interested in increasing their SAP literacy.

The same can be said of the next most often-overlooked resource in a company’s IT organization, the help desk staff. I take a closer look at this next.

- ▶ To better understand the role of change management and change control in an SAP project, see “Change Management Best Practices and Approaches,” p. 469 in Chapter 13.

The Role of the SAP Help Desk

Like the SAP Operations team, the role that the SAP Help Desk or SAP Support Center will play in supporting the overall implementation (and perception of the implementation!) seems to be largely downplayed until the big day of Go-Live looms near. Simply creating a SAP-literate help desk will never make the support organization successful, though. Preparing a legacy help desk team to take on the world of SAP troubleshooting and problem resolution is no small chore. In the next few

pages, I'll discuss this from a number of perspectives (also clearly illustrated in Figure 12.8):

- Timing, in regard to staffing
- Breadth of questions and issues to be encountered
- Training, in regard to skillsets needed to address real-time troubleshooting and problem resolution
- Managing end-user perceptions

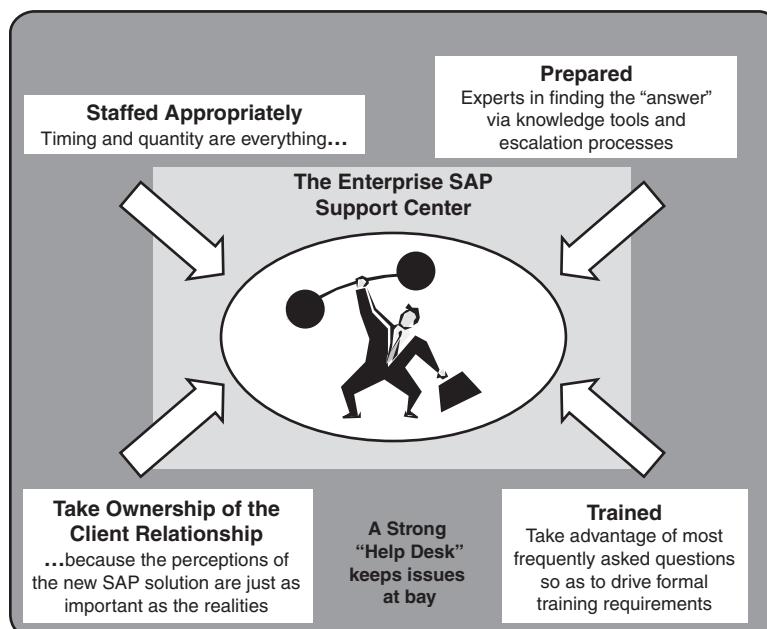


FIGURE 12.8 A Help Desk team focused on SAP issues, staffed appropriately, and prepared through proper training, will actually boost your end-user's perceptions of "the new way of doing things."

The SAP Help Desk will tackle issues large and small, with questions originating from end users as well as internal IT and SAP TSO team members. As such, the staffing and education of this team is vital to the smooth transition from supporting legacy system(s) to supporting SAP (and perhaps legacy systems concurrently). And given the fact that the Help Desk team represents the first SAP "face" that many new SAP end users will ever "see," their customer-facing role can simply not be underestimated. The Help Desk is in a position to leave each end user with either a favorable or a poor opinion of the new system, well beyond the issue that prompted their call to the help desk in the first place.

Staffing the SAP Help Desk

Without question, staffing the Help Desk with enough properly trained, experienced, and pleasant, even-tempered, customer-oriented personnel is critical—all of the education, experience, and tools in the world will be worthless in the hands of a lone help desk technician a couple of days after SAP Go-Live. In my observations, I see help desk staffing in terms of a typical bell curve, where the peak of the curve represents the first few weeks after the big imaginary Go-Live button is pushed. The need for on-call telephone support staff at this time is great, as one of the largest volumes of calls tends to occur in this time frame.

As calls taper off after the first month or two, though, it's quite normal to scale back the Help Desk. Downsizing tends to follow the growth of the most frequently asked questions (MFAQs, or simply FAQs)—as MFAQs stabilize, the number of help desk calls and therefore technicians shrinks. In other words, when the top 20 questions become a matter of routine, it's usually time to replace some of your paid bodies with pleasant on-hold telephone recordings, Web-based MFAQs and associated answers, and so on—refer to Figure 12.9 for clarification.

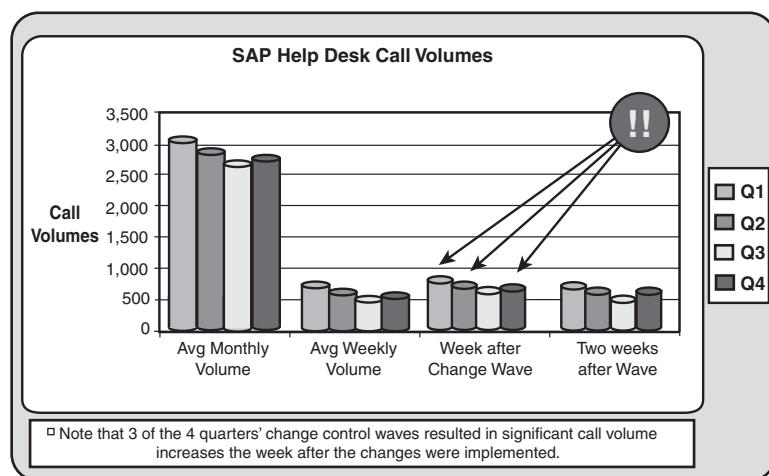


FIGURE 12.9 As the SAP implementation matures, the number of support calls drops, only to rise and fall again slightly after each change wave.

The Help Desk also tends to take additional customer calls after new functionality is added to the system, especially when this new functionality represents a new component or consists of a new method of accessing SAP. Call volumes reflect these changes, often implemented as quarterly change waves. In one case, in fact, I observed huge call times at one of my accounts where SAP Workplace was introduced to facilitate “single sign-on” (access to all SAP components was granted

through this simple portal approach), but communication about exactly *how* to access Workplace had been lacking. The result was a whole lot of extra calls the first two days of the week, from people just interested in logging in to their system.

In another case, when an account finally added the Sales and Distribution module to their core R/3 functionality, but failed to adequately resize or re-evaluate their in-place hardware platform in light of their really poor SD implementation, everyone suffered. Response times went through the roof, and the help desk was swamped with calls for days from users across the board—MM, FI/CO and more, in addition to the unfortunate new SD end users.

In the preceding cases, attention to detail and some level of testing prior to major changes in functionality would probably have taken care of both of these issues before they ever came to light. But both cases also serve to underscore the importance of the SAP Help Desk as a final net, when other nets fail to catch issues, or are used improperly.

Common SAP Help Desk Questions in the Real World

In the name of “pre-go-live training” for one of my clients, I assembled a list similar to the one that follows of common SAP help desk questions and scenarios. This list is by no means comprehensive. For a larger sampling, though, including what I have found to be the most common resolutions/fixes, refer to the Planning CD. In any case, keep in mind that such a “task list” can be easily transformed into the foundation for an SAP Help Desk training plan, too:

- A user cannot print. List two reasons.
- A user cannot log in. List four reasons.
- Explain the process to change a user’s password.
- Response time “seems” to be slow, per a user who has called the help desk—prove that the system is indeed performing to spec by digging out application server statistics and other real-time performance data.
- A user can’t run a particular transaction. List three reasons why.
- A user calls in, and says his job is not completing. How do you confirm this?
- Explain the purpose of Change Management or Change Control to a user wishing to have a change to production implemented immediately.
- Which T-code can you execute to determine whether the previous night’s backup actually completed successfully?
- One of the DBAs calls in and needs to know how big the database is. How do you determine how much of the allocated disk space is actually being used by Oracle or SQL Server?

- The SAP Infrastructure/Basis group calls in. You need to look at table locks or lock entries. Which T-code do you execute?
- A user says that they just ran a transaction and an ABAP dump occurred. How do you look at this dump, or any dump over the last few days?
- During troubleshooting, you determine that the system log generated over the last few days should be studied—how do you do this?
- Explain how to add the ‘SAP_BC BASIS ADMIN AG’ activity group to user ZGEANDE.
- Management just did another forced early retirement, and you have been requested to lock user ANDEGE. How is this accomplished?

I recommend that you add to the more comprehensive list found on the Planning CD as required, and use it to quickly train new hires, or refresh folks who might be shared between various IT help desks or support centers. And as I already stated, use this list to begin assembling a training plan for your SAP Help Desk, covered next.

A Simple Roadmap to SAP Help Desk Preparation

When the appropriate staffing plan is in place, both education and access to knowledge tools tend to be the real keys to a first-class support center. Preparing the SAP Help Desk for their new support role really amounts to the following:

- Mapping tasks and issues (like those described in the previous section) to those that will actually be faced by the client. This includes paying attention to most everything at or above the SAP Application layer of the solution stack.
 - Determining the baseline knowledge of the help desk staff. This is often done through IT product and SAP-specific surveys, sample questions, and so on.
 - Determining the need for formal training versus on-the-job training versus informal employee or consultant-provided knowledge transfer, depending upon the baseline knowledge of each help desk technician, and your budget, and timing.
 - Developing and sharing contact information, escalation processes and lists, and so on with everyone on the help desk team—this must also include a description of what each technical and business team is responsible for.
- For details on what “training” amounts to for the various technical organizations supporting SAP, see “Who Needs Training?”, p. 310 in Chapter 9.

Managing End-User Perceptions

At the end of the day, the SAP Help Desk or SAP Support Center directly impacts how end users *feel* about the new business system. With this in mind, a knowledgeable and courteous support staff can make the difference between a user accepting a one-time short-lived inconvenience and the same user creating and sharing negative long-term system perceptions with his friends and colleagues at work.

This issue is huge, and needs to be recognized as such. When a perception is “out there”—a perception of poor performance or unacceptable uptime, for example—the perception tends to outlive the system itself. We are all familiar with the adage “you can’t squeeze the toothpaste back into the tube”—here is a fine example of that adage at work.

On the other hand, if the support folks are properly staffed, appropriately trained, and have access to the proper knowledge management tools, they will then be capable of identifying the root cause of a problem, or rapidly escalating the issue to the correct team. The perception surrounding the issue might very well actually turn out positive, in fact. That is, the end user might walk away with a sense of pride in the professional manner in which he and his issue were addressed. This, followed by a timely resolution of the core problem, might even instill more confidence in both the system and support team, on behalf of the end user.

Bottom line, invest in your SAP Help Desk. These people represent the front line of your SAP support army, and as such signify a great opportunity to help their customer—and your SAP end users—be successful.

Putting It All Together in the Real World

I have touched on quite a few critical and otherwise important roles necessary to keep your SAP project moving forward. To give you a sense of how all of this fits together, though, including what an SAP Technical Support Organization might actually look like for different-sized customer environments at this stage of the project plan, I assembled the following real-world scenarios. Each is real in the sense that the organizations described in the next few sections are bona fide SAP installations. However, I have taken the liberty to augment specific areas to either make a point or to streamline very large organizations, and have also changed other entities or telltale bits of organizations to ensure that client confidentiality is preserved.

A Sample “Small Business” SAP Support Organization

The first organization I will discuss is a relatively small manufacturing company with about 2,000 employees, four major plants, two core product lines, and overall revenue measured in the tens of millions of dollars per quarter. The SAP team

supports approximately 600 users, of which perhaps 33% are online concurrently during the day. Other SAP TSO-specific staffing data includes

- Basis/Technical Staff: Three.
- DBA and other Disk Subsystem Specialists: Two (two of the three folks previously identified as part of the Basis/Technical Staff).
- Other Infrastructure Specialists: Two, specifically network and Operating System specialists.
- Programming and Functional Support: Nine (six programmers and three functional experts, all of whom address day-to-day maintenance of the system as well as legacy integration points, reporting needs, and so on).
- Security/Access: two (one *full-time equivalent*, or FTE, plus the equivalent of one of the Basis/Technical staff).
- Change Management: Two (two of the same FTEs as the Basis/Technical staff).
- Enterprise SAP Operations: Four (two dedicated operators, and two of the SAP Basis/Technical staff), though a contract firm is used for staff supplementation on occasion.
- SAP Help Desk: Three during normal business hours (though this equates to one full-time equivalent, as they also support other enterprise applications hosted within the company). Note that Help Desk is covered by SAP Operations after hours.

It should be apparent from the preceding list that many of the SAP information technology staff wear multiple hats out of necessity—the company simply does not have the luxury of additional headcount to pull in as many “dedicated-to-SAP” individuals seen in a lot of larger SAP shops. Thus, specific technology areas like “Security,” “Database Administration,” and “Network Infrastructure” are covered, but not to the depth you might see in bigger organizations. As a result, the onsite team tends to be overworked simply by virtue of their day-to-day responsibilities. Special projects often require the assistance of third-party resources, too.

Despite the organizational challenges and “stressed” nature of this customer, I really like working with these folks. Their employee turnover is negligible, which indicates to me that they take care of their people and it’s a pretty good place to work. And they prove at many levels that SAP is not successful only in large organizations—even smaller firms can realize a decent return on investment when their implementations are focused on functional and data integration, and staffed appropriately.

A Sample “Medium Business” SAP Support Organization

Like my small business example, this “medium business” manufacturing company is also a long-time favorite customer of mine. Through acquisitions in the last five years, they have grown from 3,000 to over 10,000 employees and contractors, and from three primary plants to well over twenty. Annual revenue hit two billion dollars last year, and saw the SAP IT organization grow to nearly 100 people as they worked to pilot and/or implement a number of mySAP components in the last two years. Today, the core SAP team supports approximately 5,000 global users, nearly 1,700 of which are online concurrently during normal North American business hours. Other SAP TSO-specific staffing details include

- Basis/Technical Infrastructure Staff: Three employees and four contractors (with growth of plus or minus a contracted subject matter expert as new projects are piloted).
- DBA and other Disk Subsystem Specialists: Three (with some minimal overlap/additional coverage provided by a few members of the Basis/Technical Staff).
- Other Infrastructure Specialists: Six (including network and Operating System specialists and backup/part-time disk subsystem specialists).
- Programming, Reporting, and Functional Support: Forty-three.
- Security/Access: Three (overlap exists to some extent in terms of a “SAP Workplace roles administrator” that also addresses general IT security).
- Change Management and Documentation, as well as Enterprise Integration, all rolling up to a Security/Access group: Six.
- Enterprise SAP Operations: Six FTEs (does not include another two operators that are shared with traditional IT, and provide support for an offsite DR facility).
- SAP Help Desk: Two SAP full-time specialists support the system (in staggered shifts) during normal North American business hours. An additional three help desk folks backfill as required. Like my smaller example customer, the Help Desk role is covered by SAP Operations after hours. The complete team consists of seven folks.

As shown in Figure 12.10, this medium-size SAP customer requires quite a large investment in people, both from a technical support and a business-process perspective. It takes a large team to support so many end users.

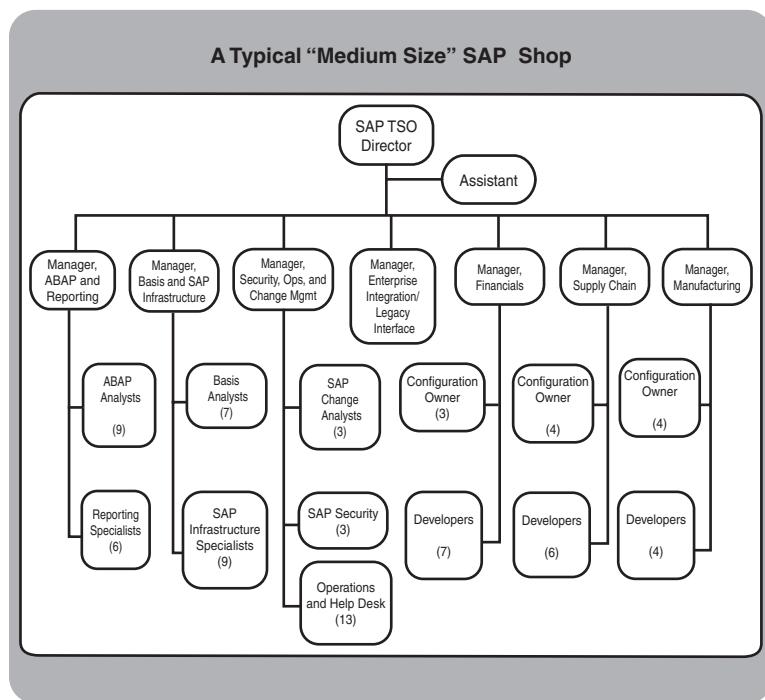


FIGURE 12.10 Note the division of duties and responsibilities within the SAP Technical Support Organization of this “medium business” example.

A Typical Fortune 50 Global SAP TSO

Although the term “typical” is arguable for a large organization running an SAP enterprise, my example illustrated in Figure 12.11 clearly reveals the level of investment in IT staffing. At nearly 300 people, this organization is a long way from being just another cog in the IT wheel. This IT organization supports tens of thousands of named users accessing multiple mySAP components and instances (R/3, BW, and CRM represent the core of this activity). Some of the more interesting staffing details include the following:

- Executive/Senior Management: Eight, which seems reasonable to me given the size of the organization.
- Basis/Technical Infrastructure and DBA Staff: Forty employees/contractors, covering both SAP and DBA support for the entire mySAP.com enterprise. This includes technology architects, SAP component specialists, SAP stress-testing support folks, batch job specialists, and at least six folks acting in a coordination role with other organizations like security, change management, interface support, and more.

- ABAP Programming and Forms support: Fifty folks (does not include functional experts, detailed next).
- Data Conversion/Functional Support: Forty, consisting of traditional business-process experts, but also including specialty roles like “Conversion Lead” (for each primary functional area), “SAP Tool Advisor,” “Rollout Coordinator” (either functionally or geographically focused), and “Cutover/Production Support Lead.”
- Legacy/Interface support: Fifteen, including folks dedicated to EDI and workflow. Note that specific teams have been deployed to address specific functional areas (for example, financials, orders to cash, purchase to pay, plant maintenance, and so on).
- Security/Access: Fifteen folks, focused on security and controls across SAP CRM, R/3, Workplace, BW, and APO.
- Change Management: Fifteen people covering business change management, communications, impact analysis, and related business readiness analysis.
- Integration Testing: Ten folks spend their time ensuring that all functional changes “work” together before each change is discretely deployed to Production. This includes regression testing and flow/integration testing, requiring both integration/semi-functional specialists as well as technology specialists. Responsibility for executing transports also resides in this group.
- End-User Training: Forty people responsible for all facets of training, such as design/development of curriculum specifically tailored for each of the deployed SAP components, special train-the-trainer programs for power users, process/flow documentation, and training tool management and oversight.
- Data Warehousing/Reporting: Thirty, focused mainly on supporting SAP BW (though four of the thirty support APO, and eight support R/3).
- Enterprise SAP Operations: Twenty-five, though some of these folks are shared with other business IT organizations, and at least ten support SAP Infrastructure more so than performing traditional operations activities.
- SAP Help Desk: Ten SAP full-time specialists, primarily contractors, not including overflow and off-hours assistance provided by both on-call staff and SAP Operations.

Many other folks support SAP in the preceding example, but from more of a business perspective rather than a technology focus. In fact, although not detailed in the preceding list, or in Figure 12.11 (due to space constraints), over 100 additional people are tasked with supporting functional areas like General Ledger, Financial Reporting, Cost Accounting, Material Management, Inventory and Warehouse

Management, Supply Chain, eCommerce, and more. These people keep the SAP TSO grounded and focused on providing a business solution to the company's employee base, not simply a "really cool" SAP technology solution.

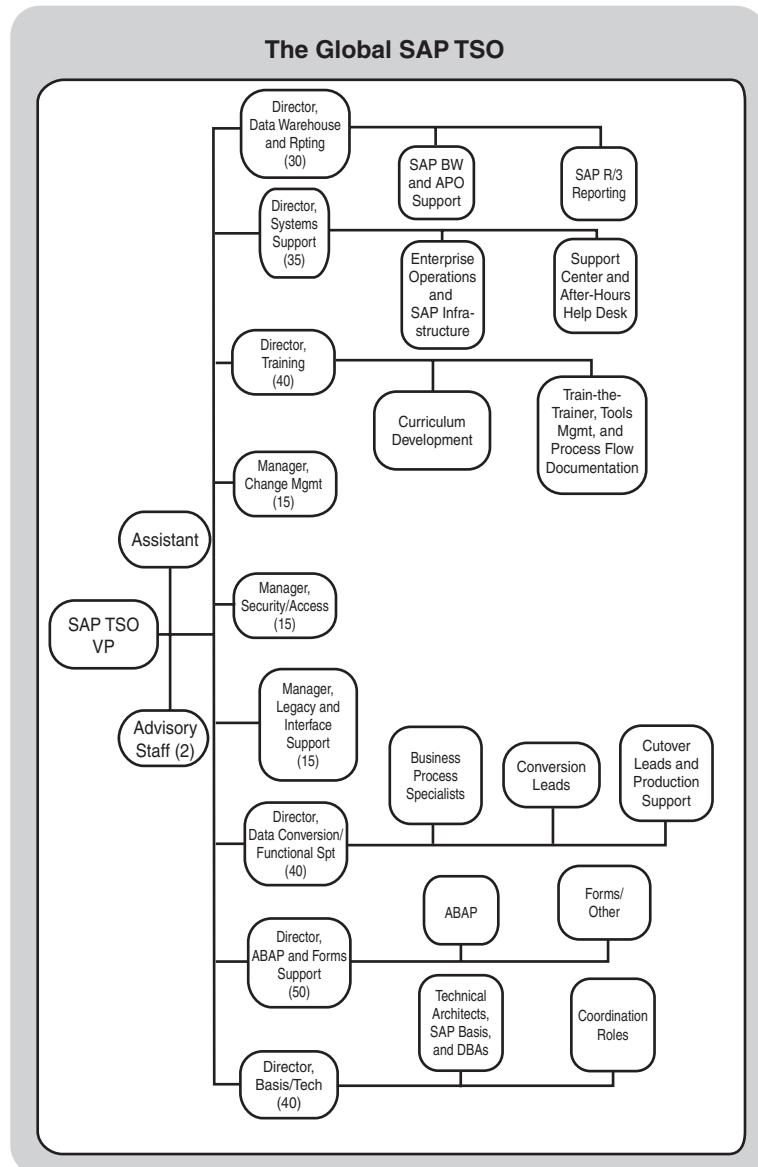


FIGURE 12.11 Note the even more diverse separation of duties and responsibilities within this large SAP TSO tasked with supporting tens of thousands of users.

In the end, as you've seen in the three SAP organizational examples, supporting SAP is not an exact science. Organizations vary considerably in terms of responsibilities, and boundaries between different groups tasked with supporting the various SAP Solution Stack layers are easily blurred. But I believe that the actual customer examples shared here represent what it takes to keep SAP up and running in complex, real, business environments.

Tools and Techniques

On the Planning CD, I have enclosed the organization charts illustrated in this chapter, plus the other figures. I have also included the SAP Help Desk sample questions and scenarios discussed earlier, a sample excerpt from a recent SAP Comparative Analysis, and a few new job descriptions.

Summary

Throughout this chapter, I spoke of how and when to insert the final pieces into your SAP support organization. Your SAP TSO has now evolved into something that will bring you through the first year or two after the system is put into Production. Some teams will shrink, and others will grow, as the user population changes or requests additional functionality out of SAP. Overall, though, roles in the following areas will remain both constant and critical:

- SAP Component Basis Specialists
- SAP Component Functional Specialists and Programmers
- Enterprise Application Integration Specialists
- Performance Specialists
- SAP Security, Access, and Authorizations Specialists
- High Availability and Disaster Recovery Specialists
- Testing, Data Conversion, and other Specialists
- SAP Operations/Monitoring Team
- SAP Help Desk or Support Center

With your support organization complete (or nearly so), we are now in a position to take a closer look at how many of these new folks actually spend their time. In the next few chapters, I'll cover change management, SAP operations and systems management, integration/stress testing, and readying the SAP environment for its end-user community. There is certainly enough work on the horizon to keep our newest members of the SAP Technical Support Organization busy, as we come closer and closer to the big day of Go-Live.

PART III

Moving into SAP Functional Development

IN THIS PART

- 13 Getting Your Hands Around Change Control
- 14 SAP Systems Management and the Operations Manual
- 15 Integration / Function Testing

13

Gaining Control of Change Control

An Overview of Change Management

At this point in your SAP implementation, your solution vision has materialized in front of you—the SAP Data Center and quite a number of SAP instances are finally running, and the development team is making solid progress by now. Your SAP Technical Support Organization is busy planning for the production phase, too. All of this activity is at serious risk, though, if you do not give attention to the process of managing change. Addressing *change management* or *change control*—the terms are interchangeable—is “just another critical cog” in the machine we call SAP. As with any enterprise or mission-critical system or application, how an organization manages changes to that system must be defined. Thus, before we take a look at best practices and approaches regarding change management, a quick review is in order. Later, I’ll highlight its value and importance in terms of maintaining a highly available SAP system, as well as covering information on organizational structures. I’ll then wrap up the chapter with real-world advice and lessons learned.

Change Management Mentality

A productive mySAP solution environment is not expected to remain static. That is, as functional and business requirements, software upgrades, OS and hardware updates, and other subtle changes are requested or required, the environment will evolve over time. Although this evolution is good in terms of keeping a customer’s employees productive and informed, an enormous opportunity to introduce instability into the production

IN THIS CHAPTER

- An Overview of Change Management
- Change Management Best Practices and Approaches
- Change Control Affects Everything
- Change Control and the SAP Solution Stack
- How to Organize and Plan for Change in the Real World
- Change Management Lessons Learned in the Real World
- Tools and Techniques

environment exists, in the form of *changes* to the production system. Therefore, if these changes are not managed well, and managed consistently, the result over time will be much different than the desired outcome—the system will prove to be less than reliable, prone to unscheduled downtime, and more difficult to manage than otherwise required.

Fortunately, change management in regards to mission-critical enterprise environments has typically been taken quite seriously. I work every day with wonderful enterprise-wide examples of sound change control in action, from mainframe-class and large UNIX implementations to successful PC server-based enterprise environments. All of these companies have embraced what can only be described as a no-nonsense “mainframe mentality” to change control—it’s taken very seriously, and it shows in their high system uptimes, low unplanned outages, and ultimately happier, more productive end users.

Often, when the topic of change management or change control is brought up in the context of an enterprise application, managing the functional changes within the application layer itself seems to capture the center of attention. That is, all too often the focus is of a functional nature. True, much of what drives changes to a system is initiated on behalf of the end users and functional folks who reconfigure and test SAP’s business processes. But many organizations that are not end-user-based, not to mention technology drivers, serve to initiate change as well, as you see in Figure 13.1. And each change ultimately must be reviewed, approved, developed/implemented, and eventually migrated or promoted throughout the SAP system landscape. With little exaggeration, then, I can say that nearly every member of the SAP Technical Support Organization can therefore impact, or be impacted by, change control.

Managing and implementing changes is a very precise activity, fraught with system-wide ramifications if addressed less than absolutely correctly day after day. And it is often unfortunately a labor-intensive task, impacting functional teams, developers, SAP infrastructure support folks, database administrators, the SAP Basis team, SAP administrators, Enterprise Operations, the SAP Support Center, and more, not to mention the Training, Documentation, and Change Management teams themselves.

Yes, change management affects much more than just the SAP functional layer. The entire SAP Solution Stack is impacted. So, too, is the entire SAP system landscape and every phase of implementation. Your management team drives and is affected by changes as well. And finally, change management must focus on human and organizational success factors, too, because these factors greatly impact whether end-user productivity or behavior is indeed improved.

“Change” often goes against the very values held dear by members of an organization. This explains why changes in the daily routine or culture of an organization must be proactively discussed and not ignored. Our natural reaction to change, on the other hand, is to deny it at first, and then to simply resist it when we figure out

that the change will not “go away.” Eventually, a critical juncture emerges—the change is either accepted, or it is rejected, as Figure 13.2 illustrates.

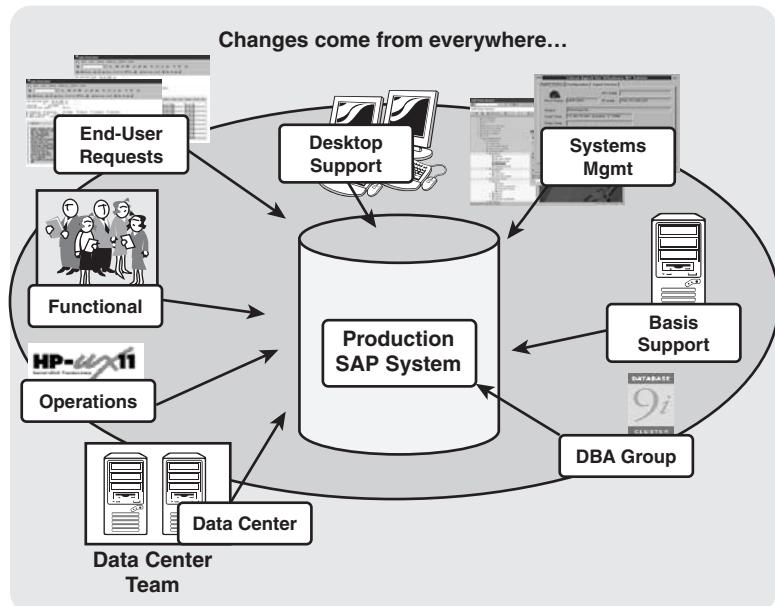


FIGURE 13.1 Changes come from everywhere, not just from within the SAP business-process realm.

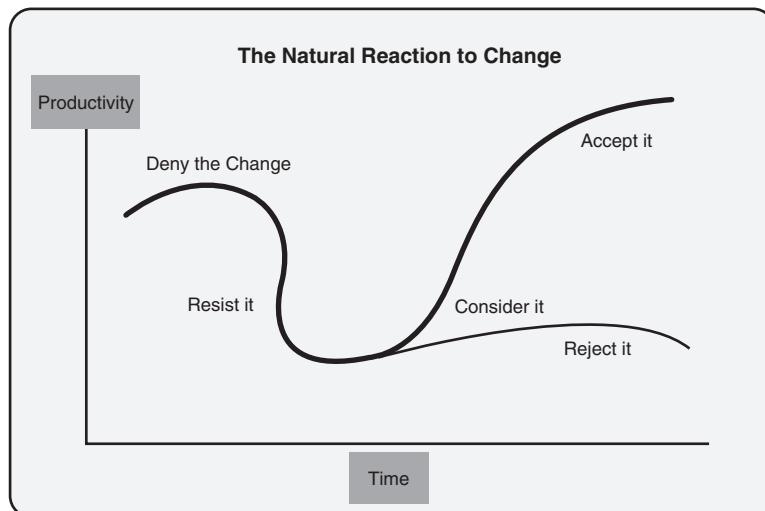


FIGURE 13.2 Our natural reaction to change must be addressed before we can ever hope to improve productivity or increase availability of our SAP environment.

This natural reaction must be addressed as much as anything else when organizational change takes place.

Change is inevitable. But even fewer people than we think actually enjoy change and can embrace and adapt to it quickly. Fortunately, the reward for organizations that handle change effectively is compelling. Effectively managing change is *focused on SAP end users* and other *stakeholders*, resulting in a more stable system, a better-performing system, a highly effective system, and so on.

In addition, the mission of change control often amounts to the following goals:

- Providing solutions to users' system problems (ushering in improved business processes), which further tends to improve system utilization or value
- Improving system testing
- Maintaining documentation
- Maintaining training levels
- Improving knowledge management
- Implementing improved operational processes and procedures

Each of these points is discussed in detail as we step through this chapter.

The Real Reason for Managing Change—Stakeholders

Stakeholders in an SAP implementation include anyone with a vested interest in the success of the project. As we discussed in Chapter 2, gaining their buy-in is essential. However, the degree in which stakeholders are impacted by the project vary, and therefore their commitment level varies as well. A common way of breaking these commitment levels down is by ownership, identity, participation, agreement, and awareness.

To best address organizational change like that driven by an SAP implementation, consider the following guidelines and best practices when it comes to designing or developing a change management organization:

- Stay focused on the needs of the stakeholders. This should drive everything from timelines to priorities, the structure of the organization, and how processes are ultimately deployed.
- Leverage an experienced change management consultant to review your environment and help design your change management organization.
- Get as much feedback as practical from stakeholders, including what they think needs to be done with regard to developing an organization focused on managing change.

- Delegate decisions to end users whenever possible. With their decisions driving the project in many ways, buy-in is practically a given, and priorities tend to be naturally maintained.
- Ensure that the project plan allows enough time to formulate and implement a change management organization.
- Ensure that the stakeholders understand the impact of your mySAP technical designs. To that end, I am reminded of a customer who moved from a single SAP production instance to a federation of systems connected via ALE. The impact of this on the business, in that all data wasn't available on all systems, had to be understood and accepted by the business.
- Finally, rather than trying to control change, instead take the position that it is more effective in the long run to understand and manage it.

If you follow these guidelines, not only will it become apparent who the primary people impacted by the project are, but it will allow for preliminary stakeholder profiles to be developed for the SAP project. And each stakeholder's commitment levels will make themselves known, too. Finally, general change management and communication models can be created with each specific stakeholder/audience in mind.

Change Management Best Practices and Approaches

In this section, I will discuss a process commonly used to implement change, and identify what I consider to be best-in-class change management practices.

Although the labels may differ, organizations possessing a firm grasp on managing change have adopted a process similar to the following:

- Create a project team dedicated to change
- Communicate the vision and goals of this team
- Acquire project team resources
- Monitor people/organizational issues
- Determine change management processes and practices to be leveraged, including tool sets
- Reorganize to support these processes and practices
- Implement these processes and practices
- Monitor completion of the project plan's tasks and milestones
- Communicate feedback, revise processes and practices as required, and continue to monitor

When it comes to managing change, topics like procedures, processes, guidelines, managing to a plan, tool sets, and so on naturally come to mind. Some guidebooks to change suggest an even simpler path to implementing change, focusing on project plan execution, communication plan execution, and stakeholder management. I have assembled a more representative list, though, that I believe reflects best-in-class change management practices as observed in my experience. These practices are illustrated in Figure 13.3, and include the following:

- Testing
- Documentation
- Standards
- Release strategy
- Clear communications
- Workflow
- Tool sets
- Feedback loop

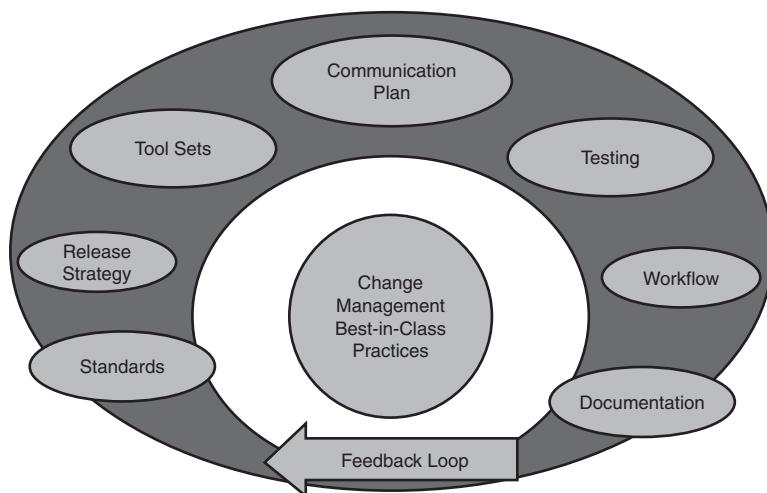


FIGURE 13.3 Best-in-class change practices all work together to help ensure that changes are managed comprehensively and effectively.

Each of these best-in-class practices to change management is discussed next.

The Core Philosophy Behind Change Control—Testing

Prior to any change being implemented in a system, it must be tested. This is true regardless of the enormity, location, or relevance of the system. In other words, if the system is truly important or otherwise mission-critical to its end users, it follows that anything that potentially affects the *availability* of this system must be thoroughly tested first. Sound testing has the following requirements or characteristics:

- A highly available and stable *test environment* is key.
- The staff is sufficient to handle a large load of new/changed test projects or “test cases.”
- To encourage repeatable and rapid testing capabilities, an automated testing tool or tools should be leveraged.
- Test cases should mirror process design, including exception handling, error corrections, reversals, and so on. By their nature, therefore, well-understood business processes often make the best test cases.
- Testing must fully cover and verify all SAP integration points, and all possible input data.
- Test cases should be able to run “standalone.” That is, there should be no dependencies on external data or other test cases, unless testing these dependencies is the goal of the testing.
- Testing must help an organization determine the impact that change has on other areas in the SAP implementation or within the company (that is, business process changes).
- To be most effective, the individuals tasked with building test cases should be very familiar with the functional or technology area in which they are testing. This includes access to formal training as required.
- In a new implementation, the *testing phase* and related timelines should be broken out from your master project plan, to allow for the granular level of detail required but not typically found or warranted in the master project plan (in other words, a high-level project plan like the SIPP included on the Planning CD is not appropriate for planning and tracking granular-level test case execution).
- The end users should determine what constitutes successful business-process output, based on what they also deem as acceptable input.
- Early and earnest involvement on the part of functional design teams helps them to understand the high priority of testing and test cases.

- Test cases are refined and otherwise updated when gaps are discovered.
- Changes resulting from testing should be reflected in documentation, from end-user-based to test-case documentation leveraged by the SAP TSO and more.

If *testing* reflects the relative importance of a system, it's safe to say that *documentation* reflects how seriously this importance is taken—the more critical a system, the better the documentation package should be that supports it. Why? Because documentation directly impacts how *effectively* the system can be used, managed, and supported. The central role that documentation plays in change management is covered next.

How Documentation Impacts Change Management

Before we begin, it must be understood that the term "*documentation*" in the context of change management means many things. To the mySAP solution's end users, the word "*documentation*" applies primarily to functional test cases. Each test case must be documented such that the case is very repeatable, and easily modified as the underlying SAP system evolves with each SAP Support Package and new functional enhancements. The key here is maintaining the data entry points, like which particular company codes apply to which materials, which storage locations apply to which plants, and so on. All of this is typically documented in a documentation "package."

To the folks tasked with training, "*documentation*" relates to everything a person new to a job task or position needs to know to become an effective mySAP.com component end user. Thus, documenting the process or work flow associated with each test case is paramount. And these trainers also need to understand how to maintain this documentation, the most effective ways of laying it out and presenting or delivering it (workshops, formal training, Internet-based, and so on), and any delta training required to be delivered between change management waves or releases (discussed later).

Documented workflow, process, and system training are all relevant and critical. This is because regardless of the goal of a specific set of documentation, it quickly becomes an integral part of supporting a business unit's standard operating procedures and processes.

Up to this point, I have only looked at documentation from an end-user perspective. To the SAP Technical Support Organization, though, the term "*documentation*" refers to the various tools and approaches used to manage change. This also includes documenting processes, like the "*promote to production*" process, or anything regarding timelines (that is, the amount of time a change stays and is tested in the technical sandbox before being promoted to the next system, and then the next, and

so on). Finally, documentation to the IS professional means frequently redocumenting the “current state” of the entire SAP Solution Stack, and updating how-to documentation as appropriate, such that all changes are easily identified and tracked throughout the life of the mySAP.com component’s system landscape.

Minimizing Change Management with Standards

For most companies, maintaining less of a variety of IT technology—whether this be a certain model of server, or type of disk drive, or version of software package—will cost less in the long run and prove easier to manage than maintaining a mix of products that might better fit each variance and niche-requirement within a particular SAP system landscape. For example, at one of my SAP customer sites, the client selected Microsoft Windows 2000 Advanced Server as their standard OS, even though Windows 2000 Server would have been adequate in quite a few instances, not to mention cheaper. Similarly, they standardized on a single model of database and application server, even though it provided additional processing headroom in some instances, and therefore cost a bit more up front than other models—in the long run, we all believed that the total cost of ownership would prove to be lower because this client had *fewer alternatives to deal with*. That is, with fewer types and kinds of hardware and software to run through the change control process every time a new firmware update or Service Pack became available, stability of the SAP landscape would be better preserved.

This example represents technology areas where standardization served to minimize the change management activities required to support the SAP landscape. Additional reasons to standardize include the following:

- Fewer hardware spares need to be maintained (less costly than maintaining one or more spare hardware components for each component deployed in production).
- You can take advantage of bulk buying or quantity discounts (where “20 of these” is less expensive to acquire than “7 of these” and “6 of those” and “2 of that” and “5 of those new ones”).
- Less training is required for the SAP support staff (no requirement to spend budget money training the operations staff in supporting different variations of the OS, for example, or different disk subsystem platforms, or database releases).
- The staff becomes very familiar and comfortable supporting fewer hardware platforms/components (no requirement to spend budget money training staff in supporting multiple server models, for example), and support calls are addressed faster.

- You will enjoy less unplanned downtime, because components are interchangeable (less risk of having to wait for a hard-to-find or out-of-stock part in the event of a critical server component failure), and less component variety equates to fewer changes that must be initiated, tested, and ultimately promoted to production.

The Release Strategy Approach to Making Changes

One of the absolute best practices that should be embraced by every SAP shop is use of the *release strategy*—industry leaders running SAP have employed it for years. As opposed to implementing changes one at a time in an unorganized and less controllable manner, the release strategy seeks to bundle changes into a “release,” as you see in Figure 13.4. The release is tested as a unit of changes, and often implemented on a monthly or quarterly basis. Sometimes referred to as a “change wave” or simply a “wave,” releases facilitate better planning, scheduling, staffing, and testing, in addition to better controlling costs (that is, costs associated with planned downtime, after-hours hardware/software support, and economies of scale when it comes to unit testing). Although a release may consist of many changes at once (all tested in concert with one another), best practices dictate that you “layer” changes into your SAP environment, keeping most solution stack layers in a consistent state. For instance, changing firmware during one release, upgrading mySAP.com functionality in the next, and upgrading the OS during the following “wave” minimizes later support issues never uncovered during the promote-to-production testing process. The idea then is to generally focus a particular release on making large changes in only one layer at a time (when possible, of course), with smaller accompanying changes in other layers as necessary.

A release’s contents—the changes to be promoted eventually into production—should be clearly documented on a file share, Web site, or through another means accessible to everyone on the project team, the Change Management Support team, and all stakeholders. A release may consist of the following elements:

- A predefined group of business-process-oriented changes and enhancements approved by the business
- Cross-application components needed to provide integrated solutions in SAP
- Support packages, SAP Kernel upgrades, and other SAP updates needed to maintain a well-performing SAP system
- Incremental SAP Basis upgrades (that is, 4.6C to 6.20 planned release upgrades, for example)
- Modifications to SAP extracts and interfaces

- Updates or patches to the database and operating system layers of the SAP Solution Stack
- Firmware and other hardware updates required of the SAP Solution Stack hardware layers

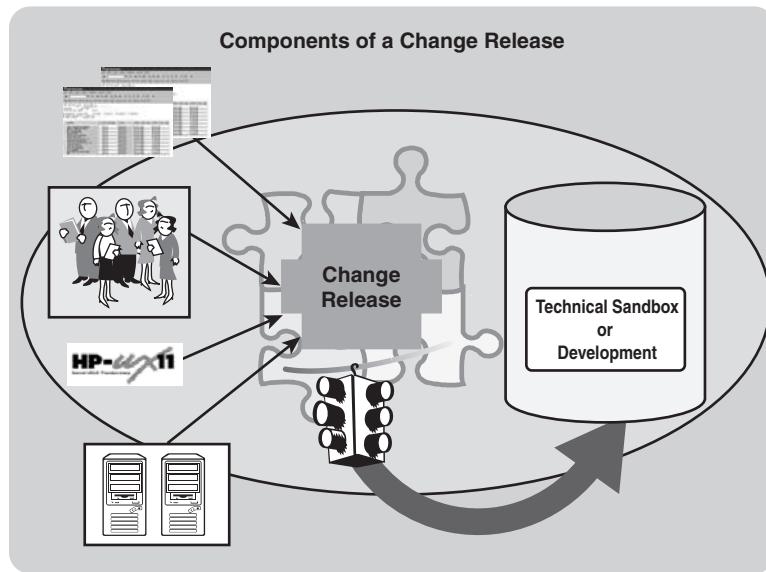


FIGURE 13.4 A change release strategy bundles multiple discrete changes into a change “package” or “release,” often also referred to as a “change wave.”

Regardless of the contents of the release, remember that all releases must be promoted “up” the SAP landscape leveraging the change management process. No exceptions! In other words, as you can clearly see in Figure 13.5, no changes must ever be put straight into production without some kind of prior testing in another system within that particular landscape.

For SAP implementations with multiple productive instances, a similar though more involved strategy is deployed. A “global” development system is recommended, and through a well-managed client strategy, this single development environment services multiple Test/QA, Staging, and production environments. As you see in Figure 13.6, such an approach allows for master data to be managed in one system, while also supporting the diverse needs of different geographies, for example.

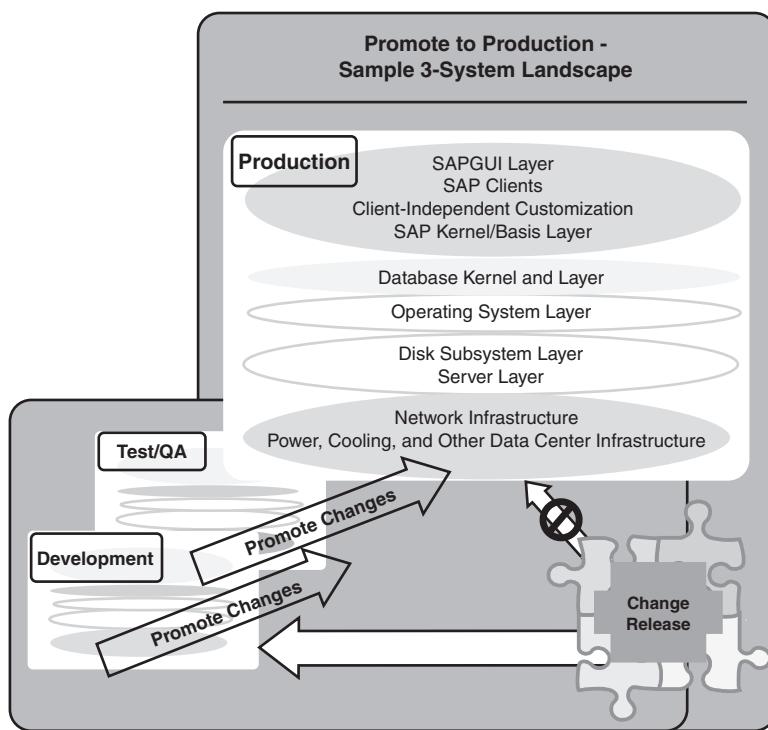


FIGURE 13.5 Promoting changes in any other manner than “up” the SAP landscape violates basic change management principles.

Exceptions to exclusively using a full-blown formal release strategy exist, however. For example, critical SAP updates needed to address “bugs and bombs” are often implemented in *Emergency Change Releases*. Similarly, critical changes may be put in quickly if data corruption or comparable risks are present. Such changes often include last-minute OS security patches, disk drive and disk controller firmware updates designed to prevent recently discovered data integrity issues, and so on. But these changes should still never be put directly into production; rather, they should instead be *expedited* through the change management process.

Finally, before we move on, it should be noted that a sound release strategy does not coincide with quarter-end or year-end processing. For example, if a company embraces a closing schedule based on the calendar year, the end of March, June, September, and December need to be “off limits” to change control waves. Similarly, month-end changes should be avoided as well. Instead, for my calendar-year customers I endorse the practice of promoting changes into production in mid-February, mid-May, mid-August, and mid-November.

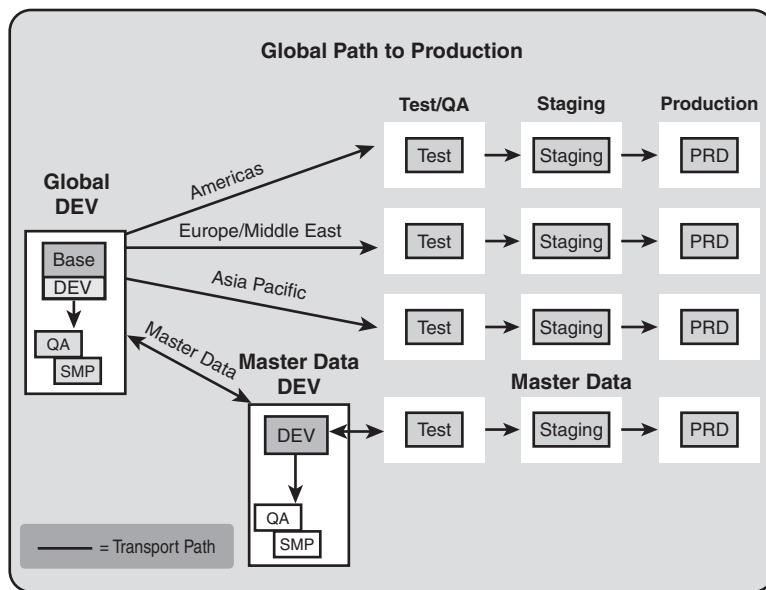


FIGURE 13.6 For SAP implementations that support different geographies, an approach like this “Global Path to Production” allows a single development instance and its associated master data to support multiple productive instances.

Clear Two-Way Communications

The difference between implementing changes correctly and implementing them otherwise can boil down to sound communication. According to the *American Heritage Dictionary, 2nd ed.* (Boston: Houghton Mifflin, 1982), communication is defined as “the exchange of ideas, messages, or information, as by speech, signals or writing” and represents another best practice for implementing change. Thus, as the organization tasked with managing change, the change management team must

- Understand that communications can always improve, and therefore strive for continuous improvement
- Communicate through a standard forum, and in several media if possible, based on the audience
- Embrace feedback, so as to improve communication quality and quantity (feedback rates its own section, and is discussed later in this chapter)

To facilitate the preceding goals while practicing continuous improvement, a *communication plan* should be put in place. The communication plan seeks to reach all

stakeholders, soliciting both input and feedback. It identifies the various stakeholder audiences, embraces different and effective communications media, and continuously looks for ways to improve.

Key challenges to creating and maintaining a good communication plan include differing physical locations, assumptions, knowledge, expectations, effort, time, perceptions, barriers between IT departments, barriers between business organizations and IT, and a lack of feedback/two-way communications. To minimize these differences, the following toolsets and approaches are often used:

- Monday meetings between the business and IT/SAP TSO
- Regular weekly SAP TSO staff/follow-up meetings
- “ERP Team” or other core mySAP Team meetings
- Published meeting minutes
- Dialog between management and business/IT groups
- Project plan schedules with clearly identified tasks, milestones, and responsible parties
- Business-unit representation within and between the various IT teams that make up the SAP TSO
- Change management Web site or intranet site
- Email between and within the various groups, including the use of standard email distribution lists
- Shared information/knowledge management repository, for example, a collaboration site hosted by SharePoint Portal Server, or a Lotus Notes database, or even a simple file share for starters
- Various “feedback” approaches (covered later)
- Department-wide team-building exercises and other non-work-related activities where valuable informal communication can take place

Although the preceding list represents a nice variety of potential communication methods, the next section illustrates what many of my SAP clients actually do in their real worlds of end users, customers, and deadlines.

Communicating Changes in the Real World

Communication tools and approaches like the following are used regularly by some of my customers. Some of these approaches are quite effective, but others are debatable. Judge for yourself:

- Effective: Leveraging a comprehensive change control application—Quite a few SAP customers look to the capabilities found in a number of different change management applications or utilities to manage, track, and report against their projects and processes. I know of one case where a Web-based application was written specifically for the customer. In most cases, though, I subscribe to the “buy it, don’t build it philosophy”—the same philosophy that drives many SAP implementations in the first place.
- Effective: Weekly change control status meetings, and project plan updates—in this case, the Change Control manager discusses current projects, pending projects, and resource issues/constraints with the entire team each Friday morning. These projects tend to drive most of the changes implemented in their release strategy, from hardware and OS upgrades to database migrations and SAP business enhancements. Updates to a master Change Control Project Plan provide for a single repository for all notes, updates, resource tracking, and so on.
- Effective: Weekly change management status reports linked to a Web site—Like the SAP customer described in the preceding paragraph, another customer also meets regularly to discuss pending changes. However, each Monday morning, the change management team lead disseminates a status update to a standard email distribution list composed of key business and IT folks. The current and pending changes are prioritized by functional area or technology stack. The unique thing about this particular team lead’s approach is his use of a company intranet site to host all archived updates, status reports, and other commentary regarding each planned change or test project. Scheduling, deadlines, and resource requirements and commitments are highlighted here as well. The site also serves as a repository for change processes, a vehicle for feedback (another link generates an email back to the team lead), and more.
- Questionable: One of my customers utilizes a very large dry-ink “whiteboard” mounted in the conference room where the change manager holds his meetings. There, in black and white, a running status update is maintained for each change wave—detailed changes, problems, and the person responsible for “fixing” the problems are all listed in a matrix format on the board. Of course, this approach is limited in that future status is difficult to communicate, the board tends to get messy after a change wave has been listed a month or more, you have to physically walk into the room to see status updates, and the risk of losing everything is possible if an enterprising member of the janitorial staff takes it upon himself to clean up the board.
- Questionable: Another customer of mine takes the use of sticky notes to a new level when it comes to managing hardware and OS-based changes. Here, the data center’s operations manager also plays the role of SAP liaison to the business (in conjunction with the SAP Basis manager, who also owns the database

layer). Planned changes, and the dates scheduled for the change, are noted and communicated via sticky notes and other labels that are physically placed on the server or disk subsystem that will undergo the change. As a change is promoted, the sticky note is moved to the next server or set of gears in the SAP landscape. In this way, a sticky note follows the change from development through production. I am by no means an advocate of this approach, but it would be unfair to say that it doesn't completely work. On the contrary, it seems to work quite well for this operations manager. The approach in general is dangerous, though, in that we all understand that sticky notes are susceptible to "falling off" over time. And from a status perspective, it seems unlikely that such an approach facilitates keeping others in the loop. Besides, there's only so much space on a note—updates or changes to the original plan must make for some really hard-to-read sticky notes.

- Questionable: Another limited approach to hardware/OS change control involves the use of a binder or log, where future changes and planned upgrades are tracked. The originator of the change, the name of the wave (if any) that the change may be a larger part of, and the implementer of the change are tracked here as well. Although this approach is more "publicly accessible" than the sticky note approach, I still think that tracking and mapping changes back to the master project/change control plan is difficult at best. And I see a lot of opportunity for missed deadlines and similar lack-of-communication issues.
- In the future? One of my forward-thinking colleagues envisions a day when devices and applications will be smart enough to track and communicate their own status within the larger scope of change control releases. For example, an entity scheduled to undergo a change would "know" it by virtue of a connection to a master project plan. When queried physically or via a Web browser, the entity would respond with a clear status communiqué, like "My processing microcode is scheduled to be updated from 2.41 to 3.10 on Monday. For schedule or implementation questions, please contact Dave Future. For technical questions, please contact Jan McKay. And have a *great GlobalXYZ day*." All of this capability would be made possible through intelligence built into each entity, enabling it to "absorb" its Pert Chart bubble and information regarding relevant tasks, responsible parties, and so on. Finally, the actual upgrade would amount to simply pushing a physical or virtual "make it so" button, complete with feedback like "Are you sure, Dave? You might want to rethink this move, as I have been scheduled to run the monthly closing on Monday." You saw it here first.

My recommendation is, of course, to employ all three of the first real-world communication plans described in the preceding list, as most of my other customers tend to do in one form or another. Some of the other approaches might have value, too, but only if backed up by "robust" communications vehicles like the first three.

Another Way to Implement Change—Workflow

As in testing, documentation, minimizing variables in the SAP landscape, and so on, the concept of workflow is another excellent change management facilitator. One of my SAP customers tells me they enjoy an exceptionally smooth change management process by employing the following “workflow-based” approach to implementing new changes. For this example, let’s say that the impending major change is with regard to adding the HR functional module to the company’s current SAP R/3 implementation. An initial budget has been approved, and executive/senior management sponsorship is already in place:

1. The customer/end-user organization meets with the change team on a weekly basis. The team consists of the manager of the group (the change manager, or a designate), a few key technical SAP TSO resources (based on the nature of the upcoming projects or change waves), appropriate functional representatives, and the supervisor of computer operations.
2. Based on the change to be implemented, a basic *work breakdown structure* (WBS) or simple “scope of work” is drafted during this meeting (for larger-scope projects, like adding HR, people are expected to come to the meetings with their initial work breakdown structures, rather than consuming valuable time to do so in the meeting).
3. Success criteria are identified.
4. A priority level for the particular change is established.
5. Test hardware, software, infrastructure resources, end-user liaisons, functional liaisons, and technical liaisons (collectively termed “resources”) are identified during this initial meeting.
6. End-user, functional, and technical team leaders are sought, and usually named at this time. For big changes, a project manager is named as well. The meeting ends, and each Team Leader takes with them action items with regard to completing their piece of the WBS, fleshing out timelines, identifying constraints, and so on.
7. In another meeting attended by all team leaders and the project manager and/or change manager, the tasks and milestones for inclusion into the *master project schedule* (leveraging what this customer calls a *change checklist*) are identified. A preliminary timeline is drafted by each Team Leader, reflecting the needs of the area in which they are responsible.
8. Constraints are reviewed (holiday/vacation schedules, availability of critical resources, and so on), and the meeting ends.
9. The change checklists are merged and reviewed at another meeting to ensure that the workflow inherent to each individual area still “works” and “flows.”

10. The merged and now revised master change checklist is formally reviewed and introduced into the master project schedule some time in the next few days. The change manager owns the master project schedule.
11. The project manager or change manager coordinates with the technical lead, SAP Basis team, and any solution stack partners to ensure that hardware resizing is addressed. With regard to this HR example, the database server will eventually be taxed more, the application server layer will need to grow, security issues inherent to maintaining HR data will need to be addressed, accessibility to the system via Web Application Server or ITS might need to be considered, and so on.
12. Meanwhile, the technical lead begins to work with the computer operations supervisor to secure technical sandbox or development resources, reserve testing time, and so on, in preparation for initially testing the change.

My customer uses this workflow approach to track changes in progress, too. Note that the weekly change management meeting ranges in time from an hour (status update meetings) to perhaps three or four hours (for major functional or infrastructure changes). This HR scenario would constitute a half-day meeting, for example.

Like the best of plans, our plans sometimes change, though. If changes to the master project schedule are required, the following activities need to occur:

- Establishing new timelines
- Reviewing resources again
- Identifying and reviewing constraints again

I'm told that changes to the master project schedule mirror typical IT project plan changes, including slips due to unforeseen technical complexities, lack of key resources, scope creep, budget issues, and lack of ability to appease everyone's vision of how the change needs to be tested or implemented. Special considerations and circumstances have taken their toll on past projects, too, usually related to other testing or change control projects taking precedence over the change being discussed.

Change Control Tool Sets and Approaches

When it comes to managing change, we are not alone. Long before SAP projects consumed so many IT and end-user organizations, we all somehow managed to implement other enterprise-wide projects. Tools and approaches like the following have long been accepted by IT project management professionals:

- Embracing formal project planning methodologies and training
- Management meetings, held to gain high-level management or team leader consensus
- Team meetings, held to obtain status updates and ensure the team was making progress
- Communication planning, and the toolkits that facilitate this
- Change management training—processes, approaches, best practices, and so on
- Training focused on problem-solving techniques
- Training focused on assessment/analysis techniques
- Business case management approaches, and training supporting this
- Leveraging subject matter experts, via internal and external consulting resources
- Change management software applications

With regard to this last item, change management software applications, a number of these abound specifically for mySAP.com. I seem to run into a particular company more than any other—Kintana. Their “Kintana Accelerator for mySAP.com” allows for end-to-end change management, while also providing visibility into potential problem areas (like resource constraints, timeline inconsistencies, and so on).

A good change management tool will also hide unnecessary complexity (again, to increase visibility into problem areas), and automate much of the work of managing change. I like Kintana’s product because it is not limited to managing a specific SAP Solution Stack—various UNIX flavors, Windows 2000/NT, even mainframe database servers are all supported. Finally, because Kintana has products for other enterprise applications like PeopleSoft, Oracle, and Siebel, an investment in Kintana can really pay off across a mixed enterprise landscape. Read more about Kintana by visiting their Web site, or leveraging the “LINK - Kintana Accelerator for mySAP.com” document on the Planning CD.

I suggest that the capabilities and characteristics outlined in the following list serve as an evaluation guide for comparing the various change management toolsets available today. That is, a change management tool should accomplish the following:

- Provide a consistent and repeatable process for managing change across the SAP system landscape
- Safeguard the production SAP system from improperly executed or unauthorized changes

- Provide audit-trail capabilities of previous changes to the system
- Provide the SAP TSO with the information needed to monitor change waves, track priorities and resources, and so on.

Such an application minimizes unplanned downtime by reducing risks associated with making changes. And it also shrinks the amount of time people need to spend in change control meetings!

When it comes to managing the thousands of table settings and other options within an SAP client, SAP change management will always present a challenge to the organization managing it. By its very nature, the flexibility that end-user organizations love about mySAP.com becomes a driving factor in terms of the amount of time that is devoted to change management. Good change management tools will minimize the time by automating repetitive tasks related to transports, recompiles, system refreshes, migrations, and similar tasks. Thus, any software tool or utility that can ease some of this burden should be welcomed and fully embraced by the SAP technical support organization.

Feedback—Improving Change Management Incrementally

When it comes to improving any process, it is usually helpful afterwards to ask and try to answer the same questions that drove creating or updating the process in the first place. This constitutes a feedback loop when end users and other stakeholders are answering the questions for us. With regard to the change management process and organization, then, the following questions are appropriate:

- How successful was the change? Were the success criteria drafted at the first change meeting on target or appropriate?
- How long after the change did it (or will it) take to stabilize the production system?
- Overall, how disruptive was the change to the business? And what can we learn from this, to further minimize the impact of changes in the future?
- What individual and organizational benefits of the change have been communicated to our stakeholders?
- What have end users given up or sacrificed as a result of the change (additional downtime, loss of other functionality)?
- What skills and resources were actually required to implement the change? That is, how can we better plan for and execute similar changes in the future?
- How well did the change team/project team understand and execute their roles and responsibilities during the change process? Where can we improve?

We have now concluded our discussion on change management best practices and approaches. In the next section, I take a broader look at exactly how change impacts not only every layer of the SAP Solution Stack, but every phase of an SAP implementation as well.

Change Control Affects Everything

As I said earlier, changes to SAP can affect everything—the solution stack, system landscape, phases in the project, and so on. The next few pages detail exactly what the effects are, and ways to manage these rather than being managed *by* them.

Change Control and the SAP System Landscape

One of the most basic tenets of sound change management or change control includes leveraging the SAP system landscape to test changes to the environment. One method often used is a *Technical Sandbox Change Management Checklist*. Such a checklist details exactly how changes are initially introduced to the landscape, and then how they are managed or “promoted up” the landscape, starting with the technical sandbox. A sample checklist can be found on the Planning CD.

As you have already read, sound change management practices are essential for maintaining a highly available and well-performing SAP deployment. Such a checklist approach to testing change helps ensure that nothing is missed and that all steps are actually completed successfully. Doing otherwise risks the success of the project and the integrity of the overall solution, as it impacts the entire end-user community, technical support organization, and more.

Change Control and the Phases of SAP Implementation

To understand the role that change management plays in an enterprise mySAP solution, it is important to understand the evolution of that solution in terms of phases of systems development—these phases tie nicely into the “SAP system landscape” previously discussed.

The solution phases described in the following list are based on a typical implementation of an enterprise SAP deployment, the beginning-to-end timeline of which consumes six months at best to perhaps two years or more:

1. Pilot Phase—This phase allows a prospective SAP customer to examine, test, evaluate, and explore the proposed mySAP solution. Here, it is “proven” that indeed SAP will solve the business problems at hand. It is also important in terms of ensuring that accurate training, development, and complete deployment plans can begin to be understood, and the enterprise project effectively estimated from a cost and time perspective. This phase may last from six to eight weeks to perhaps many months. A pilot phase for each mySAP.com

component or area is common, as well. Note that in all cases beyond R/3, a “core” transactional system must be in place. Finally, a preliminary hardware sizing is beneficial at this point, to ensure that the pilot solution is configured adequately from the beginning, so as to best set the stage for the next phase.

2. Development Phase—Building upon the solution stack prepared in support of the Pilot Phase, this phase consists of customer and/or consulting personnel configuring and customizing the system for use in the target business areas. This phase occupies much of the overall project plan, and typically continues throughout the life of a solution (though perhaps less intensely on initial business areas, to focus on new business areas and mySAP.com components). Initial changes like maintenance upgrades and bug fixes are often performed during this phase as well, hopefully in a technical sandbox or similar system. Thus, within a few months of this phase, you tend to see a fairly stable environment for continued development, and a good foundation for moving into the next phase. During the development phase, it is also important to begin planning for the expected configuration testing and preproductive deployments to take place next—changes in business assumptions, implementation plans, and a company’s particular solution stack technology roadmap must be visited.
3. Training Phase—Like the development Phase, this phase also begins when the Pilot Phase ends. Training of the development team (many of these folks are already experienced SAP consultants) and the SAP Technical Support Organization occurs first. Training of end-user and other personnel begins when a preliminary level of configuration and/or customization has been completed. The real work of end-user training commences a few months prior to Go-Live, however, to help keep their knowledge fresh. After Go-Live, training continues to some extent for new users of the system. Therefore, like the development Phase, this phase is also ongoing in some capacity throughout the life of the solution.
4. Testing or Quality-Assurance Phase—Sometimes referred to as the “preproduction” phase, the Test/QA phase begins when the first promotion of code from development to test occurs. This phase is entirely devoted to configuration, integration, and quality assurance testing. Some time a few months before Go-Live, stress-testing should also take place. Note that the final production hardware sizing also takes place during this phase, typically completed as early as required to address hardware procurement and configuration/testing lead times. This phase does not end as long as the development phase still exists. From a change management perspective, the Test/QA phase therefore represents the culmination of individual and packaged change testing—all of the individual changes that make up a change release or wave are tested here as a single package, which will ultimately be promoted to production.

5. Disaster Recovery Phase—Often called the DR phase, this is the phase where disaster tolerance and contingency plans are identified. For the most mission-critical of systems, fully redundant data center facilities are staffed and trained. A cost-effective way I have seen this implemented is by locating an organization's test or staging system in a different physical location from the production system. Then, assuming the system is sized appropriately and a disaster-recovery process is in place to allow for failover, the organization is well on its way to working through the remainder of the DR phase, including process testing, documentation, failover and fail-back testing, and so on.
6. Production or Production Roll-out Phase—This phase commences the day of "Go-Live" as the company and its organizations and business processes become dependent on the data being hosted or managed by SAP. Best practices strongly suggest not going "live" with multiple SAP components at once, unless they are tied together. In other words, there is usually little business reason (and much more to risk from a holistic perspective) to go live with both R/3 and APO on the same day, for example. On the contrary, though, it might make a lot of sense to go live with both R/3 and Enterprise Portal (or Workplace), as these products work hand-in-glove with each other.

Timelines typically overlap, as you see here in Figure 13.7. Throughout each of these phases, the impact upon the SAP Solution Stack is significant, too. The impact is so significant, and in such a fundamental way, that I have devoted the entire next section to this.

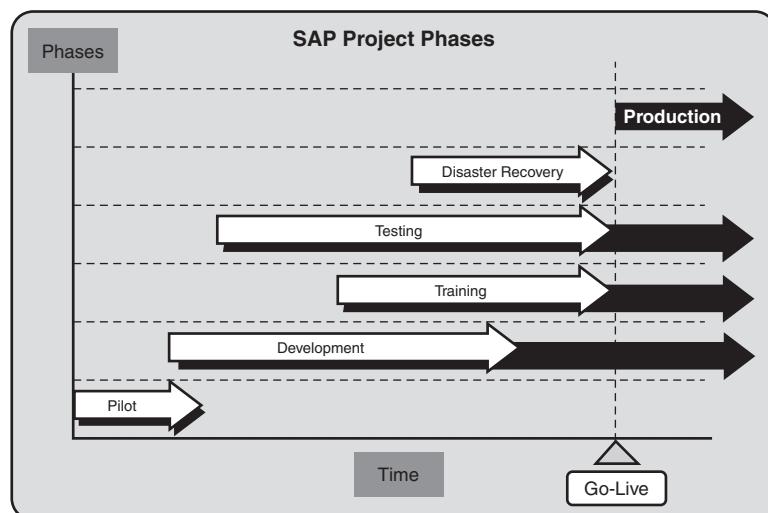


FIGURE 13.7 Note the overlap in phases—these timelines rarely support a finish-to-start approach to managing phases.

Change Control and the SAP Solution Stack

A lot of work goes into configuring and assembling an SAP solution that is supported by its Solution Stack vendors and effective in servicing its end users. When such a supported configuration is in place, the goal of the SAP Technical Support Organization should be to minimize technical changes to the stack unless absolutely required. The reasons cited most often for making technology stack changes are

- Performance reasons, especially with regard to whatever the current bottleneck tends to be (most often the disk subsystem or a factor related to CPU utilization, like growth in system usage or volume of transactions)
- Stability reasons, like published issues with specific software drivers, database releases, firmware revisions, and so on
- New functionality reasons, like the need for additional application servers or more horsepower in the existing servers when incremental SAP components, modules, and users are added to the landscape.

With these three reasons in mind, I'll next touch upon a few sample scenarios and how each impacts the solution stack.

The Hardware, OS, and Database Layers

Hardware and software vendors typically embrace a rigorous testing process of their own prior to releasing new products. On top of this process, vendors with SAP Competency Centers take the testing to a new SAP-specific level, and add what I term an *SAP Filter* to the testing process. The SAP Filter serves to filter out variables in the SAP Solution Stack. Thus, instead of trying to support or certify every single hardware platform, controller, software update, firmware release, OS patch, and so on for SAP, the filtering process seeks to test and support a specific combination of the stack's components, and "filter out" others.

The SAP Filter is therefore good for the SAP technology vendor, good for SAP AG (which finds itself supporting fewer combinations of technology platform variables) and good for the SAP customer. The only drawback might seem to be a final selection of products too limited for a particular customer. In reality, though, I have never seen this to be the case—vendors are naturally motivated to test/support their latest products or most compelling solution sets. Consider the following list of tested products and releases embraced by one of the author's favorite SAP competency centers, and judge for yourself:

- New releases of hardware platforms and major components, including servers, disk subsystems, and disk controllers

NOTE

As of August 2001, disk host bus adapters, or HBAs, are no longer specifically required by SAP AG to be tested; many SAP hardware vendors still seem to perform this level of customer-assurance testing anyway.

- Updated firmware releases that address severe stability, performance, or security issues
- New Operating System releases specific to previously tested and approved hardware platforms
- Operating System service packs and patches that will not likely be replaced in the next six months, or that address severe stability, performance, or security issues
- Hardware-specific OS drivers and updates, though typically less often than service packs/patches
- New database releases (typically just major releases only, or as specifically requested and funded by database partners or customers)
- Database service packs/patches that will probably not be replaced in the next six months, or that address severe stability, performance, or security issues
- New major releases of mySAP.com components, historically centered on new SAP Basis releases of R/3, BW, APO, EBP/SRM, WP, CRM, and other components as they are released
- Support Packages for each major SAP Basis release (that is 4.5B, 4.6C, 6.20, and so on. Other less common releases (like 4.6B and 6.10) may also be tested, but usually not without internal or otherwise-sponsored funding)

With such a comprehensive list as this one, it's easy to see why not every SAP Solution Stack option needs to be tested—indeed, much of the stack *is* tested, along with the most compelling product sets within each layer of the stack.

To perform this comprehensive testing, Solution Stack partners and vendors should embrace a new-product testing process similar to that described in the following list. These are sometimes referred to as *standard test plans* for SAP, and are often employed by large SAP customers as well as solution stack vendors. A valid test plan should be consistent with SAP's general change-management recommendations (and their own internal practices), consisting of a process something like the following:

1. Perform a “clean” install of the product to be tested, or install a clean solution stack foundation if an upgrade to a particular product is to be tested (the latter is a much more common testing scenario). For this example, we're focused on a new release of R/3 with a new release of SQL Server 2000.

2. Perform the SQL Server and R/3 installation, noting any issues and resolutions.
3. Log in to R/3 via the SAPGUI that ships with the particular release being tested.
4. Perform an R/3 data dictionary check (DDIC).
5. Perform an R/3 Client Copy.
6. Perform an R/3 license check.
7. If applicable, import data into or out of the system (more applicable to SAP BW, APO and other components that rely on a core transactional system).
8. Customize existing load scripts (using AutoIT, CATT, or a more robust tool like AutoTester ONE or eCATT), or run a “mini” standard SAP benchmark against the system, to simply generate a load. Allow this to run for 12–24 hours (whatever is consistent with your previous testing), serving as a “burn-in” for any other new components in the solution stack. Record the transaction load, average response times observed, and issues.
9. If funded, run a complete SAP Standard Benchmark, as appropriate.
10. If on the roadmap or otherwise appropriate, use this system to perform delta testing, including database upgrades, hardware upgrades, and so on.
11. Publish the testing results internally or as requested.

Such comprehensive testing by a hardware/software vendor assures the vendor’s customer base that the platform in question is truly ready for prime time, and it also allows the partner to say without a doubt that a particular solution stack has been fully tested and is therefore supported.

The preceding example reflected what I call *general new-product testing*, where a change impacts all or nearly all layers in the solution stack. If we view the SAP solution like a set of concentric circles, general testing in support of change management is typically performed from the database server out to the other components in the solution, like the Central Instance and Application Servers, and then out to the ITS Agate and Wgate servers (or Web Application Server, as applicable), and finally out to the SAPGUI or WebGUI front-end client. As shown in Figure 13.8, then, general testing tends to be the most comprehensive form of testing performed by hardware and software partners. It represents testing of an end-to-end solution, “inside-out.”

Other testing tends to be more tactical in nature. This is referred to as tactical, characterization, or delta testing, and addresses a smaller piece of the circle. It also involves less testing in general, often a subset of the standard test plan we looked at earlier. The goal of delta testing is to prove more that a new solution component *works*, rather than exactly how *well* it works. And in the process, we generally uncover and learn from the problems with installation, integration issues, and other issues as they crop up.

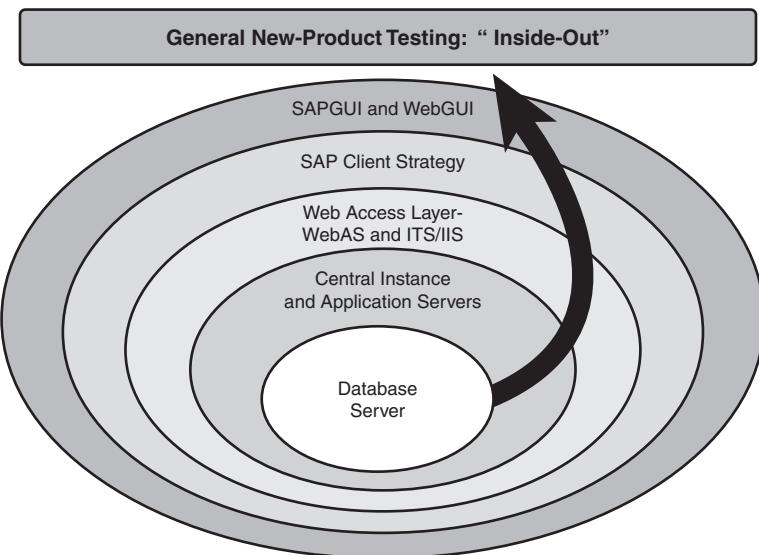


FIGURE 13.8 General new-product testing is typically performed against an end-to-end solution, as you see here.

After each test is completed, the system log files/event log, upgrade/installation logs, and other pertinent logs are reviewed for errors, and the errors are analyzed. Further testing may then be warranted, sometimes uniquely crafted to isolate or highlight a particular layer or component in the solution stack.

After all testing and error-analysis, the new product is either *approved* or *failed* (pass or fail, go or no-go). Most testing results are shared with the appropriate internal and external organizations, including product engineering, various software support groups, the OS vendor or organization, the database vendor, SAP AG, and so on through either formal certification processes or informal competency center communications.

SAP Application Layer—Transport Strategies and More

Transport strategies represent a critical piece of the overall change management strategy. For it is by this process that business processes are created and changed, support packages are applied, and other changes to the core SAP internals are performed. Historically, SAP provided something called the *Correction and Transport System* (CTS) to perform the work of transports. More recently, they renamed it to *Change and Transport System*, and introduced the much-improved *Transport Management System* to work with Transport Organizer, Workbench Organizer, and Customizing Organizer. Together with the tp and r3trans transport tools, the new system forms a comprehensive albeit application-focused change management utility.

The transport strategy or process implemented by an SAP customer should address or facilitate

- Transport request forms, which should be consistent and complete
- A process for reviewing the forms before and after the fact, to ensure that the change actually accomplishes its goal
- A process for approving changes (that is, change management meetings and review boards)
- A method for ensuring that the transported code indeed passed through the approval process
- Attention to expedited emergency transports, or those changes that must find their way into production as soon as possible

As I touched upon earlier in the “Change Control Tool Sets and Approaches” section of this chapter, a process that is automated to the fullest extent possible is highly recommended. Change management tends to be tedious and therefore error-prone—a change control tool or utility is an inexpensive way to mitigate this obvious (and otherwise quite commonplace) risk.

Upgrades—Realizing the Benefits of Change Control

With an SAP upgrade comes the opportunity to really benefit from good change control practices. The system should already be completely documented and all business cases thoroughly tested. Therefore, as the first pilot upgrades are tested in various sandbox or crash-and-burn environments, the go/no-go decision becomes easier—the test cases either work, in which case the upgrade was successful, or fail, in which case any number of issues might need to be addressed. Even if the test case fails (which is likely), good documentation will make the point of failure more clear than otherwise, and tend to shorten the time-to-resolution significantly. Often, the failure is actually the result of a change in how SAPGUI display screens are laid out, the names or location of specific data entry fields, and so on, rather than a failure in the business process itself. In other words, many perceived failures in the initial tests of an upgrade are likely to be errors in the test mechanism itself and not actually the underlying upgrade. By quickly resolving the “what you expected was not what you got” GUI-related display issues, the validity of the business process can be quickly confirmed. At that point, you can then check off another task in your upgrade checklist as complete, and continue moving forward with the upgrade.

Finally, because the upgrade process forces you to update nearly all of your test cases to some extent, you can take this opportunity to learn from test-case shortcomings or no-longer-applicable requirements, and look at some of the following possibilities as you reconstruct your test cases:

- Evaluate a new testing tool or approach, one that might be less expensive, or easier to use, or offer more benefits like automated testing and improved regression-testing capabilities, and so on.
- With your new test cases, you might better integrate or address the needs of the documentation and training teams, as well as the teams tasked with stress testing or load testing.
- Take the opportunity to educate new or less experienced team members in the process of building, testing, and maintaining test cases.
- Improve upon the test cases themselves. For example, one customer of mine rewrote most of their test scripts to reflect end-to-end business processes, such that all data needed in one script was created from a previous one. In this way, they no longer needed to worry about “stocking” a test client prior to executing test cases. And in another example, my customer took the opportunity to create more variables (instead of relying on many hard-coded input parameters) so that alternative test case scenarios could be easily created and tested.

With managing changes to the SAP Solution Stack behind us, a discussion of structuring the organization actually tasked with managing change is in order.

How to Organize and Plan for Change in the Real World

To stay linked with the entities that drive or create changes, you must integrate a “change management” organization into your SAP processes and day-to-day life supporting SAP. Doing so also promotes the need for maintaining, testing, documenting, and otherwise managing change throughout the company. To accomplish this, I recommend constructing an SAP support team focused on creating the best end-user experience achievable. Such an organization should have timely or “always on” access to an online change management system; authority on behalf, and cooperation of, the end-user community; access to the SAP TSO and other resources as required, and more, as you will see.

In my experience, there is no single best answer when it comes to building a change management organization, much less simply naming it. Placing them within the larger structure of the SAP Technical Support Organization or another group is also subject to a variety of things. As is evident from the following list, the teams tasked with managing change can be found across many different organizations, both in and out of traditional technology-focused and business-focused organizations. Some of the “homes” that SAP change management organizations find themselves a part of include

- Obvious organizations, like Change Support, Change Management Team, Change Control Group, and so on

- An organization focused on change within a business-management context, including Business Applications Services, Business and Change Integration, Organizational Effectiveness/BPR Management and Business Services, or Business Systems
- Groups tasked with knowledge management, like SAP Knowledge Management or the Knowledge Transfer Team
- A group within the general IT organization, like Information Resource Management team, Information Technology, Technology Integration Group, or the plain ol' IT Department
- Part of a larger functional area or business-oriented team, like Finance & Accounting, Global Supply Chain, or SAP Cost Management and Controlling, Operational Accounting, and Financial Information Support Team
- Part of a training group, including MIS Global Learning Solutions, Training Services, Learning Center, IT Training and Documentation, Change Management & Training, ERP Training/Information Technology, Training and Process Group, or Technology Education
- A group affiliated with customer service or help desk organizations, like Client Services, Information Support, SAP User Support, or simply SAP Support
- A member of the general SAP Technical Support Organization, with perhaps a title that reflects "Change Management Analyst" or something similar

Given this, however, an organization like the one illustrated in Figure 13.9 is deployed most often in my experience.

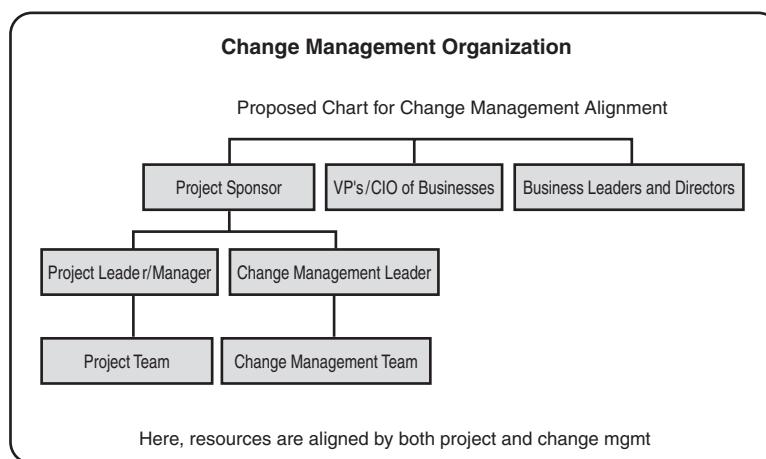


FIGURE 13.9 The change management and project management activities work hand-in-hand in this typical "Change Management" organization.

Change Management Review Board

Although the change management team might be focused on managing and implementing changes, a change oversight committee or *Change Management Review Board* tends to play a key strategic role in larger or more complex SAP implementations, staying connected with the project well after Go-Live. The Review Board ultimately serves as the gatekeeper for the release or change wave strategy, refining and enhancing the process as the business or relevant IT organizations require. In this role, the Review Board actively communicates and coordinates functionally, cross-functionally, and across regions or geographies—their scope is as large as the circle of project stakeholders. Traditional Review Board responsibilities include

- Driving standardization of common, global change management processes, and pushing these down into the functional and SAP TSO organizations
- Identifying activities and practices that must exist to effectively achieve the benefits expected from each change/enhancement to the SAP Solution Stack
- Coordinating and prioritizing changes/tasks to achieve improved productivity from the SAP system's end users, including pushing organizations in getting timely decisions made
- Measuring improvements in SAP by identifying and publishing *Key Performance Indicators* (KPIs), provided from each of the business organizations

The Review Board also often acts as the facilitator for coordinating and communicating resource allocation needs, like training materials in support of changes, updated workflow/process-flows and similar documentation, access to testing resources, and access to super users or power users identified as such within their functional areas.

Manager of Change Management

The Manager of Change Management (a generic title) is a key member of the Change Management Review Board, tasked with building/maintaining the change management team as well as maintaining and refining the change processes adopted by the Review Board. Because of this, the CM Manager must possess not only excellent communication and facilitation skills, but also exceptional business acumen. With these core competencies, the CM Manager is equipped to address a number of key responsibilities during the initial stages of the SAP implementation, including

- Securing executive sponsorship in support of a formal change management organization
- Assessing the critical implementation challenges specific to the project
- Developing or embracing/refining an existing change management organization

- Communicating the real or perceived impact that changes will have in different organizations to all stakeholders, and then working to either expedite resolutions or mitigate risks

CAUTION

Don't be tempted to outsource the "CM Manager" position to a third-party contractor or consultant house. Such a role is best served by a long-time employee of the company, intimately familiar with the informal and formal power structures within the organization.

An ability to align business operations and work processes with new technology is key as well, as is an ability to negotiate with the various stakeholder and agents of change in the company. And it must be noted that the biggest opportunity for failure, according to more than a few of my SAP customers, is not gaining the executive sponsorship needed up front. Without this, the Change Management Manager has less perceived authority, and therefore less ability to enforce change control processes vital to the stability of the system.

Senior SAP Change Management Analyst

This senior position is critical to fill as early in the deployment as possible, as the best candidates will play a significant role in supporting changes even as soon as the data center is completed and the development system is put in place. I have seen anywhere from one to eight change management analysts included in my clients' overall SAP implementation project plans. Two to four seems to be average. Key skills and experience in the following areas are of the utmost concern:

- Change management experience with SAP systems or other large ERP/enterprise solution environments
- Organization and change strategy consulting or related hands-on experience
- Familiarity with the specific releases and versions of SAP to be deployed
- Both outstanding written and oral communication and presentation skills
- Ability to work with all levels of the organization, from executives to "single contributor" employees and contractors
- Ability to keep a small team focused on working through changes that may prove difficult to implement, including maintaining a high level of resolve on the part of the entire team

Additionally, because the senior change management analyst will probably help the change manager develop the SAP change management team over time, as well as

address day-to-day and other tactical issues, the following qualifications are also important:

- Change analysis experience in support of implementing and managing SAP enhancements (functional change releases or waves)
- Experience with automated testing, and building test case models
- Leading and educating the business through readiness activities
- Ability to draft and maintain clearly written documentation
- Training logistics
- Change metrics (how to measure success)

Other typical activities might include participating in the design or redesign of the team, documenting the various business process design sessions or the technology update projects in which he participates, and so on. Assistance with developing and deploying training curriculum is a common use of this member's time, too, as is helping to continually refine the SAP communication plan. Finally, the senior change management analyst should also be focused on how to plan for, manage, and implement the emergency changes that crop up periodically.

Change Management Lessons Learned in the Real World

Change management lessons learned abound. I have spoken of various lessons throughout this book that ultimately point back to poor change control practices. If the value of managing changes is still questionable in your mind, continue reading for a few more pages:

- My colleague and I received a frantic phone call early one morning from a customer whose "highly available" SAP/Oracle cluster was down. Apparently, they had performed a firmware upgrade on their production disk subsystem *without testing this process in another environment*, and the cluster would not start afterwards. After some time, we found that the upgrade process was indeed not performed correctly (actually, it was only half completed), and we worked with our customer to complete the remainder of the process as quickly as possible. Unfortunately, this bad situation was made worse by a harried Basis administrator—we were not called until quite a few totally unrelated changes were made elsewhere in the system, in the name of "cluster troubleshooting." Changes to the cluster itself, and then to the database layer, only served to exacerbate an already tense issue. In the end, we counted our customer lucky that they only lost half a day of uptime. If they had just followed sound change management practices and first performed this change in a sandbox or test system, this "simple" firmware upgrade would have been completed well within their maintenance window, and proved itself uneventful.

- As I discussed previously in the section titled “Communicating Changes in the Real World,” the use of an intranet Web site can really facilitate communicating change. However, at more than one customer site, my colleagues and I have been the victim of limited access. That is, as short-term contractors engaged in testing to support a major change control wave, we were told in no uncertain terms at one customer site that everything we needed to acquaint ourselves with the project was communicated via a project plan hosted on an internal company Web site. Communication outside of the realm of the project plan was limited—meetings were infrequent, emails were nonexistent, and much of the rest of the staff was located in a different geographical part of the city than the data center. We were expected to understand our role, timelines, and resource/scheduling needs strictly by virtue of the project plan. Unfortunately, access to the Web site was not granted for weeks, and the project manager was not inclined to share the plan via email or otherwise. This forced us, and certainly other third-party contractors, to stumble about in the dark until a user ID and access to the site were finally granted. My points here are simple—leverage alternative means of communication, and be sensitive to access issues that might impede project progress.
- One global SAP implementation project manager was confident that he had put together an adequate “change management” process, integrating it into how they planned and deployed IT solutions for business problems across the enterprise. In reality, though, a number of issues spoke to major change management shortcomings. For example, changes made directly to production resources without the benefit of previous testing resulted in many hours of unplanned downtime. Additionally, a lack of a technical sandbox or even a Test/QA system had forced all changes to be implemented in their development environment. The lesson here—a two-system landscape approach to implementing SAP is not the right answer for most SAP projects; a three-system landscape should be considered as a minimum requirement.
- One of my customers preferred their paper-based change control process to alternatively establishing a software-based process replete with full reporting capabilities. Each Friday, a laughable moment in time occurred as the change control manager plopped down her printed stack of completed and outstanding change requests on the conference room table, and began to go through them one by one. Sure, the process worked. But I question how well-spent the change control manager’s time was as she printed everything, and later keyed in updates scribbled during the weekly meeting on each printed page into her master plan.
- A very large government entity running SAP R/3 was cited by the inspector general for having “less than adequate” SAP change control procedures in place. The violation? Absolutely all of the SAP support organization—both

contractors and employees, developers and basis personnel alike—had the ability to make changes directly to the production instance. A major no-no!

- I have witnessed many SAP projects learning early-on the value of a sandbox or technical testing/staging environment that *exactly replicates production*. Keeping a technical sandbox allows for experimenting with and verifying a multitude of changes prior to promoting the changes into production. And in this case, it would have allowed the SAP Basis team the opportunity to familiarize themselves with something new to them—using Microsoft Cluster Services to cluster SAP R/3. Because they had nowhere else to practice and hone their skills, it's no wonder that they inadvertently broke their production cluster when they went to fail over a node for maintenance. They "evicted" the node rather than failing over the node's resources to the other node, and then made some scheduled changes. Unfortunately, when they tried to reintroduce the first node back into the cluster, they discovered the error of their ways and proceeded to spend the better part of a night learning more than they ever bargained for about clustering SAP R/3 with SQL Server.
- With regard to training, one size does not fit all. Some users, like power users or super users, need to learn *all* relevant business process options and alternatives, not just the most common options. For these users, "foundation training" or "Intro to SAP" courses are inadequate. As one of my customers found, if you don't train your users, they'll train themselves on the production system. And everyone will suffer the performance penalties as 20 or 30 users simultaneously search every row in a million-row table, pulling up every sales order opened or closed in the last six months, including everything you pulled in from your legacy system being replaced by SAP.
- A comprehensive change to master data resulted in lost development time for one of my customers mere weeks before Go-Live. The exuberant and sleep-deprived developer failed to test her mass update process in a different environment or client, and effectively wiped out half a day's worth of her own and her colleagues' work. She was quite lucky, actually; it could have been much worse.

Notice the mention of a *technical sandbox* several times in this list. The sandbox is a "best-kept secret" of many SAP shops. Not only does it allow for the SAP TSO to become familiar with SAP operations and day-to-day management without interfering with "productive" development systems, it plays a key role in the promote-to-production change management process. Let's face it—development is actually a "production" system in its own right from day one. Just try asking a group of 30 ABAP developers if it's OK to reboot the development box so that a potential change can be quickly tested, and the value of a technical sandbox grows exponentially as the expensive developers sit on their hands waiting for the system to be returned to them.

Other common real-world issues and questions related to change management include the following:

- It's expensive to conduct onsite training workshops—where can computer-based or online training be used most effectively?
- Configuration guidelines and best practices provide a foundation for building and maintaining an SAP solution. Who should be responsible for really doing the work of keeping all of this documentation up to date, though?
- The “big picture” needs to be communicated to all of our stakeholders; they need to understand the amount of change that is typically tested and supported for each change release.
- How should actual business productivity be measured, to support the value behind a particular change wave?
- The term “super user” is overused and a bit trite for today’s vernacular. What constitutes a real super user? It used to be synonymous with the person in each functional area who was the “go-to” person for all of the other users. The super user worked with the help desk to resolve problems, and in doing so he made sure that these problems and their resolutions were shared throughout the end-user organization (see the Planning CD for a sample “SAP Help Desk Top 20 Issues and Resolutions” document).
- How should the change management process employed by the SAP team mimic or deviate from the company-wide approach to managing change, or should they be one and the same?
- Feedback must be a natural and easy extension to change control.

With the real-world experiences and lessons-learned described in the preceding list, I thought it would make sense to capture a list of “worst practices” as well, identified next.

Change Control “Worst Practices”

Although all of the best practices and best-in-class approaches we have discussed about managing change are valuable, change management is also all about avoiding common pitfalls, too. That is, another goal of change management processes is to avoid repeating history and making common mistakes, or engaging in poor practices that may compromise the integrity of your SAP deployment or ongoing operations. More than a few SAP implementations have learned this the hard way over the years, however. I suggest that you avoid the *change management worst practices* outlined in the following list:

1. Making firmware, hardware, OS, driver, or application software changes—including updates/upgrades or applying service packs/patches—without *first* testing these changes in the technical sandbox environment, and then for an appropriate period of time (ranging from one business day to more than a calendar month, depending on the scope of the change), and lastly, getting sign-off from the responsible vendor (typically the vendor's organization for supporting SAP) is worst practice #1.
2. Failing to adhere to the flow or “promotion” of changes or data from the technical sandbox (where infrastructure changes are first tested) to development (where data originates) to Test/QA (if it exists) to training (if it exists) to the final production solution is worst practice #2. Doing so compromises the integrity of the production platform, and therefore negates the presence of many of the other SAP landscape environments. The ability to maintain the history of data objects and configuration changes as they are moved through the SAP landscape is critical to maintaining a highly available and reliable solution.
3. Initiating configuration changes anywhere other than in development is asking for trouble, and represents worst practice #3. Further, objects should *never* be modified outside of the original development system—*do not* perform these “development” tasks elsewhere. Changes made directly in production are the worst-case scenario, of course, as this implies a complete lack of adequate and appropriate testing. But changes made directly in Test/QA, or Training, and so on, are nearly as bad because now the change is not being driven from development and *version control* becomes an issue.
4. Running additional enterprise applications on the same hardware platform responsible for supporting SAP is worst practice #4. Even if the system has been specifically sized and configured for multiple applications, I tend to shy away from this practice. Why? Additional applications complicate the solution stack, make capacity planning calculations difficult, and make performance guarantees and service level agreements much more difficult to uphold. This is why I recommend that other business applications including email, database, file/print, audio/streaming video applications, and even domain infrastructure reside on completely separate hardware platforms.
5. Worst practice #5 dictates that you never run network-intensive applications within the private networks dedicated for the SAP system landscape. Like #4, doing so affects performance guarantees. To ensure a better level of consistency in terms of end-user response times, the network requirements of your mySAP.com component should not be compromised by requiring it to share network bandwidth with email/messaging, other database products, groupware, and so on, unless this need is explicitly required and carefully “sized.”

Tools and Techniques

Quite a few templates and tools can be found on the Planning CD. These include checklists for initiating, implementing, and promoting changes throughout the system landscape, a change management communication template, and a form to be used for implementing emergency change requests. I have also included a “matrix” document detailing how to manage a technical sandbox environment, a pre-change interview questionnaire, a post-change management survey, and a sample “Master Transport List” along with a number of miscellaneous how-to and otherwise informative documents. Finally, as usual I have included all the figures found in this chapter, in Microsoft PowerPoint format.

Summary

In this chapter, I have defined change management, discussed how to implement changes consistent with best practices, and looked at how changes impact everything from the project phases, to the SAP system landscape, to the individual solution stacks residing within each landscape. I then drilled down into organizational complexities, including a common approach to structuring a change management team. Throughout all of this, I hope that the underlying theme of conservatism has shown through in a big way. Changes are an inevitable part of an SAP solution during and after deployment, inviting both much-needed new or revised functionality as well as the possibility for disaster into the organization. My recommendation with regard to change is to play it safe, and practice conservative and consistent change management as a matter of course. Doing so makes the lives of everyone much better—your end users, the general IT support team, your functional/development teams, your colleagues in the SAP technical support organization, and so on.

In the next chapter, we’ll look at another key group that truly benefits from effective change management, the SAP TSO members tasked with supporting your SAP project from an operations and systems management perspective—the systems management, SAP Help Desk, and SAP computer operations teams.

14

SAP Systems and Operations Management

Back to SAP Operations

As I discussed in Chapter 10 when we planned for and initially implemented SAP Data Center operations, it is important to set the groundwork for the SAP computer operations and systems management teams, or simply SAP Operations. Normally, this groundwork consists of SAP CCMS and CMS (Central Monitoring System, also referred to as "CEN"), manual operations processes, simple paper-based checklists, and a collection of essential utilities and tools ranging from the robust and very detailed SAP solution manager for operations to simple command-line utilities. Even if you are implementing the latest SAP NetWeaver technology or are simply managing an older R/3 3x solution, most of this groundwork, in my experience, varies little.

In the case of SAP shops with expertise in an enterprise management application like CA Unicenter or HP OpenView, this groundwork may actually already exist and be quite robust, too. Normally, though, the real work of deploying an end-to-end solution approach for managing your entire mySAP system landscape remains to be addressed; this chapter walks you through that process. I begin by making a case for the SAP operations manual and the role of documentation in general, move into piloting and selecting from a variety of enterprise systems management applications, and wrap up with a discussion of various tools and utilities focused at different layers of the solution stack.

IN THIS CHAPTER

- Back to SAP Operations
- What Is the SAP Operations Manual?
- Systems Management Techniques for SAP
- Piloting a Systems Management Application for SAP
- Evaluating Enterprise Systems Management Applications
- Additional SAP Management Tools and Approaches
- Tools and Techniques

NOTE

This chapter stops short of covering operations-related tasks outside the scope of monitoring and fundamentally managing the SAP environment. The minutiae related to going live are instead detailed in Chapter 17, “Preparing for SAP Go-Live.”

Although I focus in this chapter on preparing the foundation for going live, it must be noted that all management approaches, methods, tools, and so on are immediately applicable to other systems within your SAP system landscape, regardless of where you find yourself in your implementation or upgrade project plan. For example, your SAP Development and Test environments are already in use by now. And a technical sandbox, training system, and other components of a system landscape may also be in place. These important systems are absolutely critical to their respective users—expensive SAP developers and essential power users engaged in testing the integration of new SAP-driven business processes. So, although the thrust of this chapter is aimed squarely at crafting an operations and systems management approach, it is not necessarily aimed exclusively at production SAP environments.

What Is the SAP Operations Manual?

The collection of “as is” or “current state” system documentation, day-to-day and other regularly scheduled operations tasks, various installation and operations checklists, how-to process documents, and more constitutes the *SAP operations manual*. A good operations manual therefore supports more than basic computer operations tasks; it facilitates fall-back in the case of a change management wave gone awry, and it serves as a blueprint should the need to rebuild all or a piece of the SAP system landscape arise. Finally, many of the documentation-specific components of the Disaster Recovery Crash Kit discussed in Chapter 6 are also fashioned directly from the SAP operations manual, specifically:

- Detailed current state information as to how a particular system and its components are architected and configured (for example, server PCI card slot locations, physical disk subsystem disk layouts, OS file system details, and so on).
- Step-by-step documentation necessary to install the production system, all access methods, all operations management utilities/applications, and additional documented procedures explaining how to restore the database and other critical data.
- Operational documentation, like that which reflects regularly scheduled daily and weekly tasks, and more.

The SAP operations manual is obviously critical to ensuring a well-running mySAP solution, then, as illustrated in Figure 14.1. In the next section, I take a closer look at

what I consider to be the most important considerations when assembling an operations manual.

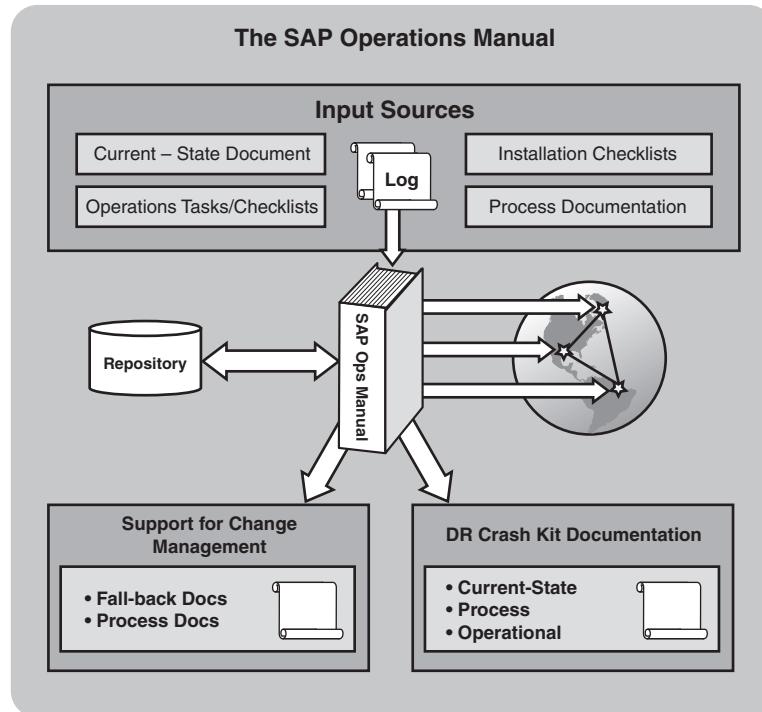


FIGURE 14.1 The SAP operations manual brings critical resources together, and feeds other functions as well.

Detailing Your “Current State”

Documentation that describes the specific configuration and layout of your unique SAP Solution Stack is often referred to as *current-state documentation*. Recording the details inherent to your particular hardware configuration, OS installation, database and mySAP application configuration, and so on provides critical data necessary to re-create the environment should it be necessary. This level of detail also minimizes complexity and time frames related to troubleshooting problems down the road—with details close at hand, minutes rather than hours are consumed when it comes time to share configuration data with others helping you to troubleshoot a problem.

In the past, I have spent quite a bit of time documenting complex SAP system landscapes for my own SAP customers. Most of the time, my directive has simply been to

get everything “down on paper,” using Microsoft Word and Excel. Later, this same documentation has often been fed into different presentation media and approaches, some of which I discuss later in this chapter. Figure 14.2 depicts a common approach I developed for easily collecting and segregating current-state documentation by solution stack layer. I call the resulting customer deliverable “the matrix” because it maps various solution stack components to their properties, in a matrix format. A simple Microsoft Excel workbook divided into solution stack layers by using multiple worksheets and descriptive tabs forms the foundation of this simple but effective approach. Details are either noted within each worksheet, or in a Microsoft Word or other document embedded into the worksheet. In this way, a single workbook becomes a master repository of current-state solution stack configuration data.

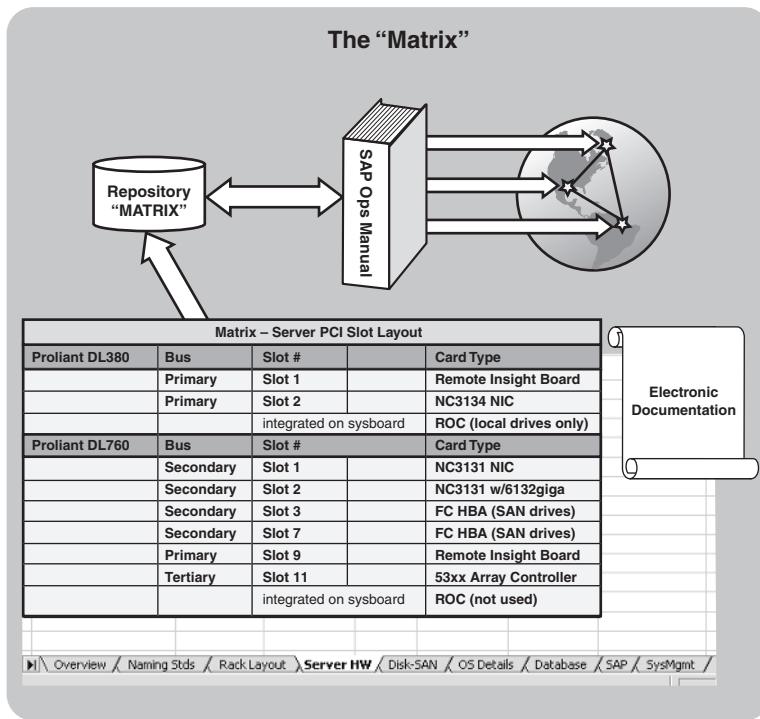


FIGURE 14.2 My approach to documenting an SAP system landscape often amounts to something that resembles this detailed Microsoft Excel-based matrix.

Refer to the Planning CD for an electronic version of a sample current-state solution stack matrix that you can use as a template for developing your own.

Documenting Daily Operations and Installation Procedures

With the current state documented, I then focus my attention on developing daily checklists and installation recipes. Checklists are useful in performing routine tasks, like those related to a set of daily procedures. Installation recipes, on the other hand, are one-time documents useful in installing and re-installing a particular hardware or software component. The idea behind a recipe should be clear—it not only provides a list of “ingredients,” but also provides the precise steps necessary to install a particular solution in a repeatable manner. And because you already have the “ingredients” documented via your own current-state documentation, a recipe can usually be crafted quickly.

Anyone serious about minimizing the time required to install something should invest the time up front to create a comprehensive recipe. Recipes reduce unplanned downtime (if, for example, a problem forces you to re-create a portion of your SAP Solution Stack). Recipes are also useful tools when it comes time to train your new-hires and others involved in installing and supporting a particular layer of the stack. This is why the best of the best—SAP customer sites, hardware vendors, mySAP consulting firms, and so on—make use of recipes and checklists, as you see in Figure 14.3.

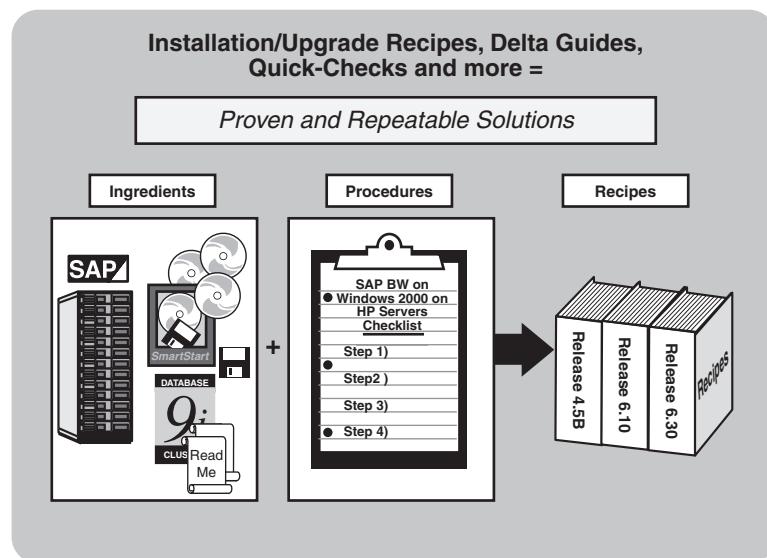


FIGURE 14.3 The right ingredients and proper checklists will create very repeatable solution-guide recipes.

Of course, recipes and checklists may vary in the level of detail that is provided. I have created many high-level mySAP installation recipes that are only four to five pages long. These *quick reference checklists*, or “Quick-Checks,” help experienced individuals stay on track and perform a procedure quickly. I have included a sample Quick-Check on the Planning CD.

At the other extreme, I’ve also assembled very detailed checklists, complete with screen shots (in the case of software installations, for example), including even very basic “how-to” procedures within the body of the larger document. These detailed checklists are designed for beginners and other novices unfamiliar with a particular process or procedure. For example, for some of my customers I have created SAP R/3 installation recipes for particular hardware/software combinations that exceed 200 pages but walk the installer through everything from hardware configuration to SAP R/3 and ITS software step-by-step installations.

One other approach has proven helpful to me in the past, too. Rather than re-creating the wheel and documenting much that is already documented by SAP, database vendors, and other solution stack technology vendors, I created an approach that I labeled “Delta Guides” and discussed briefly in Chapter 11. A Delta Guide is very much like a detailed checklist, with one exception—it references other documents published by their respective vendors, rather than reinventing the wheel and rewriting everything. In this way, a Delta Guide acts as an umbrella over perhaps 10 to 20 other documents, bringing all of these resources and procedures together in one repository. Using this method, not only can system installations be performed quickly, but the work involved with creating and maintaining the recipe itself can also be performed quickly. I’ve included a sample Delta Guide on the Planning CD, too, to provide a starting point for creating your own.

Documenting Other Regularly Scheduled Procedures

Like daily operations procedures, SAP and its solution stack partners also recommend that a set of specific procedures be executed weekly, monthly, quarterly, and even annually. These regularly scheduled operations tasks span the gamut, from facilities to server-, OS-, database-, and SAP-specific tasks, for example:

- Archiving tape backups
- Testing disaster recovery procedures
- Performing various security reviews
- Reviewing scheduled SAP housecleaning jobs
- Restocking data center consumables
- Executing test scripts to compare disk subsystem performance to historical performance

Detailed monthly, quarterly, and annual SAP Operations checklists can be found on the Planning CD.

- ▶ Detailed daily and weekly checklists and the value gained from these paper-based management approaches have been discussed previously, and electronic versions of these documents are available under the “Chapter 10” directory on the *Planning CD*; see “Planning for SAP Data Center Operations,” p. 376 in Chapter 10.

“How To” Documentation

When it comes to operational processes and current state documentation, it’s pretty safe to say that these critical components of the SAP operations manual are only made possible by “how to” or *process documentation*. Process documentation amounts to the documentation *behind* your documentation—it describes and details how to perform a particular task, or it references another source (like an Internet URL or other company-internal file share) where this is further explained. So, if for example, your SAP daily operations checklist calls for an SAP operator to execute transaction ST03 and collect performance statistics related to the number of dialog steps processed each hour over the last four hours, a process document should exist that walks the operator through gaining access to the particular system, executing transaction ST03, navigating through the various menu and screen selections, and ultimately pulling the requested data out of SAP’s Computer Center Management System (CCMS). For template purposes, I have included many useful examples of “how to” documentation on the Planning CD, including how to use the SAP Microsoft Management Console, how to use Microsoft’s PerfMon utility for managing your SAP system, how to use the SAPGUI, and more.

Documentation Best Practices in the Real World

In the course of consulting, I have embraced or otherwise come across a number of approaches that I believe represent documentation best practices. The following list details some of these approaches:

- Documentation Development—Although a fair amount of documentation is inevitably developed “on the fly” after Go-Live, or after changes are made to your system, arguably the *best* way to create documentation is by leveraging your technical sandbox or training system as part of your normal “promote to production” change management strategy. In this way, processes and procedures can be tested and recorded simultaneously, ensuring that accurate procedures are drafted or updated *before* they are required to be executed in the production system.
- Documentation Standards—To create a consistent and professional SAP operations manual requires a set of documentation guidelines, standards, and procedures. For instance, putting together a document that describes the process for

creating documentation, including formats and fonts to use, is a good starting point. Next, a working template to be used for developing new documents is a great time-saver.

- Screen shots—Precise documentation calls for detail and clarity. Sometimes, the best way to obtain this clarity is to use screen shots, so that complex options and tasks can be clearly displayed to ensure error-free processes.
- Documentation Publishing and Accessibility—I've seen all manner of sound publishing practices when it comes to operating procedures, including everything from standard word-processing-based methods (Microsoft Word and lesser packages), to true publishing packages (Quark, FrameMaker, PageMaker, Microsoft Publisher), to simple HTML-based Web sites (using something like Microsoft FrontPage or Word), to feature-rich portals (SAP's, Microsoft's, and Oracle's portal products, to be specific). Simply publishing documentation via the Web, however, seems to be the best way to go all around simply because of the ease of access provided in this way. That is, disaster recovery capabilities are strengthened, training capabilities are broadened, and escalation support is enhanced when documentation is available ubiquitously using a standard Web browser. This method improves greatly on the idea of maintaining dedicated documentation file shares or offline CD-ROM access, too, as has been popular in the past. And publishing from a Web server located at a site other than your SAP Data Center also enables you to access your documentation in the event of a disaster.
- Documentation Maintenance—Keeping your SAP operations manual and other documentation up to date requires an investment in time. I already mentioned taking advantage of your change management strategy to help keep documentation up to date. Other methods may be preferable to you, though. For example, one of my customers dedicates 8–16 hours each month to reviewing and updating their current-state documentation and standard operating procedures. They go so far as to actually schedule it into their monthly checklist. Another client of mine makes real-time updates to their hard copies, and then “catches up” every few weeks or months as necessary. Although this latter approach is not as consistent as the first, I like the fact that they pencil in updates and changes *as they occur*; this tends to help them maintain very accurate documentation. Finally, a third client of mine recently dumped all of their SAP and other enterprise application-specific documentation into Microsoft's SharePoint Portal Server. In this way, not only can they take advantage of true document management features (like check-in, check-out, role-based security, and so on), but they also have access to the documentation repository via a Web browser as well as through native Microsoft Word, Excel, and PowerPoint formats.

I believe in the use of a “Document Review Checklist,” too, and have included a sample on the Planning CD. This checklist helps to ensure that the documents you publish adhere to your writing and formatting standards and that they are reviewed in terms of technical accuracy, table of contents, general look and feel, and much more.

With the groundwork of how and what to document behind us, let’s now turn our attention to actual processes and products that facilitate SAP systems management.

Systems Management Techniques for SAP

As I discussed back in Chapter 10, the SAP Computer Operations team needs to be prepared to manage the various SAP system landscape components being implemented months before Go-Live occurs. Development systems, training systems, systems that support testing, and so on will all be installed and configured prior to production. And each of these systems will need to be managed to ensure that it is available to its users. This includes *proactively* monitoring hardware and software installed as part of each system, looking for impending failures or circumstances that may cause less than acceptable performance. The systems also must be monitored in a *reactive* manner; reactive management minimizes the amount of time necessary to respond to a failure, therefore reducing unplanned downtime.

These next few sections address techniques for SAP systems management. I cover a number of products that support monitoring and measuring the performance of different facets of your entire SAP Solution Stack, including solutions that take advantage of Java-based and HTML-based connectivity. To set the stage for the capabilities required of a set of management/monitoring tools, I pose the following requirements:

- You need to be able to measure and verify end-user response time for each and every end-user transaction. Similarly, you need to do the same for batch jobs and other background processes.
- You need to provide a way to tune mySAP, the database system, your OS, and your hardware platforms—the entire stack.
- You need a method of monitoring your network connectivity as well, including the ability to identify your specific response-time component attributable to the network.
- Your toolset(s) need to support not only monitoring but also diagnosing your observed performance within and between each layer in your unique SAP Solution Stack.

In my experience, there is unfortunately no single tool that can do all of this perfectly in a typical mySAP environment. As you will see, though, some of the tools available today come pretty close to providing a truly end-to-end SAP monitoring and management solution. And the “holes” left after using these tools can be filled effectively with additional utilities and tools, discussed later.

Leveraging CCMS for Manual Processes/Checklists

SAP's very own Computer Center Management System (CCMS), forms the foundation of any SAP enterprise management application or tool in use today. That is, all of the applications in the SAP enterprise management space today leverage the power of CCMS, or more specifically the *data* that CCMS gathers through a service or daemon called the *SAP OS Collector*. These enterprise applications simply present this data in unique or more valuable ways than raw CCMS can, hence much of their value.

A SAP OS Collector runs for each installed SAP instance, collecting data related to the performance and configuration of the particular server's operating system, database, and mySAP component. Most of this data is unfortunately very focused on the details of that particular SAP instance, however; cross-system performance, or the performance of an entire SAP system made up of multiple servers and instances, is much more difficult to rationalize using only CCMS, hence the popularity of other management applications and approaches provided by SAP AG as well as third parties.

I think it's important to understand not only what kind of data is available via CCMS, but how to access that data, before the real value of an expensive and potentially complex management system can be fully understood. For this reason, I'm a big fan of daily and weekly SAP operations checklists—though manual in nature and therefore more time-consuming than automated management approaches, operations checklists force a basic education in SAP CCMS.

Accessing CCMS is simple. For our convenience, SAP created countless shortcut transaction codes (T-codes) that exist solely to pull performance and availability data out of CCMS. Transaction codes like ST06 and DB02 provide us with operating system and database-specific data, respectively, for example. ST03 and ST04 give us information related to system-level performance. ST07 and AL08 provide us with user-based data, such as the number of users logged into a particular functional area, or executing specific functional transactions. These T-codes are not hugely intuitive, unfortunately. However, several observations are in order:

- T-codes that end in “01” (like MM01) are used to create something. Similarly, “02” is used to change or update, and “03” is used to display.
- The first two characters in a T-code reflect the functional area or technical area in which a T-code operates. “ST” provides system-related data, “DB” provides

database-related data, “RZ” addresses system profile and monitoring data, “OS” reflects operating system-related statistics, and so on.

- Most transaction codes are four characters long, like ST03 or AL08. The majority of other T-codes are five characters.
- Transactions like SM51 and OS07 can be used to change the application server to which you are connected or logged into, without logging out and logging back in. In this way, it becomes easier to more quickly collect statistics for multiple application servers within a particular system, for example (if you have to do so manually!).

You can also access CCMS through SAP’s basic menu system as well, navigating through scads of different transactions until you find the one that suits your purpose. Of course, this can be time-consuming, but in the same manner it can prove quite educational to new CCMS users, too.

Because some of the CCMS transaction codes can be quite system-intensive, or can be used to make unwanted changes to the configuration of a system, I suggest that a technical sandbox or similar system be made available to the SAP operations and help desk teams when it comes time to learn how to use CCMS. And as a basis for learning, I further suggest that the various SAP operations checklists I developed for this chapter and Chapter 10 (with their specific transaction codes) be leveraged as basic training guides.

Automating CCMS Data Collection Processes

Walking through the process of collecting statistics through SAP CCMS can be quite time-consuming, especially when multiple SAP system landscapes and large SAP instances are involved. Consider one of my favorite SAP customers, for example. Not only do they maintain a five-system R/3 landscape (with eight application servers in both staging and production), but they also operate a four-system environment for both SAP BW and CRM. Collecting and analyzing snapshots and historical statistics across three different production systems, not to mention the other supporting systems, could have easily become a full-time job for someone. The right answer for them, like most other organizations, was to ultimately evaluate and deploy an SAP enterprise systems management application. Many options exist in this regard—SAP Solution Manager, SAP CMS/CEN, HP OpenView, BMC Patrol, and a host of other alternatives—and are discussed later in this chapter.

Evaluating, selecting, licensing, deploying, and then really learning how to use such a potentially complex management application does not happen overnight, however. So in this particular client’s case, rather than waiting the three to six months to evaluate, install, and configure an enterprise management tool to automatically collect system-wide performance and availability metrics, they instead asked me to script some of the basic performance monitoring transactions (ST03, ST07, ST04, and more) in the interim. This scripting is what I am referring to when I

talk of “automating” CCMS data collection processes—scripting takes manual processes to the next level, so to speak, and enables these processes to be more rapidly and consistently executed.

For the most robust results, I recommend using an SAP API-aware scripting tool like AutoTester’s product AutoTester ONE. API-aware tools communicate directly with the SAPGUI or SAP application server to which the tool is connected, and because of this these tools enjoy privileges beyond traditional scripting utilities. For example, an API-aware tool can pull data that is otherwise only displayed through the SAPGUI, and dump this data into a text or spreadsheet file to be further manipulated. I call this “scraping” the screen, the equivalent of precisely extracting specific data out of a SAPGUI window. A great example is the detailed performance data that is displayed via transaction ST03 and illustrated in Figure 14.4.

Similar data is available via transaction ST03N, which is more “graphical” in nature, and allows easy access to cross-system global data. Note however, that ST03N lumps RFC and CPIC loads into the Average CPU time, making it a bit more difficult to determine true CPU utilization.

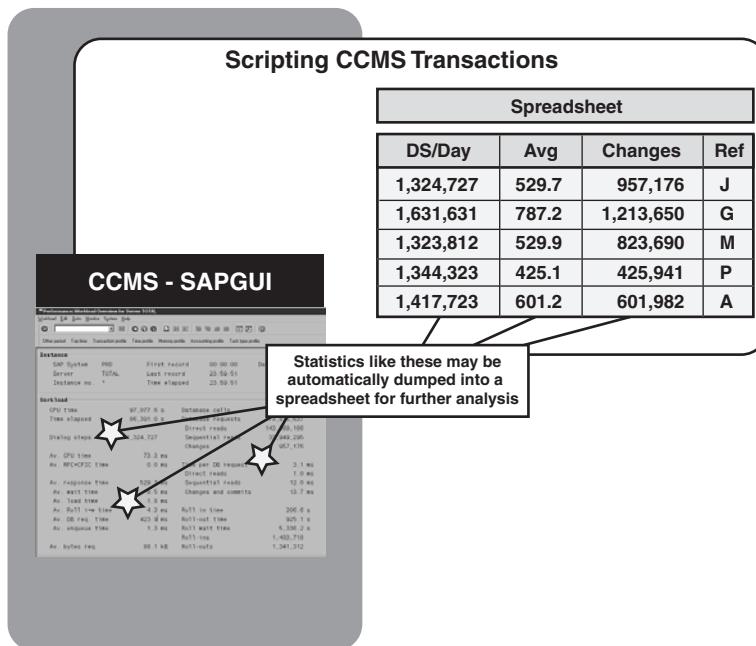


FIGURE 14.4 As a precursor to a full-featured enterprise management application, consider using an automated SAP API-aware scripting tool to rapidly “scrape” critical CCMS performance statistics and then dump this data into a spreadsheet.

An SAP API-aware tool can pull these displayed values out of the SAPGUI and dump them into an Excel spreadsheet, so that over time you can track things like average

wait times, roll times, DB request times, and so on for dialog response time, background task response time, update response time, and more.

Other scripting utilities, such as *Perl*, *Visual Test*, or *AutoIT*, do not enjoy screen-scraping capabilities, and can therefore never provide this level of value. Not surprisingly, these non-API-aware utilities make up for their lack of capabilities in terms of price, however—in fact, tools like *AutoIT* are offered as freeware.

I've included a few sample automated scripts on the Planning CD. Due to differences in versions of mySAP components, they will probably not work on your system as-is. But with a bit of tweaking (and the proper licensing when it comes to the *AutoTester ONE* scripts I included), you should be able to quickly take advantage of these simple automated approaches to leveraging CCMS. And you'll save an enormous amount of time if you are asked to monitor more than a few application and database servers but granted very little time and budget money to do so.

CCMS, Transactional Monitors, and CEN

As I've already mentioned, shortcomings in using CCMS to monitor complex SAP system landscapes have helped to greatly drive the development and subsequent popularity of SAP enterprise management tools. SAP enterprise management tools fill in the gaps left wide open by CCMS prior to the most recent SAP Basis releases, such as those identified here:

- Before release 4.6C, CCMS suffered from an inability to easily consolidate performance and other data across multiple instances within a single SAP system. For example, a production system with eight application servers, one database server, and a central instance required most CCMS transactions to be executed ten different times, once per server.
- Similarly, CCMS was less than easy to use when it came to reporting performance, availability, or other metrics across more than one server (a few exceptions existed, but this was generally true). Reports must be generated ten times, once for each server, and then somehow rationalized afterwards.
- CCMS by itself could therefore only weakly address any kind of SAP solution-wide data gathering or reporting. Each component had to be addressed separately.
- CCMS graphics are even today quite limited, therefore further impacting CCMS's ability to easily communicate trends and changes in the performance of a system or systems over time.
- When it comes to transaction codes, CCMS is still not very intuitive today, either. Transaction codes mean little for the most part, and even the short descriptions tied to a particular transaction are not all that helpful most of the

time. Learning how to really use CCMS, therefore, really requires hands-on hit-or-miss activity (which strengthens my previous argument for providing a technical sandbox to folks tasked with learning and using CCMS).

Of course, CCMS is “free”—it is built into the SAP basis system, and therefore paid for when your particular mySAP solution is licensed. SAP AG sought to remedy some of the shortcomings of CCMS, though, with basis release 4.6C when it offered improved monitoring capabilities in the form of *transactional monitors*, a group of transaction-specific dialog monitors accessible via the “SAP CCMS Monitors for optional components” monitor collection. It also finally became possible to set up a transactional monitor for each system and even an entire system landscape. This was good for monitoring the performance of a particular set of transactions, like your “top 10” online end-user transactions or background jobs. In doing so, you could easily monitor overall response time, queue time, load+gen time, DB request time, front-end response time, and so on across a complex R/3 solution, for example.

But this approach was still limited, as it was difficult to address holistic performance of the system outside of the core transactions monitored by a specifically configured transactional monitor. And it was this shortcoming specifically that drove much of the success of third-party SAP enterprise monitoring applications like BMC Patrol, HP OpenView, and others in the last five years. Never content with standing still, SAP AG continued to improve upon its own native toolsets and offered another management alternative in the form of their Central Monitoring System (CMS, or CEN). Accessible and configurable via transaction RZ20, CEN supports monitoring

- Overall availability of the system, including general status, cluster heartbeat, status of sessions and threads used, and more.
- Detailed hardware properties, like disk space, RAM, and CPU utilization, status of system parameters, and so on.
- Detailed system information, such as the OS or Java version running on each server, its host name, and similar current-state data.
- Performance, measured in functional terms (like the number of documents processed per hour or the average size of these documents).
- Performance, measured in technical terms (like buffer quality and other memory conditions, disk hit ratios, swap/pagefile utilization, and other data typical of CCMS).
- Error logs and similar system-generated messages and events.

CEN is quite capable today, allowing you to develop custom monitors in addition to leveraging built-in ones. CEN leverages a typical software “agent” approach to monitoring servers; agents are loaded on systems to be managed, thus enabling them to

be monitored. The actual agent deployed in each environment differs, depending on the system being managed. For example, SAP AG publishes agents for systems running SAP Basis release 3x (SAPCM3X), 4x/6x (SAPCCM4X), and even non-Basis systems (SAPCCMSR). For non-SAP systems, you also want to review SAP Note 420213 as well.

The SAP Solution Manager

The SAP Solution Manager has been available since Basis release 4.6x, though it is much better known and more readily associated with Web Application Server (release 6.10 and 6.20, for SSM 2.1 and 2.2, respectively). In the short time that SSM has been available, it has improved significantly. Although SAP Solution Manager addresses much more than performance monitoring and systems management, for our purposes in this chapter the SAP Solution Manager (Operations) provides the following fundamental benefits beyond CCMS, which it leverages heavily:

- SAP Solution Manager (Operations) can monitor multiple mySAP components from a single centralized management console.
- Solution Manager ties directly into support and service functions, both online and through “packaged” best practices documentation. For example, Support Desk functionality allows your online SAP end users to contact and work with your SAP TSO and SAP’s Customer Support Organization real-time.
- Solution Manager also extends monitoring beyond mere systems management; it enables real-time business process monitoring, taking transactional monitors to the next level.
- Service Level Management (SLM) is supported as well.

The SAP Solution Manager must be installed on a dedicated server (to avoid dependency conflicts with other Web AS-enabled mySAP solutions) as an “add on” using transaction SAINT. Both “Implementation” and “Operations” are installed in the process, and either may be used afterwards. Although SSM 2.2 runs on Web AS 6.2, per SAP AG its long-awaited successor, version 3.1 will not be generally available until the Fall of 2003. Together, we will take a closer look at SAP Solution Manager, and a number of other third-party SAP enterprise systems management applications, in more detail later in this chapter. In the meantime, you may want to review the online SAP Solution Manager Tutorial, available from a “link” document found on the Planning CD.

Other Tools and Utilities

Many other tools exist that can aid you in managing your SAP systems. Hardware vendors offer hardware-specific utilities, as do OS vendors, database vendors, and so

on. The key here is to find tools that support not only point-in-time snapshots, but also historical analysis. Similarly, there are plenty of infrastructure management applications on the market today that leverage common protocols such as SNMP, DMI, and WBEM to help you manage much of your SAP Solution Stack—in many cases, in fact, these applications have matured to the point where they even offer “snap-in” modules capable of addressing mySAP components, specific databases, and more. But again, unless they are capable of displaying point-in-time as well as historical data, I see little long-term value in them. I will take a look at many of these tools and utilities shortly, after we address a method for evaluating and implementing a systems management application for SAP.

Piloting a Systems Management Application for SAP

To successfully determine the right SAP enterprise management application for your environment, and then successfully pilot, implement, and use it, a strong subteam under the SAP TSO must be developed. In the past, I've seen teams composed of folks from the following groups work well together on a systems management project:

- Manager of SAP Computer Operations, who will eventually be responsible for the team that provides 24×7 SAP coverage
- SAP Basis manager or senior technologist
- One or two senior technology-focused computer operators who manage the enterprise today, and will eventually be tasked with managing the SAP systems
- A senior technologist from the network infrastructure team
- A senior technologist from the hardware and OS support groups
- A technical member of your existing systems management team, if one exists
- One or two vendor-neutral SAP consultants intimately familiar with the workings of a number of SAP enterprise management packages

This team provides a broad perspective of monitoring and systems management practices over your current environment, and brings needed knowledge and experience in managing mySAP solutions, too. I've seen this team given the label of “Systems Management Team” or “Enterprise Management Group”—for our purposes in this chapter, I'll refer to this SAP TSO subteam as the *EMG*.

As it stands, the team may not be comprehensive enough for your particular situation; be prepared to add additional skillsets or additional IT unit representation in the case of complex, global, or otherwise large mySAP implementations.

Defining Requirements

With the EMG team created, their first priority needs to be understanding the SAP project's systems management requirements. Sometimes this becomes something of a parallel short-term/long-term effort, where a tactical solution and set of processes needs to be put in place as soon as possible in support of Go-Live, although the team also looks at perhaps different approaches or toolsets to address strategic SAP systems management. In the best of worlds, there is enough time to simultaneously define short-term and strategic requirements. Regardless, though, the team needs to develop a first cut of the operational processes that will support the SAP project's SLAs (service level agreements). This in turn will help narrow down the list of toolsets and applications that may already be (prematurely, of course, at this stage in the project) under consideration, so as to map the capabilities of various applications to the needs of the SAP project's systems management requirements.

In the short term, then, I suggest developing the following:

- A clear statement of the responsibilities that SAP operations will have in regards to the SAP environment, including where those responsibilities end and are picked up by other groups or teams
- Clear processes for SAP operations to follow in meeting these responsibilities
- A similar set of responsibilities for the various groups that may be involved in assisting SAP operations in problem escalation and resolution
- Clear processes for implementing, documenting, and updating all processes associated with managing the SAP environment

After these requirements are discussed, refined, and documented, I then suggest that the EMG turn its attention to long-term strategic requirements, for example:

- Identification of the owner of the systems management application or toolsets to eventually be installed
- Rationalizing your SAP systems management approaches with long-term toolsets and approaches that can be utilized across your larger computing infrastructure or enterprise
- Developing a strategy focused on minimizing monitoring tools, by implementing a "broader" systems management application that supports different solutions common to your unique enterprise
- Deployment of niche, specialized, or best-of-breed systems management tools that complement your broader systems management strategy

With requirements nailed down, the EMG can then begin looking seriously at what the company's IT organization is already doing when it comes to systems management, and identify areas of overlap in terms of common systems management applications that *also* support monitoring and managing SAP environments. But first, I think it's important to understand the real challenges to SAP systems management as I have seen them unfold in the real world.

Systems Management Challenges in the Real World

When it comes to implementing and using an enterprise management application, the actual design of a solution followed by the installation of various software products does not tend to be a problem. Instead, the real gotchas in my experience consist of the following:

- Fully understanding the regular operational tasks that must be performed day in and day out, or on a regular recurring basis, in a mySAP environment.
- Identifying the most critical items and conditions to monitor, at the SAP, database, operating system, and hardware levels.
- Identifying appropriate thresholds *for each of the critical items being monitored*—if you get too many alerts for a particular condition, for example, your SAP operations staff and the team that they escalate these issues to will become numb to the issue, and a *real* problem may well go unnoticed later (consider the boy who cried wolf).
- Identifying appropriate responses to alerts. In other words, which alerts need to be escalated, and at what point, and which alerts can instead be simply noted or archived?
- The need for actual references that can illuminate some of the issues with which you will be faced if you choose a particular product or application suite—even the most successful systems management implementations still have their share of “challenges.”
- The need for product-experienced folks who have implemented your enterprise management solution of choice *in an SAP environment*—SAP technical consultants experienced in HP OpenView or BMC Patrol or whatever you decide to implement will avert hundreds of post-implementation support hours in the long run, saving money and increasing your system’s availability along the way...
- Earmarking a particular person and a backup to maintain your systems management approach is key long-term as well because changes to your SAP environment generally equate to the need to update your systems management console (in terms of adding new systems to be managed, removing old systems, revising thresholds, and so on).

With this in mind, we can now turn our attention to how your IT organization currently manages its computing resources, with the hope of leveraging some of this in-house expertise and experience to make better decisions when it comes to managing SAP.

Leveraging In-House Applications

Looking inward at the suite of management applications, utilities, and tools in place in your organization today makes a lot of sense before further scrutiny of a “new” SAP enterprise management application is in order. For example, if you are running an IBM shop, and have deep experience with Tivoli and other IBM-centric tools, it might save a lot of time and energy to leverage this expertise and implement Tivoli for SAP management as well. The same goes for HP shops, where HP OpenView might be the best way to go simply by default. Similarly, if you have adopted Computer Associates’ software products across the board, CA Unicenter might be a natural fit for your organization just because of the pricing and support benefits that can be gained given your existing relationship with CA.

On the other hand, in my experience even the best general enterprise applications change significantly after SAP is introduced. This can almost eliminate the benefit of implementing a particular enterprise management application simply because your organization already uses it. Why? Because, at minimum, an SAP software agent or similar construct must be added to the solution stack. And other agents new to your technical organization may be required as well, bringing with them their own changes and complexities.

A typical example is HP OpenView—thousands of systems management support organizations around the world leverage HP OpenView to manage their network and server infrastructures. But if you want to create an end-to-end SAP management solution, you really need to add OS, database, and SAP agents to various servers within the SAP system landscape. And you might require the deeper capabilities of a snap-in hardware management tool on top of these other changes. Even worse, if you implemented an HP UNIX-based solution for network management, for example, but have mySAP running on top of a Microsoft Windows or Linux architecture, the actual versions of many of your HP OpenView software components will differ. In the end, the HP OpenView solution originally implemented and understood by your company will hardly resemble the HP OpenView systems management solution implemented for SAP, negating much of the operational benefit you hoped to gain in the first place.

Regardless, I still like the idea of at least *considering* your current systems management approach when it comes time to investigate the realm of management possibilities. Worst case, you will at least benefit from a pricing perspective. In the next section, I’ll look at other factors that help you put together a “short list” of enterprise systems management candidates.

Creating the Short List

Where I have been engaged to assist my customers with selecting and implementing an SAP enterprise management application, my client and I tend to narrow down the initial list of prospective management solutions in terms of the following categories:

- The capability of each potential management solution to address a particular client's requirements. Feature set, capabilities, and underpinning technologies used (especially with regard to clear differentiators) are all important factors. Don't forget to include the current systems management approaches and toolsets, too, unless they are grossly incapable or simply unsupported by the specific SAP Solution Stack being implemented.
- Additional capabilities that, although not deemed critical in the present, may prove valuable in the long run. Examples of these kinds of capabilities might include the ability to schedule and monitor batch jobs running on multiple SAP or other systems (sometimes referred to as a *cross-application batch scheduler*), or the ability to monitor servers running various operating systems, databases, or Internet services.
- The underlying method or architecture leveraged by a particular management application to interact with (and therefore monitor) SAP. For example, some applications leverage native SAP ABAP code to manage, monitor, and collect data pertaining to an SAP landscape, but others require deployment of a separate server and associated management database.
- Overall cost and licensing structure, which normally includes software, hardware, documentation, and support. Thus, things like the license cost of a central management console, or costs of software agents for each SAP server to be monitored, and even the cost of purchasing printed installation guides, all come into play.
- True value provided above and beyond that provided by SAP's CCMS—I usually sum up this by noting what I believe to be a tool's greatest differentiator, like its low cost, vast enterprise systems breadth, superior historical analysis and archiving abilities, robust problem/event correlation, and so on.
- The ability to address event management and escalation well, including the ability to route problems to the proper "owner" or on-call entity, and notify responsible parties through multiple channels (via pager, email, cell phone, and so on).
- Robust reporting capabilities, including the ability to create standard and ad-hoc reports, the ability to define "downtime" and consequently report availability statistics (SLA data), the ability to automatically generate and distribute regular reports (through email or via a Web site, for instance), and more.

- Ease of installation and system maintenance, and access to SAP-specific product support (that is, support tailored to the SAP solution stack being implemented), including the ability to automatically obtain updates or patches and general support through the Internet or other expedient means.
- References—other SAP customers running an identical (or nearly so) SAP Solution Stack, with the identical systems management application being considered. A confident vendor will provide customer names and contact information after they make the short list.
- In-house considerations—consistency with known in-house technology strategies going forward (such as a tool's long-term fit in a company-wide enterprise systems management capacity), or presence of in-house skillsets.
- The ability to implement a particular systems management application within a required time frame; some tools take longer to set up and master than others, or are otherwise more complex than others.

Often the “short list” can be further refined simply by virtue of the strengths and weaknesses of your SAP operations, basis administration, and help desk teams. And your relationship with a particular systems management vendor can also sway your decision one way or the other.

Evaluating Enterprise Systems Management Applications

With the short list of SAP enterprise systems management candidates in front of you, I suggest creating a comparison matrix to facilitate comparing and contrasting the different solutions. I have successfully used a simple Excel spreadsheet approach in the past—the details related to costing can be clearly deciphered in this way, and arbitrary scoring for the categories described in the previous section (plus any others you create) can easily be tallied up to provide a final “score” for each potential solution.

In revisiting the broad category of cost, for example, I like to break down the following line items:

- Software—Basic management console
- Software—Basic agents for each server to be managed
- Software—Uplift for special OS agents required
- Software—Uplift for special database agents required
- Software—Uplift for special mySAP component agents required

- Software—Any operating system license(s) that may need to be purchased, for a dedicated systems management console or other similar solution-inherent component
- Hardware—Any required server, desktop, or similar solution-inherent component, such as the cost associated with a central management console or software distribution server.
- Documentation—Printed and electronic versions of all installation and other software- and hardware-based guides or documents.
- Support—Design or architecture support, including any time that needs to be spent reassessing requirements.
- Support—Installation.
- Support—Configuration of the product(s), such that a product may actually prove useful in managing your SAP environment. This includes figuring out what thresholds to configure, how to monitor specific events, setting up pager/email alerts, and so on.
- Support—Setting up reporting and similar features outside the scope of “configuration,” again making a product truly useful and valuable.

With regard to support, it needs to be noted whether internal resources (company employees or contractors) can be leveraged, or if access to external consulting expertise is required. The trade-off in time versus cost needs to be noted, too, when it comes to engaging consulting resources to install, configure, and support a particular systems management application. All of this usually helps to paint a clearer picture of each systems management option, ultimately illuminating one option as superior over the others.

To help you narrow down your own options, and benefit from my experience, I take a closer look at some of the most common third-party SAP systems management tools in the next section. And for each of the following enterprise management applications—BMC Patrol, CA Unicenter, HP OpenView VantagePoint, and NetIQ AppManager Suite—I consider the various categories and criteria set forth earlier in this chapter, identifying clear differentiators where warranted.

BMC Patrol

BMC Software’s core systems management application, Patrol, is probably the most well-known tool for SAP out there today. The foundation or code-base for BMC’s Patrol product lies in ABAP, which may be good or bad depending on your point of view. That is, for shops that like the idea of maintaining all systems management reporting data in the same database as that used by the SAP system, Patrol is a great

way to go. Similarly, those shops adept or comfortable with coding in ABAP will enjoy BMC's flexibility in developing reports, creating custom events, and so on. Other differentiators from the pack include

- BMC's large installed base can be leveraged in terms of deployment experiences as well as consulting and other SAP-related expertise.
- Supports SAP APO, Business Information Warehouse, CRM, EBP/SRM, R/3, and ITS, and likely newer "mySAP.com" components as well—see BMC's "Patrol for SAP Solutions Suite" Web site at http://www.bmc.com/products/proddocview/0,2832,19052_19426_23176_7206,00.html to review all currently supported mySAP.com components.
- Using a custom scripting language called PSL, Patrol can be used to easily develop powerful custom monitoring and reporting screens.
- Because Patrol pulls classes of data rather than discrete one-offs, it is easy to tune and troubleshoot Patrol-managed servers.
- SAP events can be combined to create custom events, which can prove very useful.
- BMC also has software and modules for nearly every supported SAP server and storage system on the market today, including modules for SANs, SAN and network switches, and more. This, along with mature support for snap-ins from many SAP hardware vendors today, makes it easier to deploy an end-to-end BMC management solution.
- Finally, BMC offers support for a cross-application job scheduler (Control+M), and a document/report management tool (Control+D).

With its popularity and proven success in managing SAP environments, BMC Patrol is certainly a safe choice to make. At something like twice the price of some of its less capable competitors, it's not the cheapest offering on the market, however.

CA Unicenter Application Management for R/3

Like its tier-1 competitors, Computer Associate's (CA) Unicenter Application Management for R/3 monitors activities such as the creation of ABAP "short" dumps, status of active processes, system availability, and more. It watches over enqueue locks, users, and spool problems, too. Other key capabilities or differentiators include

- CA Unicenter can monitor the number of short dumps created when a report or transaction aborts, and then trigger a message to both the CA Unicenter Event Console as well as page a member of the SAP TSO.
- Multiple servers across diverse SAP system landscapes can be monitored simultaneously from a central site—all alerts are sent to a single Event Console.

- Unlike SAP CCMS, CA Unicenter monitors all system alerts, not just the most severe.
- Both the number of running processes, and the fact that a process may be running over an extended period of time (a “long-running” process), are noted by Unicenter. This includes detailed information on the status of update work processes, too, including whether enough are configured based on the workload of the system.
- Unicenter allows you to track performance based on both historical analysis and observed end-user response times. In this way, both non-SAP and SAP resources can be correlated to create powerful views into the entire enterprise.

Perhaps Unicenter’s biggest differentiator is its ability to schedule jobs with dependencies across applications, such that the successful completion of a non-SAP job will automatically trigger an SAP job to start, and vice versa. This lets you avoid scheduling jobs manually, saving time and eliminating human errors. Unicenter accomplishes this by using SAP’s open job scheduling BAPI, or Business Application Programming Interface. In this way, Unicenter can schedule both SAP and non-SAP jobs from the same console, monitoring job dependencies and automatically executing jobs as these dependencies are satisfied. This feature is not unique, but because it’s included with Unicenter (rather than offered as just another bolt-on tool to your core systems management application), I like it.

HP OpenView VantagePoint for Windows

HP’s systems management architecture for mySAP is probably the best on the market, in that it’s modular—a “SMART Plug-In” for various applications, such as SAP R/3, SQL Server, Oracle, Informix, Internet services, and even Windows 2000, allows for managing an end-to-end solution stack from a single console. Other advantages of HP OpenView’s intelligent agent technology approach include the following:

- SAP computer operators and systems administrators are provided with complete and continuously updated graphical maps and views of the SAP Solution Stack (rather than a less intuitive list of devices like that displayed by its competitor’s products). Further, these maps and views can be tailored for a particular audience, allowing a hardware infrastructure specialist to monitor only specific servers and disk subsystems for which he is responsible, for example, whereas a network specialist’s console might be configured to include only network devices and those tools required to manage them.
- Other specialized maps can be created that are application-specific or even business-process-specific—a line-of-business manager might request a system map that reflects data on new orders taken, for instance.

- HP enjoys the tightest relationship with SAP of any SAP systems management vendor in business today due to HP's status as SAP's #1 global technology partner.
- HP OpenView supports not only R/3, but also BW, ESS, ITS, and SAP's eProcurement offerings. And with its strength in managing both Windows- and UNIX-based SAP systems (including HP-UX as well as Sun Solaris), OpenView is indeed well-positioned to manage an entire enterprise.
- At installation, the SAP agent (SPI) offers instant discovery of the entire SAP system, including dependencies between it and other systems, instances, and processes. Then, with its integration into SAP's CCMS, it can help you manage multiple SAP systems running different release versions while ensuring that end-to-end response-time service levels are met.
- In many cases, automatic actions resulting from triggered events can correct problems immediately, helping to ensure constant service availability. "Due to real-time monitoring capabilities and centralized administration, we are now able to prevent most faults up-front," reports Alain-Michel Cieters of Usinor. "OpenView VantagePoint automatically builds relationships between technology and business services, so managers can immediately determine how technical problems affect a business service."
- Initial pricing is mid-range, but because it includes annual maintenance fees, it is actually on the low side of average.

Given its breadth of coverage and potential complexity, though, HP OpenView is not the fastest enterprise systems management product you can implement. On the contrary, a typical SAP systems management implementation from start to finish could easily take three months. And with a robust set of actions, alerts, events, and corresponding thresholds, an expert in both HP OpenView and SAP needs to be employed to help ensure a successful OpenView deployment.

NetIQ AppManager for R/3

Like its competitors, AppManager for SAP R/3 offers central monitoring and management capabilities for SAP. As one of a number of components of NetIQ's AppManager suite, AppManager can also manage SQL Server and Oracle databases from the same console. Beyond its best-in-class interface, it has additional benefits as well:

- A unique *proxy* solution for managing servers and resources helps keep your SAP servers "clean"—no agents are loaded on your servers, and therefore you never have issues with memory leaks or solution stack conflicts when it comes to systems management agents, for example.

- A tight partnership with Microsoft can be quite beneficial to SAP customers already betting their solution stack on Microsoft's OS and database products.
- On the other hand, the use of proxy servers creates additional failure points within your enterprise management framework, unless they are clustered (in which case, they then become more complex to manage than their competitor's solutions).

Limitations are numerous, unfortunately. For example, AppManager historically only supported Windows environments—there was no UNIX support until recently, beyond the fact that the database server could reside on any SAP-supported platform. And of the UNIX flavors accommodated today, SuSe Linux (SAP's key Linux partner) is missing. Further, as of this writing only SAP R/3 is supported. And AppManager's proxy approach can only support monitoring something between five and eight SAP instances. Finally, there is no support for archiving or even purging old performance and availability data; instead, each customer must develop their own (admittedly easy) solution to this shortcoming. All of this explains NetIQ's weaker penetration of the SAP enterprise management market relative to the previously discussed enterprise applications.

Enterprise Management Applications—Lessons Learned

When you understand the importance of managing your SAP systems, you will have a strong desire to create a powerful, full-featured enterprise management solution. However, because the time to really pull this off is generally prohibitive, I have found that the following rules of thumb help my SAP customers start off on the right foot and keep them on track to successfully deploying their enterprise management solution:

- Keep it simple; deploy the basic systems management package before attempting to implement additional hardware vendor-specific snap-ins, or specialized agents, and so on.
- Utilize “out-of-the-box” SAP monitoring capabilities. In other words, refrain from changing the default management parameters associated with deploying the basic package.
- Turn off monitoring of parameters that are not of immediate or critical need—focus on the most critical performance and availability statistics first.
- Deploy a “systems management testing” environment for monitoring, and use this environment to initially manage your test or technical sandbox systems, so as to evaluate and later refine how your systems management tool performs. Often, this special testing environment is simply an extension to your SAP Technical Sandbox (in the form of an incremental server and storage, for example).

- Adjust SAP and other solution stack performance and HA thresholds based on input from consulting or in-house subject matter experts. At a minimum, in the absence of this expertise, stick with the defaults until you have a better grasp of the tool.
- Develop a fall-back plan in parallel to deploying your automated systems management solution, preferably in the form of SAP CCMS-trained computer operators and help desk technicians armed with paper “SAP Operations” checklists.
- Leverage your “systems management testing” environment to practice continuous review and improvement of your systems management approach both before and after Go-Live.

Although your SAP enterprise systems management application is being developed and deployed, you would do well to take a look at other management tools and approaches. After all, it’s unlikely that a single enterprise tool can address more than 50–75% of your solution stack, much less all of it (HP OpenView is a notable exception, covering perhaps 90% of your stack on its own). In the following section, I detail some of the common tools and utilities that make end-to-end SAP operations and systems management possible by filling in your systems management “holes.”

Additional SAP Management Tools and Approaches

Several monitoring and management tools and single-purpose utilities are available to assist you in managing your mySAP solutions. Many of these are “free” tools for monitoring hardware components and alerting you of potential issues. Insight Manager, for example, ships with each HP ProLiant server and “snaps in” to HP OpenView and other enterprise management applications, which can then give you enterprise-wide monitoring capabilities from the hardware layer up through the mySAP component application layer. Other tools and utilities are built into different components of your solution stack, most notably the operating system layer. Still other utilities and applications can come into play to extend your solution monitoring capabilities, for example, across disparate systems, or up to the client desktop, or down to proprietary storage and tape-backup solutions. In the next few sections, I take a closer look at many of these tools and approaches, and note how truly useful they have proven themselves time and again in my own consulting travels.

Hardware Management Tools and Utilities

I’m a big fan of using utilities that typically come bundled with their respective hardware platforms, such as HP’s Insight Manager and TopTools offerings, Dell’s Open Manage, and Sun’s Sun Net Manager. Utilities like these provide a foundation for enterprise-wide systems management, often working seamlessly with all-purpose

management applications such as HP OpenView and CA Unicenter to provide deeper insight into a solution stack's hardware layer.

Other tools can provide quite a bit of value from an operations perspective, too. For example, after each change control wave at a number of my productive SAP sites, either the SAP operations or Basis team executes a set of scripts that verify the performance of the production disk subsystem. Remember from previous chapters that the disk subsystem tends to be one of the biggest culprits behind SAP performance problems. By executing a set of scripts over a period of 5 to 10 minutes at the tail end of your planned downtime window, you can verify the performance impact that a planned system change may have one last time before turning a system back over to its users. My two favorite utilities in this regard are Microsoft's SQLIO and Intel's Iometer products. Both are easy to set up and simple to execute.

- ▶ To read more about how SQLIO and Iometer are leveraged in terms of systems testing, see "System-Level Stress Testing and Pre-Tuning," p. 574 in Chapter 16.

Other hardware-based tools are also focused on disk subsystems or storage devices. HP's SAN Management Appliance, for example, allows you to monitor the activity observed throughout an entire Storage Area network, from individual disks or groups of disks (called LUNs), to disk controllers, to individual ports on a fibre switch. This granular management capability assists SAP operations teams in quickly identifying the source of SAN-related performance issues that might crop up. It also allows for historical analysis in terms of collecting and analyzing typical throughput numbers observed for specific SAN components under certain conditions (like month-end or seasonal peak processing). Many other disk controller and disk subsystem utilities exist, as well.

Finally, additional hardware management and monitoring tools enable you to monitor specific pieces of your SAP Solution Stack. Examples include the following:

- HP's (formerly Compaq's) *RackBuilder*, which allows you to plan for deploying new servers and storage, or change the makeup of existing racks, including calculating weight-bearing, power consumption, and thermal requirements.
- Uninterruptible Power Supplies (UPSes) ship with utilities used to manage and track battery consumption and status. In some cases, third-party enterprise management applications can perform some of the same general features, too.
- Tools that measure or monitor a specific aspect of a disk subsystem are common, too, especially those related to managing the availability of redundant storage paths (such as *OpenPath* and *SecurePath*) or the replication of data between different LUNs or storage systems.
- Tape drive/library management tools and utilities, like those offered by leading tape backup and restore vendors.

- Utilities used to manage network infrastructure that interconnects SAP solution components, or software that facilitates managing a server's network cards, represent common network management tools. And, of course, some of the most robust systems management applications on the market today were born from these humble beginnings, too.
- Utilities that manage hardware-based IP load-balancing devices (such as those found front-ending SAP eProcurement or Employee Self Service solutions). Java applets and other Web-based utilities are quite common in this regard.
- General system performance tools, such as Smith Micro Software's excellent *CheckIt Utilities*, which allow for measuring and baselining the performance of a server's major subsystems, including processor, RAM, disk, and so on.
- Hardware documentation tools, such as Ecora's *Documentor*, HP's *Survey* utility, and Breakout Software's *MonitorIT*, document hardware and software configurations—server software revisions, firmware levels, patches applied to the system, and more. And I find the freeware program PrintKey quite helpful when it comes to capturing screen shots for documentation, too.

Of course, other utilities prove themselves useful to a technical support staff day in and day out, like PKWare's ubiquitous compression tools, any number of virus protection programs, and so on. I suggest using the preceding list as a starting point, however, for creating an *operations toolbox* of sorts that matches the monitoring and management requirements of your particular SAP Solution Stack.

OS Management Utilities

At an operating system level, I have found without exception that built-in OS utilities like those associated with Windows and Unix variants offer immense value to SAP operations teams. Anyone tasked with managing and monitoring a mySAP solution needs to add the following utilities to their operations toolbox:

- Windows MMC (Microsoft Management Console) is Microsoft's premier OS management tool. In fact, it is so powerful in the case of pure Microsoft SAP solutions that I give it more attention in a dedicated section towards the end of this chapter, under "Specialty Utilities."
- Window's Performance Monitor, or *PerfMon* for short, should be used for any Microsoft-based solution component both to create a baseline and then to monitor system performance proactively on a regular basis. I like to run PerfMon tests after any change management waves (before the system is turned back over to its end users). Similarly, I like to capture PerfMon statistics every 30 or 60 seconds or so, to provide a baseline from which to compare future system performance. Most important in my eyes are disk queue lengths, mix of

reads to writes, CPU statistics, RAM utilization, pagefile utilization, and basic network I/O.

- A host of Unix command-line utilities are used regularly in support of day-to-day SAP systems management. One of the most common utilities is *vmstat*, used to monitor the run queue, swap file page-in count, paging daemon scan rate, CPU status, and more. Another common utility is accessed by executing the single letter *w*, which shows process usage by monitoring load averages and generating an abbreviated *Top Sessions* report with load averages over 1, 5, and 15 minutes. For a full CPU activity report, *top* can be executed. And for swap file performance measured as a percentage of swap space utilized, *swapinfo* is valuable.

Many other utilities and tools are available as well, especially in the world of various Unix flavors. Refer to your operating system's administration guide for details specific to your OS release.

Database Management Tools

Of course, a database vendor's own tools and utilities are most often leveraged for basic database administration outside of SAP's CEN and CCMS T-codes. This includes Microsoft's SQL Server Enterprise Manager and Oracle's Enterprise Manager. However, other tools may also prove useful. Oracle's Statspack utility, and Unix-based *bstat* and *estat* command-line utilities are classic examples, as are various SQL trace tools. And *iostat*, a common Unix utility, can be employed to display real-time disk subsystem performance, too. Finally, *sar*, the System Activity Reporter, can be used to report disk I/O and buffer activity.

I also include various data archive utilities and applications under the category of database management tools. Of course, because this involves database/SAP administration expertise, functional expertise, and the support of an SAP operations team, it's probably not fair to call them strictly operations tools. However, after an archive solution and approach is nailed down, the process of extracting business data out of your productive SAP instances and relocating this data to other storage media eventually represents an operational activity.

More Value from the SAP Solution Manager

The SAP Solution Manager can fill in quite a few holes that other solution stack tools leave empty. As you see in Figure 14.5, it embraces much more than simply technology management, adding change management and business process management to its core capabilities.

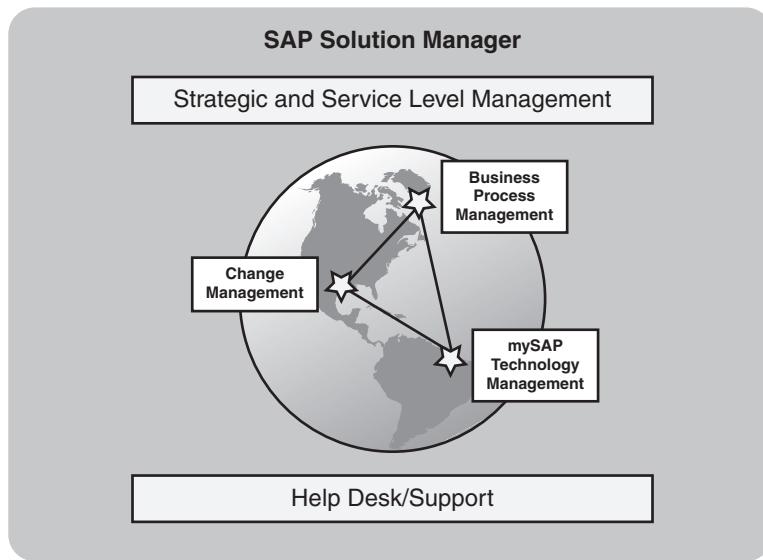


FIGURE 14.5 The SAP Solution Manager brings together SAP's know-how under one management umbrella, and underpins this with Service Level and Help Desk capabilities.

For example, it's central command station approach to operations management and monitoring can oversee your entire mySAP system landscape, even including third-party system interfaces. And because the SAP Solution Manager is a self-documenting data repository, it provides great value beyond the operations functions available from its main user interface screen.

The operations section of the SAP Solution Manager includes three broad categories of services, facilitated by detailed process documents as well as traditional SAP consulting engagements:

- Predictive and Proactive Services, which includes SAP's *GoingLive* and *EarlyWatch* services, both of which analyze mySAP systems end-to-end (the first serves to audit your system prior to going live, whereas the latter seeks to assure excellent ongoing operations). Note that these services can be performed as a set of self-service procedures, or via more traditional onsite or remote consulting venues.
- Continuous Improvement Services, delivered exclusively by onsite SAP consultants, which include analysis or optimization of things like system administration, storage subsystem performance, SQL statements, and even business process management and more—everything needed to improve the performance and availability of your mySAP solution across the board.
- Best Practices, reflecting best-of-breed documented processes, procedures, and services specific to managing each mySAP component.

Beyond these operations services, though, lie monitoring and support capabilities. Especially valuable are SAP Solution Manager's real-time system monitoring, service-level reporting, and business-process monitoring—again, everything needed to ensure a well-performing system. And the presence of an online help desk (discussed previously), support for real-time assistance through Microsoft's NetMeeting product, and the ability to search the SAP Notes database are a huge boon for SAP operations teams. Remember, SAP Notes are provided to SAP's customers to share information regarding corrections and enhancements—SAP Solution Manager's ability to automatically implement SAP Notes rounds out its offerings.

SAP's Solution Services

To implement the SAP Solution Manager, I recommend leveraging SAP AG's own consulting services (at least until other consulting organizations are as familiar with implementing the product). SAP has assembled a service offering under its "mySAP Services" umbrella called *Solution Operations Services*. Solution Operations Services focus on implementation, integration, operations planning, and systems optimization—the entire spectrum of issues encountered throughout an SAP system life cycle. Specific operations-related services include

- Implementation services, which focus not only on solution conceptualization but also on the technical implementation of business processes, including how operational processes can be built into your solution
- Systems integration, which addresses integration of your existing systems with your mySAP components
- Operations planning, which assists you with planning for daily systems operations, including infrastructure planning and overall SAP system change management
- Optimization services, supporting continuous improvement through capacity planning, configuration management, performance optimization, system upgrade/migration services, and more

Plus, in the end you can also take advantage of the SAP Solution Manager itself, allowing you to effectively manage multiple mySAP systems from a single platform while SAP's consulting organization waits to assist you further if the need arises.

SAP Note Assistant

Another tool I've seen deployed by either the SAP Basis team or a capable operations team is the Note Assistant. Used to rapidly implement specific SAP Notes (code updates, bug fixes, and so on), the Note Assistant makes it easy to install these corrections to mySAP components. Dependencies between SAP Notes, Support

Packages, and other system modifications are tracked as each is implemented, to ensure that they are implemented in the correct or best order. Together, these powerful capabilities make the Note Assistant invaluable to the SAP TSO, and an important addition to your change management processes.

However, for SAP projects that have implemented the SAP Solution Manager, the Note Assistant is no longer required—these capabilities are integrated into the SAP Solution Manager.

Prior to the Note Assistant, to implement the corrections and enhancements found in SAP Notes, you had to import them as part of an SAP Support Package (which meant waiting until the support package was released by SAP), or worse, manually insert the corrections into your own mySAP source code. With the Note Assistant, this is performed easily and quickly without the need to touch your own system's source code—a huge improvement in regards to making error-free changes and therefore increasing system availability.

Additional compelling reasons to implement the Note Assistant in your mySAP environment include

- Support for SAP Basis releases from as far back as version 4.5 (implemented as an add-on); as of release 6.10, it is integrated as part of the Basis installation.
- Automated implementation of SAP Notes allows you to download the notes appropriate for your particular mySAP solution from SAP's Service Marketplace and load them directly into your technical sandbox, development, or development/test systems.
- *Dependency resolution* ensures that corrections are implemented in the proper order.
- Notes are protected in terms of modifications that your own team may have made to the system, to ensure that your modifications are not overwritten while also guaranteeing that only pristine code is applied to your systems.
- The Note Assistant's intuitive user interface and features like a split-screen editor facilitate fast implementation of SAP Notes, without all of the manual work required in days gone by.
- Comprehensive reports as to the SAP Notes that you have implemented, or the status of SAP Notes in progress, aid in project organization and administration.
- SAP Notes can be assigned to particular SAP TSO members for processing; status of their work can easily be checked as well.

With SAP's Note Assistant, handling and implementing SAP Notes makes the process of correcting small bugs in your system easier. This feature, combined with updates

and fixes found in regularly released SAP Support Packages, helps guarantee a stable well-performing SAP environment and represents an excellent alternative to the SAP Solution Manager when working with older SAP releases.

Specialty Utilities

In my travels, I have come upon a number of specialty tools that I believe aid an operations or SAP support team significantly. In the next few sections, I discuss some of these with regard to managing mySAP solutions—the Microsoft Management Console, a number of cluster administration tools, and multiple remote management options.

Microsoft Management Console (MMC)

As I indicated earlier, the Microsoft Management Console (MMC) is Microsoft's premier OS management option. But it is much more than that when it comes to SAP solution stacks that leverage Microsoft databases, for example. Combined with SAP's MMC snap-in, the MMC becomes a full-featured SAP solutions management tool—it supports managing the OS, database, and mySAP components all from the same console.

During an SAP installation, the MMC and its SAP snap-in are loaded on the central instance, database server, and any installed application servers. Thus, it's quite easy to manage these individual servers from a local perspective. However, an SAP operations group would find this style of management cumbersome, time-consuming, and therefore largely ineffective. So SAP AG assembled a package and set of instructions with which the SAP MMC snap-in could be installed on a central management station and configured to manage and monitor multiple SAP instances. In this way, even machines without any installed R/3 instances, for example, can be used to monitor any number of R/3 instances remotely. And for Windows 2000 environments, no prerequisite software need be obtained or loaded (as opposed to NT 4.0 environments, where minimum versions of Microsoft Internet Explorer 4.0, Microsoft ADSI, and the core Microsoft Management Console version 1.1 needed to be installed).

To load the SAP MMC on a machine without an installed SAP instance, perform the following steps:

1. Extract the SAP-related files from the SAR archive provided by SAP on a distribution CD (or available via download), by executing the following:
`sapcar -xvf sapmmc.sar`
2. Copy files `sapmmc.dll`, `sapmmcms.dll`, `sapmmcd6.dll`, and `sapstartsrv.exe` to your `%windir%\system32` directory.

3. Verify that `librfc32.dll` is installed in the `%windir%\system32` directory. If there is no `librfc32.dll`, or if the `librfc32.dll` that resides in the `sapmmc.sar` is more recent than your version, copy it to the `%windir%\system32` directory.
4. Using a command prompt, navigate to the `%windir%\system32` directory.
5. From this command prompt and directory, register DCOM Typelibrary:

```
sapstartsrv -t
```

6. Next, register the SAP MMC snap-in:

```
regsvr32 sapmmc.dll
```

7. Register the SAP MMC extension snap-in for Microsoft SQL Server, as necessary:

```
regsvr32 sapmmcms.dll
```

8. Register the R/3 MMC extension snap-in for DB6, as necessary:

```
regsvr32 sapmmcdb6.dll
```

9. If, on startup, other DLLs need to be registered, perform the same procedure I've outlined in the preceding eight steps. In my experience, I've had to also register the `librfc32.dll` with `regsvr32`, for example.

10. In order to actually use the SAP Logon functionality offered by the MMC snap-in, you must install the SAPGUI on the machine, too. I highly recommend this.

11. Finally, to use the SAP Database MMC extension snap-ins, you need to also install your database vendor's specific database management tools on the machine.

At this point, you can begin configuring the SAP MMC as a central management tool by simply executing MMC from a command prompt, and then executing the following procedures to add SAP instances to be managed:

1. Add the R/3 MMC snap-in by clicking "Console" and then selecting the "Add/Remove Snap-in" option.
2. Click Add.
3. Select SAP R/3 Manager from the list displayed, and click OK.
4. Follow the installation wizard by confirming or changing refresh periods (R/3 Manager Refresh Periods).

5. Select Use Fixed R/3 Instance List and enable Expert user mode (R/3 Manager Operation Mode).
6. Add each of the desired R/3 instances to be managed by clicking Add. The Import option can be used, too, to add SAP instances from a list of installed instances.
7. To begin using the SAP MMC, after the MMC snap-in has been set up, you can expand the SAP R/3 Systems node in the scope pane (the left side of the screen). Also, refer to the Planning CD for a basic how-to document regarding usage.
8. Select an R/3 instance-level node.
9. Right-click this node to access and use the context menu, noting that the menu that is displayed includes tasks and actions like the following:
 - a. Start
 - b. Stop
 - c. Logon
 - d. Task->View Start Profile
 - e. Task->View Trace File
 - f. Task->View Developer Traces
 - g. Task->View Environment
 - h. Task->Restart Service
10. Choose the desired operation from this context menu.

I find the SAP MMC especially useful in managing my SAP/SQL Server customers: With its ability to stop and start SAP, check the status of OS, DB, and SAP alerts, and generally drill down into the deepest SAP Basis and database layers, the SAP MMC is an excellent all-around administration and troubleshooting tool, too.

Cluster Administrator and CCMon

Cluster management utilities, such as Microsoft's Cluster Administrator and HP's Cluster Consistency Monitor (CCMon, for HP-UX clusters only), help an SAP shop manage the inherent complexity of an SAP cluster. Even a SAP Cluster Specialist can miss defects and inconsistencies easily identified by CCMon, for example, including

- Different versions of SAP start/stop scripts between cluster nodes. In the best case, inconsistencies at start-up are observed, and other support problems are likely. Worst case, though, SAP will be unable to start after a failover to a different node.

- A file or login group exists on the primary node, but not the secondary node. In this case, TCP/IP-based communications may not work after a failover.
- Important R/3 Kernel parameters are at different values. This could lead to system monitoring thresholds never being reached in the best of cases. On the other hand, if a failover is caused by this, and the node that SAP fails over to maintains even *lower* thresholds, the system may continually start and immediately crash and fail over again.
- Software products installed on the primary node, but not the secondary node. This could simply indicate an extra application not necessarily needed at all. But it might also reflect a critical missing application such that failover would simply not work due to an unsatisfied application-dependent prerequisite.
- Patches installed on only one node, or a difference in patch release versions between nodes. This might never manifest itself if the differences in patch levels are small and the reason for developing the patch in the first place never manifests itself. However, if the patch fixes a critical error, a failover to the node running the older patch release would appear to exhibit an old “resolved” problem, and possibly complicate troubleshooting the issue again.
- Swap size significantly smaller than another cluster node. Performance degradation will therefore occur after a failover. Worst case, if the swap file is too small, SAP might actually crash again under what appears to be a normal load of online users and batch processes.
- Crash dump area not present, or differs in size, or is occupied. In this case, it would be possible to suffer from a crash, and execute a failover, but obtain no dump data to analyze later.
- An SAP-specific group missing. In the best case, some functionality is sacrificed; worst case, SAP cannot even be started again on a particular cluster node.

As you can see, the value provided by a good cluster management utility is clear, and such a utility should be added to the operations toolbox of any clustered SAP enterprise.

Remote Management Tools

Remote management tools are popular in SAP implementations, and often installed during the operations phase of an SAP implementation (if not already installed to enable system installation). Software packages like pcAnywhere, Windows Terminal Services, and PC-Duo are all quite popular, and equally effective in my opinion. And given the growth in “lights-out” data centers, approaches like these will only continue to grow over time.

For *out-of-band management*, which is provided outside the boundaries of your network, hardware-based approaches like HP's Remote Insight Lights-Out (RILO) Board and Sun's dedicated system controller that supports remote management for "lights out" operations are also quite popular. Products like the RILO board occupy a PCI slot in a server (or are built into the server's motherboard), and typically allow for connection via a dedicated secondary network segment or dial-up phone link. This minimizes a technical support organization's investment in staff in that it reduces systems monitoring workload while simultaneously focusing their efforts in an exception-based manner. And in the case of all out-of-band management products that I am aware of, these snap in or are otherwise integrated with the specific hardware vendor's systems management utilities. For example, Sun's dedicated system controller is manageable through their flagship management utility, Sun Management Center, thereby offering a single point of management for all Sun systems and storage components running under the Solaris OS, up to and including applications running on Solaris. HP and IBM offer similar products and approaches.

Tools and Techniques

Quite a few SAP operations-specific and other supporting documents can be found on the Planning CD, including the following:

- A sample Current-State Solution Stack Matrix: a simple Microsoft Excel workbook divided into solution stack layers, detailing each layer through in-line text and embedded documents.
- A sample Help Desk Top 20 Issues and Resolutions, which can be used as a template for developing your own monthly communication vehicle (to be shared with the end-user community as well as the SAP TSO).
- Both a sample Quick-Check and a sample Delta Guide have been included, illustrating two different approaches to creating system installation checklists or scheduled operations procedures.
- Operational checklists to be executed on a monthly, quarterly, and annual basis have been included.
- For template and general operational purposes, I have included examples of "how to" documentation on the Planning CD.
- Automated CCMS scripts, some coded in AutoIT (freeware) and others in AutoTester ONE (you need to purchase a runtime or developer license to use AT) are included. Remember, these scripts will *definitely* need to be slightly modified to execute on your own system. But they're very cool, and worth the effort!

- Miscellaneous illustrated “how to use” documents, which detail the steps and processes required to use various systems management utilities like the SAP MMC and PerfMon.
- Each of the figures found in this chapter, in Microsoft PowerPoint format, as well as other miscellaneous operations-related documents, a link to the SAP Solution Manager tutorial for version 2.1, and more.

Note that the Daily and Weekly SAP Operations Checklists referenced in this chapter can be found instead under Chapter 10’s *Planning CD* directory, as these documents were first mentioned and detailed in Chapter 10.

Summary

We covered a lot of ground in this chapter, creating a foundation for both SAP systems management and SAP computer operations. The SAP operations manual was detailed, including both what to document and how to do so using the various current-state, regularly scheduled, and how-to documents needed to support a mySAP enterprise. I then provided real-world documentation practices, followed by a detailed look at various systems management techniques for SAP. Next, a process for selecting, evaluating, and implementing an SAP systems management application was discussed, backed up with my real-life observations. I also provided my own assessments of four of the most popular SAP management applications on the market today—how they stack up to each other, product differentiators, relative weaknesses, and more. I wrapped up Chapter 14 by identifying common utilities and tools that help fill in the holes left by even the most excellent of enterprise management applications. My solution stack approach to doing so should prove useful to anyone wanting to identify and add a utility to their SAP shop’s operations toolbox.

15

Functional, Integration, and Regression Testing

Testing SAP Business Processes

After our SAP developers have coded solutions that address our company's discrete business processes, testing beyond simple unit testing becomes necessary. That is, the months of work involved in translating, upgrading, or creating an organization's business processes into SAP best practices must culminate in a comprehensive testing phase. No implementation is complete in the absence of testing. Indeed, no project will succeed without it. This brief chapter focuses on a number of types of business-process-specific testing that must occur, when each must occur or is appropriate, and how each type helps to ensure a smooth transition from an old way of doing things to an integrated mySAP.com-based approach.

Thus, the focus here is exclusively on *business process testing*, which verifies that processes actually execute as expected and create output as expected. Business process testing is not surprisingly resource-intensive, especially when you consider the amount of time consumed in recoding and retesting your business processes after you've gone through the first wave of testing. Testing teams must be assembled and given direction, and everyone must understand that they are engaged in an iterative and very detailed process, not a one-time exercise.

Other types of testing focus more on system-level performance and overall mySAP.com load or *stress testing* (the terms are used interchangeably), tasked with ensuring that these same business processes execute as rapidly as possible

IN THIS CHAPTER

- Testing SAP Business Processes
- Three Types of Business Process Testing
- How to Approach Business Process Testing
- Executing Business-Process-Specific Testing
- Tools and Techniques

under loads similar to what the production environment will eventually endure. And somewhere between integration testing and stress testing lies *volume testing*, the purpose of which is to understand the capacity requirements (including elapsed time) of a system. For example, prior to Go-Live many companies execute a large volume of transactions such as might occur during month-end closing; in this way, performance limitations are better understood. Both volume and stress testing lead to improved planning and scheduling of resource-intensive system events as well, allowing an organization to set expectations with the system's end-user community. Volume and stress testing are covered in detail throughout Chapter 16.

Introduction to CATT and eCATT

SAP components include *test organizer* capabilities that let you plan for and manage the various types of testing that take place throughout a deployment. Instead of relying solely on the test organizer, I prefer to leverage a detailed master project plan that allows for managing the big picture. In this way, the test organizer is relegated to simply organizing and managing *test cases*, which are basically documented business processes complete with input, processing, and expected output.

The test organizer historically leveraged in SAP implementations is referred to as the *Computer-Aided Test Tool (CATT)*. CATT supports recording and documenting test cases, generating input data, and executing these test cases, even to the point of doing so in an automated fashion. CATT provides for capturing and creating output data as well, in the form of straightforward log reports that facilitate low-level test results examination. CATT has been around since R/3 Release 3.0 and is not only free, but also has the advantage of being tightly integrated with the system. And it's fast—by using Batch Input technology, test cases are executed directly on the application server rather than relying on the ability to drive the SAPGUI interface. Furthermore, CATT allows testers to incorporate extensive behind-the scenes checks into their test cases, based on access to data simply not available to the SAPGUI (like verifying the contents of a database table, or performing a field check to verify customizing settings). CATT has its limitations though, including

- Limited support for external applications that might play a role in an organization's extended enterprise solution; CATT cannot support user interfaces outside of the SAPGUI for Windows or the JavaGUI.
- Limited support for distributed mySAP.com system landscapes, resulting in the need to use RFC destinations in your scripts (or to build multiple test cases across your mySAP enterprise to simulate an end-to-end business process).
- Limited function module support; calendar functions are available, for example, but there is no function module support for tabular parameters.

- Controls and other technology (like tree controls and list viewers) found in more recent releases of mySAP.com and the EnjoySAP SAPGUI are not supported, thereby severely limiting the “testability” of certain business processes.
- Limited ability to execute testing remotely; it’s possible (again, through RFC calls to SAP systems, or multiple test cases otherwise) but represents a lot of work in terms of script development and management.

To address these shortcomings, SAP developed an even more powerful and capable integrated testing tool, as you see in Figure 15.1. Referred to as eCATT, or *extended CATT*, this highly refined tool was introduced with Web Application Server 6.20, and supports testing any mySAP.com solution running on basis layer 4.6C or greater (refer to SAP Note 519858 for current details regarding any support packages that might be required). Support for older basis releases is planned as well.

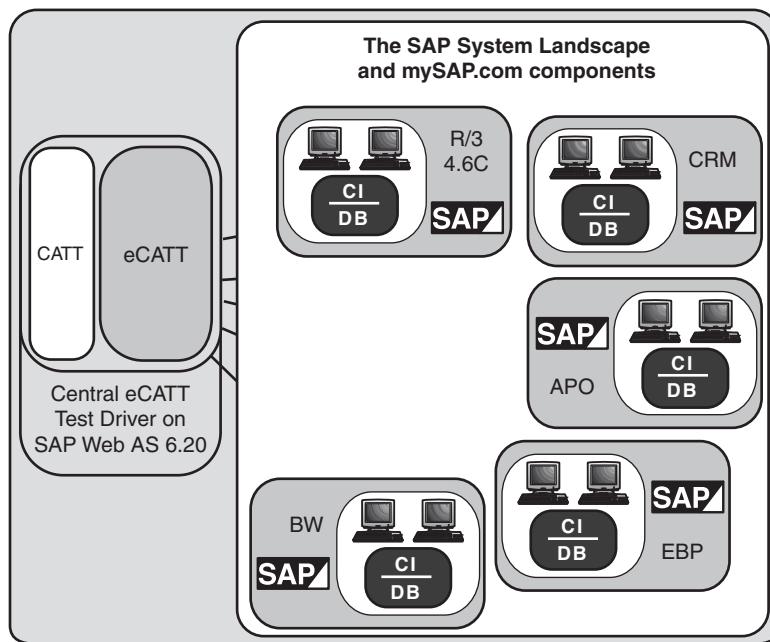


FIGURE 15.1 The value of eCATT as an enterprise testing solution is apparent in the variety of mySAP release platforms and components supported.

Throughout this chapter, I will be referring to eCATT unless another tool or approach is explicitly noted. And again, keep in mind that when a business process

executes over and over again flawlessly, a different type of testing is called for—stress testing. In this case, testing is taken to the next level, and specific business processes are executed concurrently by many users, in what is also referred to as a stress test or load test (a close relative to *Capacity Planning*). In this way, the unit-level activities tested to this point are further refined and proven to ensure that the Production system is truly capable of hosting the number and diversity of users and business processes required by the business.

- To read more about stress or load testing from a high-level perspective, see “The Goals of Stress Testing,” p. 571 in Chapter 16.

Before we examine *how* to approach and actually execute various types of testing, let’s step back and take a closer look at the three broad types of business process testing first.

Three Types of Business Process Testing

Different goals exist in regard to testing business processes. In the most generic sense, *functional testing* represents any business-process-specific testing executed to ensure that the business process *works*. In other words, every applicable option, every drop-down menu selection, every radio button, and so on needs to be tested initially, and then retested after every change.

Integration testing takes functional testing to the next level, ensuring that an organization’s business processes work together with other business processes, both within and across multiple functional areas, such that the SAP system as a whole operates as expected. Thus, much of the work prior to Go-Live falls under the category of integration testing, to the point where it is quite common to maintain a dedicated system within the SAP system landscape for integration testing. Some organizations refer to this system as the QA, or Consolidation, or Test System. Others call it simply Integration; I like the term Test/QA. Regardless, the goal of this core landscape component is to allow for end-to-end mySAP solution testing outside of the primary development system, prior to deploying the business processes to be used productively by end users.

Finally, *regression testing* allows for quickly proving that a specific set of data and processes yields consistent and repeatable results, even when a subset of that data or process changes. Thus, regression testing is all about testing the integrity of a given business process, regardless of the minutiae related to specific changing business logic/coding and programming practices. And regression testing ensures that currently implemented stable business processes continue to work after changes to other business processes are made.

As you see in Figure 15.2, all three types of business process testing complement one another, and aid in creating and maintaining a functionally useful SAP environment.

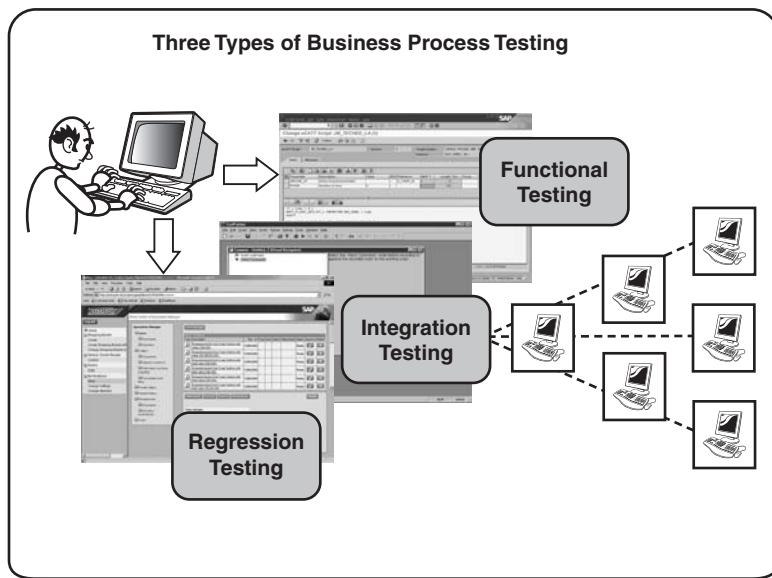


FIGURE 15.2 Clearly, all three types of business process testing play a part in ensuring that your mySAP solution is truly productive both at Go-Live and throughout the solution's life cycle.

In all three cases, business processes are typically scripted into repeatable business cases, and documented to some extent. Functional tests tend to create the most documentation in terms of volume, as every case must be covered. Integration tests are fewer, and therefore drive less raw documentation. Regression tests in my experience tend to be more pointed (very discrete or specific test cases, like creating a sales order for a specific sales organization), such that the documentation requirements are very detailed but leaner *per test scenario* than in functional testing. Thus, a huge number of specific regression and functional tests are created, perhaps many tens or hundreds of times more than those related to integration testing. Given this relationship, it is nearly always mandated that one set of scripted business scenarios be used for both functional and regression testing, and another set used for end-to-end complex integration scenarios.

When to Execute Business Process Testing

There is no general scheduling rule when it comes to testing the functionality of your mySAP solution, other than to say simply that testing never truly ends. True, much of it is performed after the initial configuration is drafted in the Development environment and subsequently promoted into a QA or Consolidation or Integration environment. But if you ask any ABAP or functional consultant engaged with

deploying SAP whether their work ends on the day of Go-Live, you'll get a quick "not likely." Why? Because it's not likely that a particular solution will NOT be further refined, or process extended, or feature added or otherwise modified over time. And preparation for full-fledged SAP upgrades demands even *more* functional and integration testing, especially if you're unfortunate enough to have implemented custom routines or user exits. Bottom line, as long as *changes* are made to the SAP solution, some level of functional, integration, and regression testing will certainly be required as well.

I tend to see functional testing play a significant role in the entire life cycle of an SAP solution, therefore, until the solution is retired. And because the functional testing continues throughout the life cycle, so too must the integration and regression testing. Sure, the quantity of work tends to slow down somewhat after Go-Live. But it picks up just as quickly when a new business group is added to the SAP system, or when a new plant or distribution center is brought online, or a new product line is released. And it picks up noticeably as legacy and other integration points are brought in or changed, or new functional areas are added to the implementation. And significant changes arise from full-scale SAP upgrades, migrations, or corporate mergers, or when a new mySAP.com component is added to the SAP system landscape—these and other such activities potentially touch countless business processes, and therefore beg to be thoroughly tested prior to deploying updated processes to Production.

In addition to the really "big" changes just noted, other less obvious SAP business process changes like the following drive functional, integration, and regression testing:

- Adding or modifying custom reports
- Adding new output mechanisms, from electronic (via workflow and similar methods) to fax machines and printers
- Legal changes, especially those related to tax laws and so on
- System or business-process enhancements, and other similar development activities designed to refine a specific business process
- Various bug fixes and other system-wide updates designed to avoid known or potential issues
- Transitioning processes facilitated by older SAP transaction (like ME21) to newer equivalent transactions (like ME21N) that offer expanded functionality through the use of controls like drop-down list boxes and tree controls.

Again, any change to the system justifies retesting the affected business processes prior to redeploying the updated process to Production. Just how this is accomplished is covered next.

The Critical Nature of Functional Testing

Regardless of the enterprise application or software package, the attention given to functional testing directly impacts whether the end product—a productive SAP solution in this case—is successful or worthwhile. In the increasingly complex world of mySAP.com, this has never been more true. SAP continues to add new functionality to mature products like R/3 and BW, and in the last year alone has added completely new offerings like PLM and the SAP Exchange Infrastructure. Combined with the rest of the mySAP solution offerings and underpinning technology components, the opportunity to tightly integrate business processes across diverse functional and productivity areas has never been better. Similarly, the opportunity to really fail has never been as great, either. So it's safe to say that the primary roles of functional and integration testing are more critical than ever.

The ASAP methodology and its newer counterparts ValueSAP/Global Roadmap and especially the *SAP Solution Manager for Implementation* go a long way toward generally guiding an SAP test team in their endeavors. But there are a few key areas that remain truly pivotal:

- The leadership of the team tasked with functional testing
- The general aptitude or ability of the programming and business process staff when it comes to testing
- The specific experience each tester has in their business area of focus, such as enterprise controlling or asset management or procurement and fulfillment
- The specific experience each tester brings to the table in terms of other mySAP.com projects in which he has participated from a testing perspective

Experience is *huge*, as should be evident in the preceding list. One reason for this is that very little formal education is offered that is centered on testing methodologies or building test cases. The training that SAP programming and functional folks receive most often amounts to the experience that each person gains during their paid SAP projects or engagements! Very few functional or SAP basis courses devote more than a few cursory minutes to functional, integration, or regression testing.

Short of complete failure, what is the risk of inadequately testing a mySAP solution? In the best of cases, it is simply that a much less refined system is ultimately offered up as “production.” Worst case, though, business processes fail, and key areas of customer satisfaction, manufacturing, human resource management, and so on are put at risk. To address this, consultants like myself and even more so my programming colleagues are then pulled into “post-production” support roles. In some cases, my programming colleagues seem to never even leave their client site after Go-Live, transitioning into post-Go-Live roles while our SAP customers continue to pay for further development and more testing of an end-product that in all reality fell short of everyone’s expectations. To net it out, then, ignoring or compressing the testing

phase in an SAP project can be phenomenally expensive to everyone, affecting not only the project budget but the business's ability to execute as well.

The Real Value in Integration Testing

When 95% of the coding and configuration (“programming” in the most general sense) has been completed within each mySAP component, work can begin in earnest testing how all components integrate together to create an enterprise solution. Integration testing must continue to focus *inside* each mySAP component, too, to ensure that business processes execute as expected, but *outside* is where most of the work lies. That is, SAP is integrated such that back in the functional testing phase, it was largely validated that SD interfaced well and worked as expected with MM, PP, and FI (for example). But in integration testing, what started as a suite of functional test cases was broadened to include an even larger scope of test cases. Thus, I contend that integration testing will not only always overlap functional testing, but add to it substantially.

During this phase in the project, all team members work together to run through the various scripted and otherwise recorded business processes, noting failures, successes, and unexpected variations or outcomes. Again and again, throughout the testing phase, each team comes together to test their configuration changes. The risks are many:

- In a perfect world, it would be ideal to start integration testing after the entire solution has been completely configured at a component level. Normally, though, the teams are fortunate to be 95% of the way there.
- After each set of failed integration tests, the team must go back to their respective drawing boards and address any shortcomings, variations, and errors in their code. The opportunity to introduce new errors into business processes that have performed flawlessly up to this point is possible, however.
- The teams must be kept in check against the project scope. It's very easy for an ambitious developer to begin adding nice-to-haves or otherwise extra features. This feature-creep only adds complexity, though. And it further complicates integration testing as the Go-Live deadline looms closer and closer.

The next section covers business process testing that most often takes place *after* Go-Live, although a good bit of it can still occur prior to that big day—regression testing.

Regression Testing

Testing how well a business process works after a change to the system is performed is the goal of regression testing. This helps to ensure that your changes (or the addition of new functionality, which in itself represents a change, too) do not have a

negative effect on business processes already in production or earmarked to be released to production soon. In my experience, I tend to see regression testing activities spike most during migration testing or when a new mySAP.com component is added to and integrated with an organization's enterprise. And it's not uncommon prior to Go-Live, too, when an organization is going through dreaded scope changes or succumbing to other alterations to the master plan. Overall, though, the bulk of regression testing occurs at the following times:

- After Go-Live, when new functionality is added to the currently implemented SAP components or products
- After Go-Live, when new components or products are integrated with the SAP enterprise
- After Go-Live, during the course of normal system maintenance after applying support packages, legal changes, and other patches/fixes
- After Go-Live, in support of planned migrations to new versions or releases of mySAP components

This type of testing, like functional and integration testing, takes place on the SAP instance implemented for testing—Test/QA, Integration, Consolidation, or whatever you call it in your environment. Keep in mind, of course, that the actual changes are made in Development, and then promoted to Test/QA. After these changes are tested and proven here, they are eventually promoted from Development to the Production instance.

Functional Versus Stress Testing

Whereas functional and other such testing is geared toward ensuring that a business process *works*, stress testing or load testing ensures that a business process *works quickly*. That is, functional testing focuses on the correctness of implementation, and stress testing assumes this correctness in order to focus on how the system behaves during daily and high-load periods. In the real world, after the bulk of the configuration work has been locked down, more attention is paid to functional and integration testing than is ever paid to load testing. A few months before Go-Live, however, a comprehensive systems and stress test *should* be executed. I take a closer look at this level of testing in the next chapter.

How to Approach Business Process Testing

Planning for, executing, and analyzing the results of business process testing is a detail-oriented job, to say the least. And even at its best, it is still phenomenally time-consuming. Historically, much testing has been performed manually. Drawbacks to this approach, as you see in Figure 15.3, are fairly obvious though. For example:

- Each tester recruited or otherwise brought in from the business must be able to devote an adequate amount of time to testing. This in turn takes away from the tester's other duties within his organization.
- Testers brought in from third-party consulting and integration partners are probably not experts in your particular business processes—relevant industry experience is therefore a must. Further, they are by no means cheap (though this is relative).
- Testers must be trained in the tools and methodology employed by your team for testing, not to mention the method of documenting output and other results such that the data collected is captured and presented in a clear, consistent manner.
- These testers need physical resources—office/lab space or a “war room” to easily facilitate coordinated testing, complete with enough physical client devices (desktops, laptops, PDAs, and any other access devices eventually to be used on the productive system) to actually perform the testing.
- Finally, because these expensive human resources operate in “real time” only, the ability to consistently and repeatedly execute business processes is impacted. That is, testers must be present to execute a test, and they must be lucid and otherwise at the top of their game *all day long*.

Drawbacks to Manual Testing

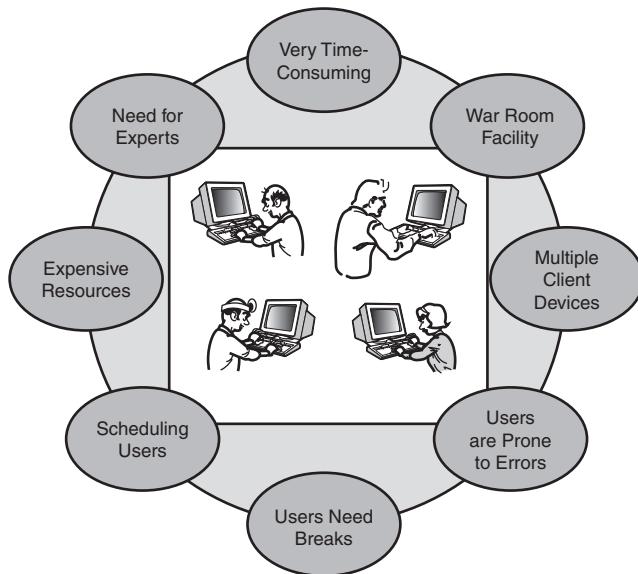


FIGURE 15.3 Testing business processes using manual approaches is subject to many drawbacks, compared to automated approaches.

It is because of the drawbacks shown in the preceding list that automated testing tools for SAP have really grown in popularity over the last six or seven years. Tools like SAP CATT, SAP eCATT, AutoTester ONE, Compuware TestPartner, Mercury Interactive QuickTest, and quite a few others solve the dilemmas described in the preceding list in that they

- Require fewer expensive human resources
- Can execute the same tests over and over again without getting bored and making mistakes, or needing a coffee break, and so on
- Can consistently execute a test much more rapidly than their manual-approach counterparts
- Can be easily modified to include/test a set of variants or other values unique to similar test cases (such as testing the creation of a sales order for many different sales groups or distribution centers)
- Can act as a single repository for all test cases and related documentation, thus facilitating management of the testing process and key deliverables—CATT, AutoTester ONE, TestPartner, and QuickTest are more limited in this respect than eCATT, though, as you see in the next section.

Thus, compared to manual testing methods, the investment in automated testing tools to enable script development consistently pays for itself quickly. Even in the case of third-party tools (where license fees might range from tens of thousands of dollars to a hundred thousand dollars and more), the return on investment is still typically rapid—less than a few months.

Third-Party Tools and Other Resources

Although AutoTester's products were the first to be certified with SAP, Compuware's TestPartner has the special distinction of being the first (and as of this writing, only) product to be certified to run with eCATT. If you're interested in TestPartner or eCATT's interface for third-party products, please check out the Planning CD, especially the PDF published by SAP AG entitled "BC-eCATT Interface Description - Test Tool Integration into SAP eCATT."

In my own experience, though, I've seen a lot of different testing tools outside of SAP-approved tools that have been used to drive business process testing. The most basic of these are simple scripting utilities adept at propelling the SAPGUI or WebGUI simply by virtue of being Windows products. More sophisticated products exist, too, as do custom scripting approaches. Tools or approaches not already mentioned include

- AutoTester Client/Server
- Mercury Interactive WinRunner

- HiddenSoft AutoIT
- Custom Perl scripts
- Custom Visual Test scripts
- Segue SilkPerformer

Most of these represent older offerings, or embody little in the way of SAP-specific capabilities. This is where SAP's eCATT represents a quantum leap forward, in that it can handily address integration testing within the SAP framework.

SAP eCATT Differentiators

SAP eCATT's ability to drive business process scripts on newer mySAP.com solutions that leverage the SAP Control Framework, as well as the ability to work through the legacy SAPGUI API and interface to external tools, makes it my favorite automated testing tool of choice for pure business process testing.

To access all of this value, Web Application Server (Web AS) version 6.20 is required at minimum, along with SAPGUI version 6.20, of course. The price is right, however, as eCATT ships "included" with Web AS. Additional key benefits of eCATT include

- Support for remote centralized testing, insofar as being able to drive tests and act as the single test platform for many mySAP products, thereby simplifying data and script management.
- Support for all mySAP.com components running on SAP Basis Release 4.6C or greater; the core testing tool must reside on Web AS 6.20, but it can test a greater suite of SAP products.
- It can be integrated with external testing tools (like those discussed in more detail in Chapter 16) that may already be used by a particular organization or necessary to support testing business processes that touch non-SAP systems.
- Full integration with the Test Workbench.
- Tight integration with transaction ST30, the Performance Analysis Tool.
- It supports the reuse of test data, by storing this data in separate container objects.
- Support for various GUI interfaces, as seen in Figure 15.4.
- The *TCD command* allows you to test business processes using the proven batch input technology found in earlier releases of CATT.

- The SAPGUI command provides a new flexible test option for transactions that cannot be tested via TCD; precisely, it's the SAPGUI command that supports the SAP Control Framework discussed previously.
- Older CATT scripts (called test procedures) can be simply migrated to newer eCATT test scripts—note that a bit of script tweaking is still required, though, to reflect changes to RFC destinations (created with transaction SM59) and other details that enable eCATT to run tests across multiple SAP instances.

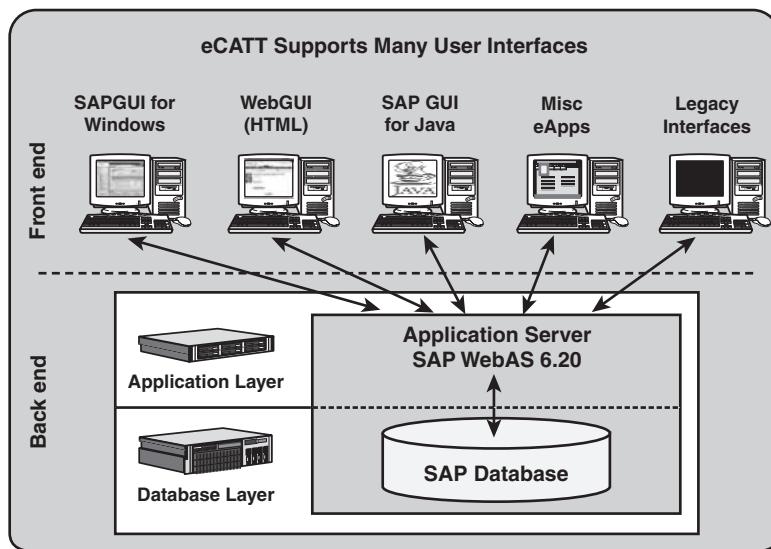


FIGURE 15.4 Consistent with its wide enterprise reach, eCATT supports multiple SAP graphical user interfaces.

The first point in the preceding list is especially exciting, in that eCATT allows end-to-end mySAP.com solution testing to be executed and managed from a single Web AS 6.20 or newer instance. Figure 15.5 illustrates this well—in the past, test cases usually had to be built and run from each SAP component. That is, every test system within the BW landscape, CRM landscape, SRM landscape, and so on became the de facto testing platform/repository for its component. This complicated business process testing quite a bit, and forced a lot of duplication of effort when it came to maintaining test cases where business processes crossed over to other mySAP.com components or different enterprise applications altogether.

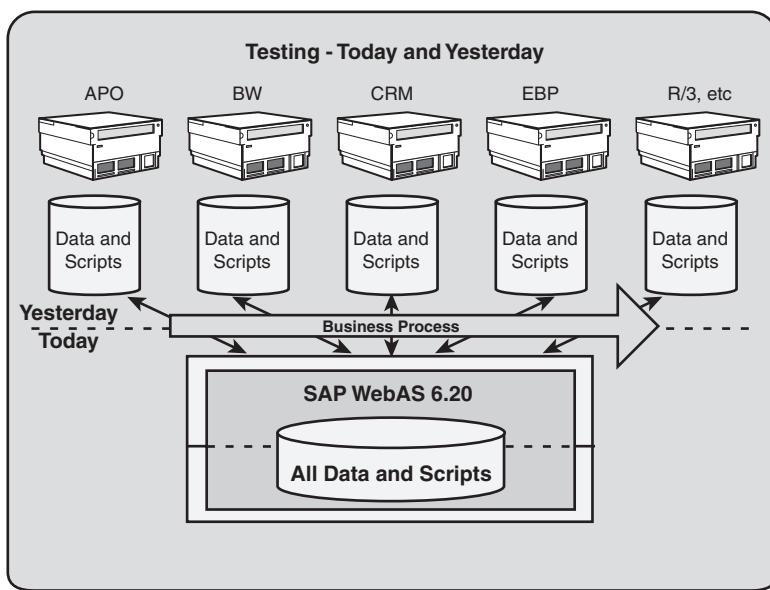


FIGURE 15.5 In the past, each SAP product or component required its own set of test cases, data, scripts, and so on; however, today eCATT facilitates centralizing test cases and data.

mySAP.com Landscape Considerations

As I indicated earlier, managing and executing test cases from each and every test instance within a particular mySAP.com component's system landscape was the norm until the arrival of eCATT. Today, eCATT lets you simplify functional, integration, and regression testing. But eCATT alone does not allow you to minimize the number of servers or instances in your testing environment. As you see in Figure 15.6, the number of development, test, and production platforms in a typical mySAP.com system landscape remains unchanged.

Other SAP initiatives, like *Multiple Components, One Database* (MCOD), will serve to reduce the number of test or development *database servers* within a particular testing environment. As you can clearly see in Figure 15.7, all things being equal, the number of SAP instances will still remain unchanged, though—testing and the need for test instances exists regardless of how the database for a particular test system is accommodated.

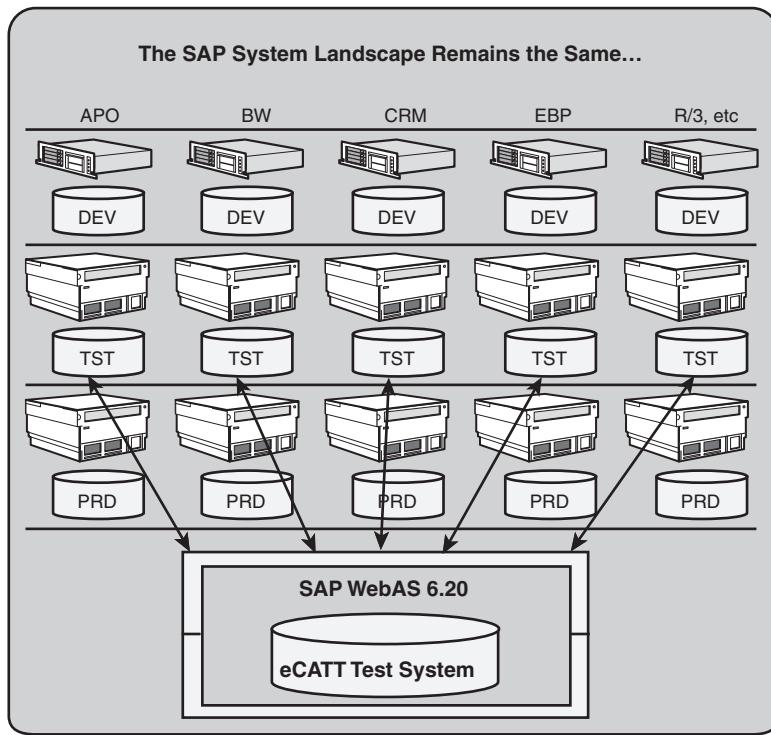


FIGURE 15.6 Clearly, eCATT does nothing to minimize SAP system landscape requirements.

Other test-related areas of importance are impacted too, and are examined next.

Additional People Considerations

With the kind of automated testing I have discussed thus far, there is a significant potential reduction in human testing resources—people—that can be realized:

- Test execution specialists will be reduced—as you saw earlier in this chapter, the number of testers is reduced when an automated tool approach is taken.
- Fewer test-case developers and maintainers are needed. However, the people tasked with these responsibilities must understand how their core business processes impact and otherwise touch other mySAP.com components. So, although fewer people may be required, they must be highly qualified across the board—in their business area, in testing, and in general.
- The need for testing coordinators is similarly reduced, given the reduction in people needed overall.

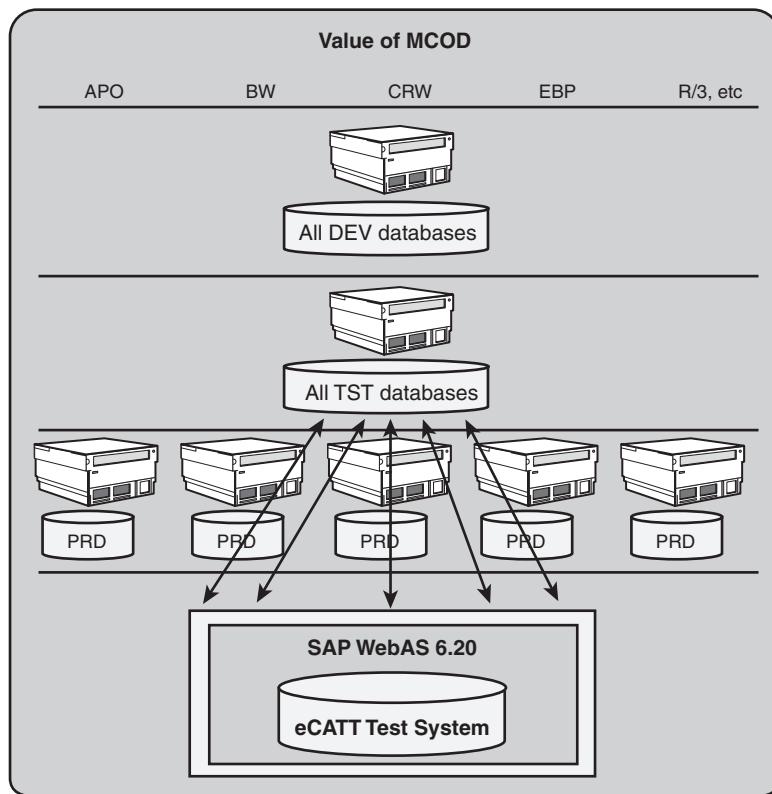


FIGURE 15.7 The MCOD initiative can reduce the number of database servers required in a test environment, but it will not change the physical number of databases.

However, the number of folks responsible for identifying and fine-tuning the business processes remains pretty consistent regardless of the specific testing approach (manual or automated). Elsewhere in the SAP Technical Support Organization, head-count remains the same, too.

Process Constraints and Issues

Testing includes much more than just testing valid data or combinations of data. Rather, a good testing process also embraces what SAP refers to as *negative testing*, where test cases are created with invalid data. The goal of these particular cases is to determine how well the system recovers or handles incorrect data, and how well the system communicates this fact back to the end user. For example, test cases should

be created and tested where invalid customer and material numbers are fed to sales order processes, to ensure that error handling and general feedback is both present and appropriate.

Sound testing also includes experimenting with additional boundary testing as well as all combinations of client and user interfaces—a comprehensive testing process requires true end-to-end execution and analysis. Thus, if the Java-based SAPGUI will exist in your environment, and a particular version of the classic SAPGUI is required to support long-time SAP R/3 users, testing must be performed that includes both user interfaces.

Other Areas of Impact

Another area impacted by automated testing regards the number of physical desktops (or other client access devices) needed to perform the testing. Because fewer people are required, fewer SAPGUI, WebGUI, and other access devices used to execute these interfaces are needed as well.

In addition, because scripted test cases are inherently easy to duplicate, fewer test runs are required. This is because every test run executes precisely the same steps, in precisely the same order, as every other test. Omissions, errors, timing inconsistencies, and so on are virtually eliminated with automated testing. By reducing the number of test runs required, we reduce the number of billable hours that consultants will charge a client, and free up valuable hardware resources for other tasks.

Finally, as I said previously, a good automated testing approach should not be limited to the tools you currently have on hand. On the contrary, it is always preferable to use a tool that can be “extended” to include third-party testing/scripting tools, to support testing cross-platform, or to allow for testing complex enterprise-wide business processes. As you can see in Figure 15.8, SAP’s eCATT offering is perfectly positioned to address this, too. It allows third-party vendors and software developers to integrate their test tools using the BC-eCATT interface—for more information on certified interfaces like BC-eCATT, see
<http://www.sap.com/partners/icc/scenarios/technology/>.

Applications that cannot otherwise be tested exclusively with eCATT can leverage this extensibility, thereby opening the doors to testing applications written in HTML, Visual Basic, and so on. With eCATT handling the transport of these additional third-party scripts throughout your mySAP.com test landscape (actually, the SAP Change and Transport System performs the actual transports), your centrally managed eCATT scripts can call scripts created by your third-party tools, and then execute them anywhere in the system.

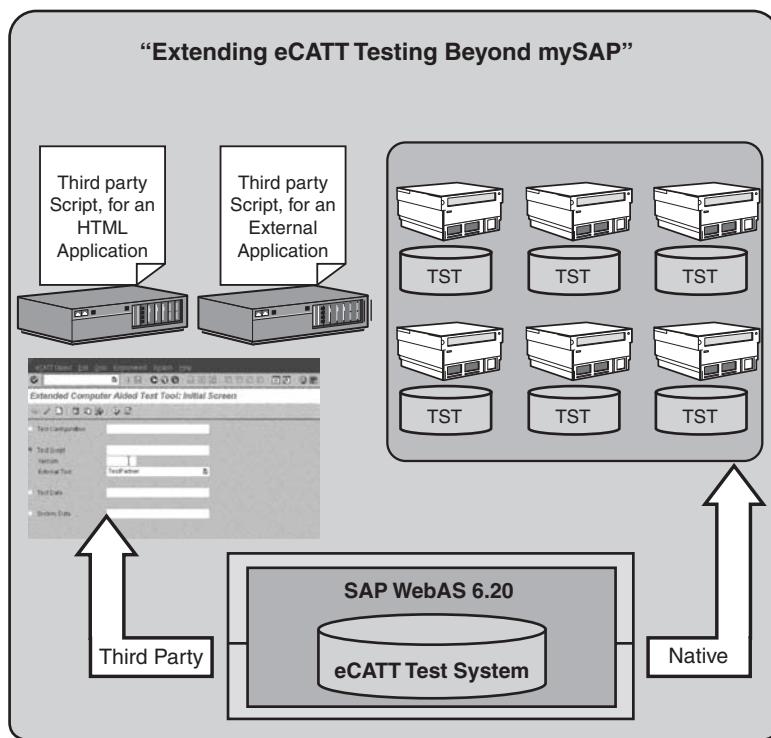


FIGURE 15.8 Note how eCATT can be leveraged to test business processes across multiple mySAP components as well as third-party enterprise applications.

Executing Business-Process-Specific Testing

At this point in your SAP project implementation, all but the Production system is probably deployed. Before testing can commence in the testing environment (or across multiple environments, depending on your situation and the specific business processes being implemented), minimum SAP Release levels need to be in place. Ensure that your central test system—the system from which all eCATT testing will be managed and driven—is at Release 6.20 or higher. Other systems, like CRM and EBP and so on, may remain at their current Releases. In these cases, though, check with SAP to verify that any minimum Support Package requirements are met. As of this writing, 4.6C Basis Releases are supported, with support expected to be provided down to Release 4.6B soon.

I further assume that various teams have also been assembled at this point, each geared toward specific functional areas or mySAP.com components. The teams

consist of a team lead, and both functional folks and developers working with ABAP, Java, and any other coding or programming languages used in the enterprise. In this way, each team is well-positioned to capture the details surrounding a business process, and create the necessary test cases using eCATT. This is accomplished via the Test Workbench, covered next.

The Test Workbench

As mentioned before, eCATT is fully integrated with the Test Workbench. This allows the team leader for a specific testing team to:

- Assign test configurations to individual testers
- Include test configurations in test catalogs
- Monitor test execution and results

Creating and executing eCATT scripts is straightforward. However, if your particular enterprise needs to test processes outside the scope of eCATT, I suggest that you consider “wrapping” your third-party external scripts in an eCATT script, and then storing them in the SAP eCATT testing platform database. This saves in terms of administrative overhead, complexity, and time spent in managing the third-party scripts.

To “wrap” your third-party scripts, you must first load the third-party tool on the desktop from which you run the SAPGUI that accesses eCATT. This is because when you click on the Script button in the SAPGUI, it will actually start up the third-party tool and bring up the script you want to edit or maintain. Incidentally, the Script button also allows you to create a new script using your third-party tool.

Prior to executing test cases, create a test matrix where all test cases are documented, and where the results of test runs can be stored and analyzed. I like to use an Excel workbook for this purpose because it allows me to track multiple test areas—I separate each area by worksheet, labeling the worksheet’s tab to make this clear. Plus, this approach allows me to insert input files, output data, screenshots with error messages, a copy of a particularly troublesome script, or other objects that might be useful when sharing or discussing the testing process with others.

Last of all, take advantage of the eCATT system data containers to allow simple administration of the various mySAP.com destinations that come into play when working with a central test system. These data containers eliminate

- The need to administer essentially the same test scripts in several different systems
- “Implicit” remote calls

And remember, leverage the ability of eCATT to maintain and manage all scripts in a single repository. Avoid the temptation to copy scripts and test cases everywhere.

Data to Track During Test Execution

During each functional, integration, or regression test run, note the date and start times, test run name, run number, version of script executed, version of input data, GUI details, and whether the business process completed successfully, or how far it progressed before it failed. I like to “tier” integration and regression testing as follows, to help me document and track where failures occur:

- Within the specific mySAP.com component and functional area (for example R/3 SD)
- Within the mySAP.com component (that is, the process completed everything it needed to do within and across the various R/3 functional areas)
- In other mySAP.com components (for example, the process successfully provided or pulled data to/from another mySAP.com component, like procurement replenishment data shared between R/3 and EBP)
- In non-SAP applications (to determine, for example, that the process successfully updated an HTML page or a data field on a system outside of SAP)

Maintaining a tiered approach like this helps focus additional development and further testing activity where it is needed most. When a test case has finally completed (whether successfully or not), additional data needs to be collected, and other processes need to be addressed. These post-processing tasks are covered next.

Post-Execution Tasks

After each test run, perform the following post-execution tasks as soon as possible:

- Note the output results, such as whether an order number was created and the actual order number itself. In this way, the order can be reviewed in detail at a later time if necessary.
- Note any SAP warnings or errors displayed by the SAPGUI.
- In support of “version control,” make a backup of any data, scripts, or test cases that have not been backed up yet.
- Note basic performance metrics (like end-to-end “wall clock” execution time, which is simply the time it takes to run through a complete business process).

At this stage in testing, achieving excellent performance is not the goal. But really poor performance might drive further business process development down different

avenues. And basic performance data serves as a baseline of sorts to share with the folks who will eventually perform any stress testing later in the project.

Compressing the Testing Phase in the Real World

At one of SAP's largest customer sites, the client really had a good handle on how to tackle and execute test cases. There was heavy involvement by the various business organizations, resulting in accurate detailed test cases of all transactions. They spent a lot of time getting their input data in order, too, and truly covered all business processes and their unique variations in great detail. However, for timeline reasons, they glossed over what turned out to be two key areas:

- Testing the reporting function, including system generated, ad hoc, standard or default reporting, and so on
- Training their end users to understand the reports they actually created

Instead, nearly all the testing was focused on identifying and addressing traditional business processes, specific variants or one-offs, and other highly unique test cases. The rest of their project plan time dedicated to testing was actually consumed by last-minute data loads and similar updates. As this customer told me later, SAP is terrific when it comes to easily drilling down into the details surrounding a specific transaction, and it's great at presenting the big picture. But getting a view of the "middle ground" via custom reporting takes some work, and this work was never thoroughly tested or shared with others.

They ultimately got all this information they needed, but it was sacrificed unknowingly up front just to get the SAP system in place. What does this have to do with business process testing? Simple—a business process is not complete, and therefore not completely tested, until it affords value to the organization that depends upon the business process. For this customer, the fact that the business process worked was nice, but because they could not really leverage the information gleaned from the business process, it was actually of little value until many months after Go-Live.

Using Testing to Support SLAs in the Real World

Although regression testing is geared toward proving that a business process works or continues to work after a change, I helped one of my customers leverage the data we collected to build a case for establishing minimum service-level agreements (SLAs) and then monitoring adherence to these SLAs. That is, we used the business processes—the workflow surrounding end-to-end business flows—to build timelines that described and supported different business functions in terms of minimum timeframes that could be tolerated between events. These timelines served as the foundation for SLA baselines because we knew the typical time required to execute a business process under varying system loads.

I later took this same approach and applied it to the benefit of another customer's Basis/Technical Infrastructure team. Scripted "basis business processes" (in this case, CCMS transactions used to monitor the system) were measured against typical response times. Thus, if it took more time than usual to execute a transaction to display all active users, for example, a threshold would be exceeded. This would in turn automatically trigger a message in their problem escalation system, and in doing so notify the senior Basis team, the help desk, and SAP Operations. Similar transactions comparing response times, buffer quality, and so on triggered similar messages. All of this became quite useful to the entire SAP TSO after a change control wave took place; monitoring the system in this way provided a "QA" check beyond the scope of basic regression testing, and kept everyone apprised of exactly how well the change control package performed in the real world.

The Weakest Link

SAP AG touts its products as business products configured by business consultants. True, the developers who perform much of the functional configuration are often experts in the business first, and programmers/developers second. But this can work against a project team if a balance of both functional experts and programming experts is not maintained. Consider any number of typical business processes where the base SAP configuration is simply not robust or flexible enough for a particular company. The development team gets around this shortcoming by coding a custom solution in ABAP, Java, or whatever is applicable. A development team consisting of too few real programmers has a decent chance of writing poor enough code that it becomes the solution's weakest link. Sloppy ABAP not only frustrates performance tuners, but also exasperates upgrade attempts later on. And it certainly plays a role in complicating regression testing throughout the life of the solution.

Tools and Techniques

I recommend getting your hands on eCATT as soon as feasible (again, it ships with any product that sits on top of Web AS 6.20 or greater). It's a great tool, and really simplifies business process testing in general. Unfortunately, the list of supported third-party tools is currently quite limited—Compuware's excellent TestPartner product is currently the only certified third-party testing tool. However, I expect the other big players in the integration and load testing market to join Compuware soon. Check SAP's partner Web site often for updates:

<http://www.sap.com/partners/search.asp>.

And in the meantime, refer to the Planning CD for all of the figures found in this chapter plus a document titled "Compuware Introduces Functional Testing Support for SAP Web Application Server Technology," which is a copy of Compuware's press release. I've also included a recent white paper published by Compuware, which I

have retitled to “saptp - Compuware TestPartner and SAP eCATT.pdf” to be a bit more explanatory than the original title of “saptp.pdf.” I recommend that you point your browser to <http://www.sap.info/public/en/article.php4/Article-268593d92fa98cbdd2/en> to read an article published by SAP AG that discusses eCATT-enabled testing. Other information relevant to eCATT is available from <http://www.SAPinsider.com> (check their article archives). Finally, <http://help.sap.com> provides excellent information on the eCATT scripting language and additional details as well.

Summary

The focus of this chapter has been on functional, integration, and regression testing. Together, we looked at the value that an automated approach provides, especially in the realm of using SAP’s eCATT product, and I stepped through various benefits of using eCATT over other methods or products. I then wrapped up the chapter with an approach to testing, followed by real-world testing experiences. In the next chapter, I take business process testing to the next level, and spend time dissecting everything from hardware-level subsystem performance to end-to-end mySAP solution stress testing.

PART IV

Preparing the Production Environment

IN THIS PART

- 16 Systems and Stress Testing
- 17 Planning for SAP Go-Live

16

Systems and Stress Testing

Introduction—Preparing for Production Stress Testing

Finally, we have arrived at a point where Go-Live is just a few months away. Although ValueSAP’s Global Roadmap calls this the tail end of the Realization phase, the older ASAP methodology referred to this stage of an SAP project plan as “Final Preparation.” I like this term, as in my experience, systems and stress testing falls into the collection of final tasks performed in the name of Go-Live preparation. ASAP devotes very little attention to these key project milestones, however. So in the final two chapters of the book, I have taken the liberty of detailing what in my experience *really* needs to occur before the big day of SAP Go-Live, starting with end-to-end systems and stress testing here in Chapter 16.

Data Center and IT Project Managers have long had to develop processes and approaches that address the growth and performance of the computing environments for which they are responsible. Labeled benchmarking, system-level performance testing, load testing, and more, this stress testing (my favorite term) seeks to understand and identify the limits of an IT solution—what kind of performance can be expected day after day, and how the system will react when put under the stress of month-end processing or other weighty business processes.

For mySAP solutions, not only is the work of stress testing critical, but it is complex and not accomplished quickly. It’s hard work, plain and simple. Most anyone can bang together an SAP test system, come up with a guess as to

IN THIS CHAPTER

- Introduction—Preparing for Production Stress Testing
- System-Level Stress Testing and Pre-Tuning
- Key SAP Stress-Testing Considerations
- How to Test mySAP Components
- Script Development and Preparation in the Real World
- Stress Test Execution During “Test Week”
- Additional Stress-Testing Goals
- Extracting the Last Drop of Value out of Testing
- Lessons Learned in the Real World
- Tools and Techniques

what kind of load the system will be hosting on an average day, and get some users together to execute a few representative business processes. But to do a stress test justice, you need to consider the following:

- Specific goals that you need to achieve through stress testing. This includes more than identifying the peak loads you want to see; you may also want to test failover during the peak load, test how well your online backup performs, or play “what if” and throw another 500 active users into the mix, for example.
- Pre-tuning your SAP Solution Stack, to at least start off on the right foot.
- How to achieve those goals—what stress testing approaches to employ, methodologies to embrace, tools and expertise to bring to the table, and so on.
- Identifying the proper mix of mySAP components, functional areas, business transactions, users, batch processes, and more.
- Scripting or otherwise automating the ability to create a known load on your system.
- The process you will employ for “warming up” and then saving the database against which all test runs will be performed, and later restoring this known state; in this way, master and transactional data will be consistent at the beginning of each test run, supporting apples-to-apples comparisons.
- Monitoring and capturing statistics during test runs.
- Analyzing the data from your test runs, to ultimately drive “smart” solution stack tuning and configuration.

Stress testing is all about giving you the peace of mind you prayed would be yours prior to Go-Live, validating that your SAP technology partners did *their* jobs in terms of sizing and recommendations, and that the SAP TSO has done *their* job equally well from an installation and tuning perspective.

Thus, it strikes me as odd when I hear of SAP implementations that shelve the idea of stress testing to save a few dollars or man-hours. It has always been my thought that this last big phase in your implementation is too critical and just too risky to ignore. Before we dive into this chapter together, I'll let you in on a little secret—you can have the peace of mind that comes from stress testing for as little as \$35,000 to \$50,000. And that's not a “simulation” or generic benchmark test—I'm talking about a full-blown stress test, complete with SAP-aware testing software executing online users and batch jobs running *your* business processes against a copy of *your* actual pre-production database. I'll go into more detail later, but suffice it to say that with millions of dollars of budget money, tens of thousands of man-hours invested, and your core business processes at stake, an SAP implementation is just not complete without a stress test.

So here in this chapter, I will drill down into the different business and technical reasons why the pre-production system needs to undergo stress testing, and walk you through performing an end-to-end test. You will gain insight into the tools and approaches that can prove that your solution is ready for the real world, and I'll help you capture the statistics and other data you need to back up those claims of being prepared for Go-Live.

The Goals of Stress Testing

Why is stress testing valuable? In the simplest terms, your customers and end users have expectations of the SAP system to be implemented or upgraded. These expectations are translated into service-level agreements (SLAs) guaranteeing a certain level of performance and availability. From a performance perspective, things like average user response times, batch job processing requirements, standard and ad-hoc reporting, and so on will all typically be addressed by the SLA. SLAs not meeting their targets will result in penalties, either self-imposed (like an IT team not meeting its goals), explicitly negotiated (like fees levied against an SAP Application Service Provider), or as a result of poor customer service (like losing customers, partners, or vendors to competing companies). Typical stress tests seek to prove service levels associated with the following:

- Business transaction response times, or how long it takes to refresh your SAPGUI or WebGUI after pressing the Enter key, for example. You might even take this a step further, and indicate the need to process screens of a specific transaction within a specific amount of time.
- How quickly a background or “batch” job will execute. This is often measured in wall clock time, the time it takes from start to finish.
- Understand how well a system addresses its expected capacity or volume requirements. Performing a volume test will aid in the planning and scheduling of peak-load events, and serve to set end-user expectations to boot.
- How quickly a report or other query will make it through the system and actually be printed.
- Average speed in which database transactions will be processed and committed.

In a nutshell, your goals for stress testing usually revolve around proving that your SLAs can actually be met; that they're achievable. Back in the real world, I have helped countless companies structure and perform SAP stress tests for a lot of different reasons, and their goals varied accordingly:

- To prove that a new SAP technology component indeed performed as advertised, or as promised by the hardware or software vendor! In these cases, the stress test validated the sizing of the system.

- To prove that a new mySAP component behaved as expected, or could host a certain system load.
- To prove that a solution could *scale* to twice or even ten times the typical daily load expected.
- To prove that the solution could host a certain load after failover, or after failure of critical system components.
- Most of all, they wanted to mitigate a very controllable piece of risk associated with Go-Live.

Much of this can be summed up by saying that each SAP implementation project sought to prove that its unique SAP Solution Stack performed as expected, whether from a day-to-day perspective, during a month-end crunch, after failover to a disaster recovery site, or during any number of other inescapable situations.

Performance Baselines and Success Criteria

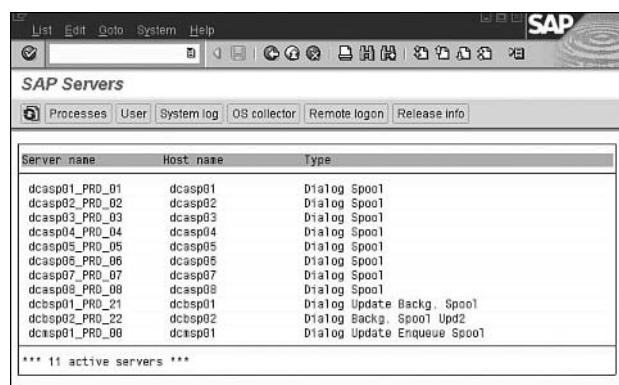
Even in brand-new SAP implementations, where no “legacy” SAP system is available for comparisons, a performance baseline of sorts still needs to be established. This baseline might be the response time seen in the Test/QA or Staging environment. Or perhaps it might represent the average throughput for a particular configuration observed in published benchmarks or other load testing exercises. Worst case, you might even jot down performance statistics related to the current legacy system. Regardless, though, you need a baseline for three reasons:

- To prove later that your SAP implementation delivers better online or batch performance than the system it replaced (assuming that “better” is indeed a goal; lower cost or improved availability might be the motivating factors in your particular case).
- To verify that the new solution performs as advertised. That is, a baseline might simply represent the “minimally acceptable worst-case” performance expected of each layer in the solution stack. A baseline therefore helps to determine whether or not a stress test was indeed successful.
- Finally, a baseline becomes a useful measuring stick after every major change wave. Many of my clients execute a small core set of transactions and basic hardware utilities after changes are made to the production system, to ensure that it performs at least as well as it did previously. In other cases, a baseline will also help you prove that a solution component upgrade or replacement indeed resulted in a faster system; the proof lies in the delta between your baseline statistics and the results of your updated testing after the upgrade.

Most often, your stress test success criteria amount to either performance observed by the end users of the system, or the amount of work completed in a specific time period. In many cases, both relative performance and hard throughput statistics are collected. The most common success criteria include achieving

- A specific number of users logged into the system, for example 1,000 (use transaction AL08 or ST07 to verify this number).
- A specific percentage of users logged in who are actually busy doing work. This is also referred to as *concurrent users* (use transaction ST07).
- Achieving a response-time target for specific functional areas. For example, a company might have purchased a system sized such that financial users would achieve an average of a half-second response time, and sales/distribution users might achieve an average response time of less than 1.5 seconds (use ST03 immediately following the stress test, and plug in the appropriate data collection time interval).
- A specific number of concurrent processes (that is, SAP dialog, batch, or update work processes being used simultaneously). In a system with five application servers, each configured for 20 dialog work processes, a stress test might seek to utilize 75% of these work processes and then observe average response times or throughput values (using transactions like SM66 or SM51—simply add up the number of “lines” displayed by the transaction code, and compare this to the number of configured work processes).
- A specific amount of work performed in a certain time period, or throughput. For example, a customer might want to process 10,000 five-line item sales orders in 15 minutes, or execute a certain suite of transactions in an hour, or process 20 custom reports in less than 30 minutes, and so on. I have found that this kind of data is most easily gathered by tracking the results of the scripted business processes (creation of output data, for example).
- Success factors might involve any of the preceding points, with the caveat that the system cannot exceed 65% average CPU utilization, or average disk queue lengths in excess of 10, or memory utilization that drives greater than 2% Pagefile consumption, for example.

How to quantify and measure your success criteria is discussed later. Suffice it to say, though, that transaction SM51 will underpin much of your testing, as displayed and explained in Figure 16.1—SM51 allows you to easily move around between different servers in your SAP system, noting and collecting relevant performance and other data as required.



The screenshot shows a SAP transaction interface titled "SAP Servers". The menu bar includes "File", "List", "Edit", "Goto", "System", and "Help". The toolbar has icons for search, refresh, and file operations. The main area is titled "SAP Servers" and contains tabs for "Processes", "User", "System log", "OS collector", "Remote logon", and "Release info". A table lists 11 active servers:

Server name	Host name	Type
dcasp01_PROD_01	dcasp01	Dialog Spool
dcasp02_PROD_02	dcasp02	Dialog Spool
dcasp03_PROD_03	dcasp03	Dialog Spool
dcasp04_PROD_04	dcasp04	Dialog Spool
dcasp05_PROD_05	dcasp05	Dialog Spool
dcasp06_PROD_06	dcasp06	Dialog Spool
dcasp07_PROD_07	dcasp07	Dialog Spool
dcasp08_PROD_08	dcasp08	Dialog Spool
dcbsp01_PROD_21	dcbsp01	Dialog Update Backg. Spool
dcbsp02_PROD_22	dcbsp02	Dialog Backg. Spool Upd2
dcmsp01_PROD_09	dcmsp01	Dialog Update Enqueue Spool

At the bottom of the table, a message reads: "*** 11 active servers ***"

FIGURE 16.1 CCMS transaction SM51 provides insight into the activity of each server; by double-clicking any active server, you are connected to it, and can then leverage that connection to collect application server-specific data.

Given all of this, arriving at a set of success criteria is no easy task. I recommend working with your various business groups, SAP TSO, SAP Solution Stack partners, and systems integration/consulting firms to develop success criteria that are reasonable, auditable, and applicable.

System-Level Stress Testing and Pre-Tuning

Although one of the goals of SAP stress testing is to guide and direct tuning of your soon-to-be-production system, it makes sense to engage in a certain level of *pre-tuning* as well. This involves testing and tweaking the individual solution stack layers and components to operate well in a standalone manner. When things run smoothly in this respect, you can then more intelligently optimize the entire technology stack to perform as a cohesive solution of integrated components and technologies.

Testing the individual components or layers in your particular SAP Solution Stack is also often referred to as system-level testing or systems testing, and will save you countless hours and much energy down the road. Key reasons for system-level testing include

- To establish a baseline for individual component-level system performance today, to be measured again as your change control processes upgrade these components or introduce new components or solution stack changes.
- To simply understand where potential performance bottlenecks might lie in the future. This is also called characterization testing or capacity testing.
- To therefore understand where future budget dollars can be best spent—where you get the most bang for the buck when you need your first performance

boost. For example, if your network infrastructure is capable of supporting 200% growth in the number of online SAP users, but your disk subsystem can only handle another 30% growth in online users before performance becomes unacceptable, this information will help drive intelligent upgrades.

As you have been accustomed to by now, I will look at system-level stress testing by leveraging the SAP Solution Stack model. Specifically, the next few sections will drill down into server hardware and OS, disk subsystems and databases, network infrastructure, and mySAP.com component testing.

Server Hardware and OS Testing Tools and Best Practices

Before you commence a full-blown stress test, I believe that it is important to verify that the server is optimized for its role in your SAP solution. For Web and other Application Servers, CPU and memory performance are key. In a few cases, I have helped my clients benchmark how well different server and memory hardware configurations perform.

However, server performance is greatly affected by memory configuration at an operating system level, too. Windows-based systems must be set to maximize data throughput for network applications, for example, as displayed in Figure 16.2. And in all cases, memory swapping or paging needs to be minimized. With this in mind, I have found a number of utilities that support server hardware and OS-level testing:

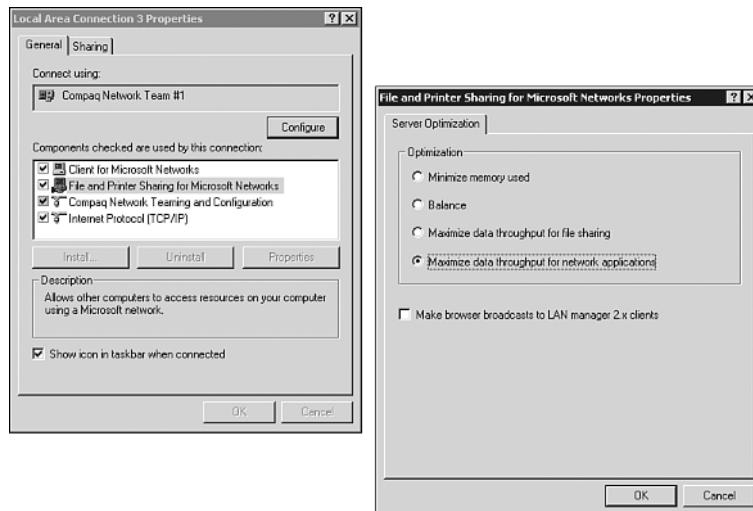


FIGURE 16.2 Memory configuration goes beyond simple hardware configuration; the amount of RAM made available to applications like SAP can be set or impacted by the OS itself, for example, through the Server Optimization tab in a Windows system.

- SmithMicro's CheckIt utilities, inexpensive and effective, offer an easy way to quickly benchmark different processors as well as different servers. CheckIt is granular in its reporting, identifying metrics as diverse as system, video, network, and even disk performance.
- Various hardware vendors offer their own proprietary server load tools, usually available via download or on supporting software CDs. HP, for example, uses MeatGrinder and Thrasher for internal testing. And HP has made other tools available in the past, like Performance Stress Test Utility, which exercises the memory, disk, and network resources of your system. Another, System Stress Test, exercises memory, caching, and paging capabilities of Windows-based platforms.
- I recommend checking with each of your SAP Solution Stack partners to obtain testing tools specific to their platforms or technology products.
- Microsoft's WAST—Web Application Stress Tool—is ideal for baselining your future IIS-based servers, like SAP ITS WGATE server, for example.

Ziff-Davis Media provides a number of load testing tools, too, that they use to benchmark different systems. Both NetBench and WebBench are excellent products, relatively easy to set up and configure, and not exactly pricey. Download these load test utilities yourself from www.eTestingLabs.com.

Disk Subsystem and Database Testing

Quite a few tools are available that focus on proving the performance of disk subsystems. Each disk subsystem vendor, in fact, makes various tools available through its respective service and support organizations. Plus, other parties make tools available (that usually help promote their own goals), such as the following:

- Microsoft publishes a simple command-line utility called SQLIO that can be used to quickly provide MB/second and I/Os per second throughput data. This tool and sample scripts can be found on the Planning CD.
- Microsoft also makes its SQL Profiler testing tool available, used to record and play back SQL Server transactions.
- Intel used to publish Iometer controller and Dynamo workload generator utilities; today, these utilities are now in the open-source community. Refer to <http://sourceforge.net/projects/iometer/> for further information. Iometer is excellent for testing network throughput as well as disk subsystem and disk controller performance.

Why would Intel (at one time) and Microsoft go to the trouble of creating utilities that test disk subsystem performance? Simple—these tools help prove that their respective products perform well in the enterprise.

But other tools are useful from a pre-tuning perspective, too. One of the simplest is Microsoft's chkdsk utility, which can be executed from a command prompt for any locally attached drive letter. For years now, Microsoft has recommended that data and log drives for SAP database servers be configured for large block sizes. When SQL Server 2000 was introduced, Microsoft's ERP support team went so far as to say they recommended 64KB block sizes, in fact. It's simple to set this up when initially formatting a new data or log drive, using either the Windows Disk Administrator or the format command-line utility. But what becomes more difficult to check after the fact is whether a system has truly been formatted for 64KB. Getting back to chkdsk, I have found that running this utility is the simplest way to verify that a disk subsystem has indeed been set up correctly from an OS disk partition perspective. As illustrated in Figure 16.3, simply ensure that the number of bytes in each allocation unit equals 65,536 (which is 64KB), and that the file system is NTFS.

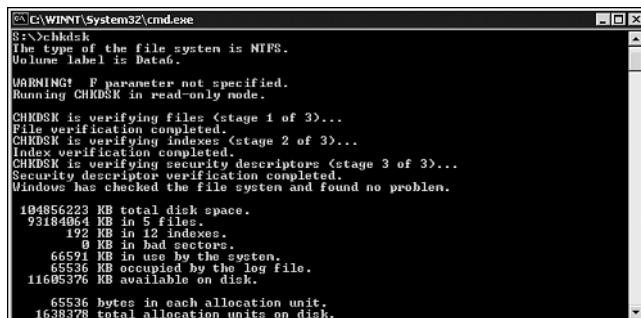


FIGURE 16.3 Microsoft's chkdsk utility is still quite valuable after all these years. Here, it makes it easy to verify that your data files are formatted for 64KB NTFS.

I recommend that you execute this utility without any switches, by the way, to ensure that you do not inadvertently write to the partition; without switches like /F, this utility is simply a display-only device in that it executes in read-only mode. Also, keep in mind that you must execute this command directly from each drive letter upon which a database data file or log resides. Thus, you need true access to these drives; a network share will not allow you to execute chkdsk.

Network Infrastructure Testing in the Real World

Testing the network infrastructure underpinning your mySAP solution can be quite important, especially if you are testing unproven technology or new user interfaces. I think back to the EnjoySAP initiative SAP AG embarked upon a few years ago, and

remember the “unknowns” everyone encountered that could only be solved through testing or lessons learned. To level-set those readers unfamiliar with this time, SAP was well known for its thin GUI since practically its inception; the SAPGUI was tremendously efficient in the days of R/3 3x through version 4.5x. One of the many goals of my second SAP load test ever was to prove just how efficient the GUI was, in fact. My client and I arranged for the installation of a T1 network connection between their client LAN and the SAP server LAN, and we ran a number of stress test runs over this 1.544mbps link.

The results were impressive. A total of 600 SAPGUI front-end clients utilized that link concurrently, running typical SAP R/3 transactions related to sales order entries, materials management, production planning, HR, and finance/controlling. And the link showed less than a 20% average utilization, which we figured later represented a range of 40 bytes and 1,500 bytes per transaction. These were real numbers, mind you, and proved not only the efficiency of the SAPGUI but also the scalability of network links typical at the time. If 600 users could run so well over a relatively slow T1 connection, think of how fast they could operate over a 10baseT or 100Mbit network—they would figuratively scream. And at the time, with the throughput benefits of Gigabit Ethernet around the corner, it seemed as though testing the network tying SAP and other systems and their front-end clients together would simply be a waste of time in the future.

But things changed, as they tend to do. Remarkably, it was an end user who responded to a question asked by Hasso Plattner, a co-CEO and co-chairman of SAP AG, that sparked one of the biggest changes to SAP up until that time. The question? “What do you think of the SAPGUI?” From what I understand, the end user was not aware of Hasso’s role at SAP AG, and quickly gave Hasso a rundown of the SAPGUI’s perceived shortcomings—it was a bit difficult to learn, unintuitive, and just plain ugly. And just like that, a new initiative was born.

From the onset, EnjoySAP was all about improving the end user’s experience, and by all accounts I believe that SAP AG was successful in most every aspect. The only downside was that the GUI gained a bit of weight, so to speak. Not that it couldn’t be trimmed down on the fly—the new EnjoySAP SAPGUI boasted controls from nearly the beginning that allowed its innovative features to be throttled back, thus returning it to fighting condition when it came to running over slower network links. But many of my colleagues and I learned of these changes the hard way—from our customers, through testing.

In response to this customer-led education, a few of us at the competency center got together with some networking gurus, and we set out to see exactly how well the EnjoySAP GUI would perform. We built out a test lab with 10, 100, and 1000mbit network infrastructure, brought in a pile of 32-bit and (at the time) newer 64-bit NICs, and went to work building and testing SAP systems. The results, I believe,

stunned all of us on two fronts. First, the SAPGUI testing showed that the public network traffic from a host of typical R/3 functional areas had grown to between 4,000 and 15,000 bytes (with everything enabled), a huge difference from the so-called classic GUI. Second, compared to 100mbit, the new Gigabit Ethernet gear was making little to no performance difference. In fact, in quite a few cases the NIC drivers were so poorly written that a server's CPU utilization actually *increased* a couple of percentage points, while the number of network packets and the average transaction response time remained nearly identical across the board. Even in the best of cases, we could not manage to get more than 15 percentage points of extra bandwidth out of these first Gigabit NICs and infrastructure. And these best-case examples were not even representative of typical SAP implementations.

Fortunately, things changed again and subsequent versions of SAP's user interface reclaimed much of what it had lost. The WebGUI interfaces were introduced in both HTML and Java formats, adding to the stable of SAP-supported user interfaces. Alongside this, Gigabit offerings matured into the products we know and use everywhere today. I hope I'm given an opportunity to do some more network testing for mySAP one day in the future. Until then, I feel pretty comfortable with things as they stand.

mySAP.com Component Layer Testing

When you think of optimizing specific SAP Solution Stack layers and components, you might not automatically think of the different SAP products themselves. But optimizing the mySAP components unique to your environment *before* you travel the road of SAP stress testing is huge! Consider the following areas:

- The sizing and utilization of your Program, Screen, and other buffers, various caches, Nametab buffers, SAP memory configuration/allocation, and so on dramatically impacts performance. Much of this can be reviewed by running transaction ST02.
- Verify that your “Top 50” online and batch transactions have been optimized in terms of ABAP and Java coding. You also want to pay particular attention to the Top 50 with regard to database request time, CPU time, dialog steps processed, and so on—this and much more is available through ST03.
- Review and validate that your Logon Load Balancing not only works, but that it effectively segregates different functional areas between different application servers. Don't forget to factor in the throughput differences inherent to different application server platforms; older processor technologies and older servers in general cannot host the same number of users as newer platforms, and more obvious differences also exist between completely different hardware vendors/platforms.

- Using SM50 or reviewing your SAP Profiles, analyze the distribution and number of the different work processes across each of your application servers and the Central Instance. Pay particular attention to dialog, update, and background work processes.

Much of this can be addressed through the judicious review and application of SAP Notes. I recommend that an SAP Basis person experienced with performance tuning and SAP CCMS spend a day or two with your systems, to both review and then optimize them. You might also consider bringing in a utility to do this for you. For example, Precise's Optistore for HP will improve your SAP solution's off-the-shelf performance, touting itself as a key way to your SLAs while reducing overall costs. More information and a number of free white papers can be obtained at:
<http://www.precisesoft.com/>.

Key SAP Stress-Testing Considerations

Before an SAP stress test can be conducted, much in the way of planning and preparation needs to take place. Development and use of an actual project plan is recommended to ensure that nothing falls through the cracks, as articulated next.

Creating an SAP Stress Test Project Plan

For starters, your stress test project plan will need to address the following:

- Pre-tuning your SAP Solution Stack.
- Selecting a stress-testing approach and tools, and determining the technical skillsets and other resources (and related time commitments) required to support the testing.
- Analyzing business processes to determine which mySAP components, functional areas, and so on will come into play.
- Time required to work with the business groups to understand how these business processes flow.
- When these business processes are understood, they need to be scripted.
- Developing a process for “warming up” the database, and then restoring this known state prior to each new test run. Regardless of whether you use a simple tape drive approach, or leverage the capabilities of your disk subsystem to “break off” a “snap” of your database, the process needs to be clear.
- Preparing for, executing, and monitoring each test run.
- Collecting source data, sifting through it, and performing data analysis.

Later, as you prepare for the stress test and begin to better understand the challenges associated with scripting complex business processes, you will probably make changes to the plan. It's not unusual, for example, to initially underestimate the time needed to find enough valid test data, build the test infrastructure, and even execute the test runs. These challenges and others are covered throughout this chapter.

Analyzing Online Users and Batch Processes

You must characterize your system in terms of the business processes that run as a part of the load you want to simulate in your stress test. This is a big part of identifying the mix of business processes, and usually boils down to characterizing online users and background or batch processes. Thus, if your goal is to simulate an average day on the system, your mix of online users and batch processes will probably be skewed toward the users. It will therefore be important to characterize which functional areas are represented, and how many users will typically be "active" within each functional area—I suggest leveraging your sizing data to nail this down.

However, if you want to execute a stress test that simulates your peak load on the system, you will probably need to run more background processes representing month-end reports or quarterly financial closings and so on. The nature of the work being performed by the online users will be different, too. For example, warehouse and inventory management users will be closing out inventory reports and displaying/updating material-related information. Manufacturing and supply chain users will be updating and running new demand planning and forecasting jobs. Business management teams will be seeking to better understand forecasting models, and may be updating budgets, employee records, and so on.

My point here is that the less you understand the nature of the business transactions inherent to a particular load, the less likely your stress test will prove useful to anyone. So take care to understand which mySAP components and SAP functional areas best represent the load of your system. And identify "backup" transactions and processes as well—these may become useful if your first choice in a particular functional area proves too difficult to script or too weak in terms of acquiring useful data necessary to drive stress testing.

It's All About the Data!

Though it's probably not obvious at first blush, access to the right data combinations is as important a consideration in selecting a business process for scripting as is anything else. That is, an easy script is essentially worthless if only a few combinations of data are available for use in a stress test. Why? Because after the data is used it is immediately cached at a number of different levels—by the hardware cache in your disk subsystem, by the database server's RDBMS cache, even by the application server's cache. Thus, the next time the same transaction is executed in your stress

test, this data will be immediately accessible and therefore will drive absolutely no work on your disk subsystem. If you recall from previous discussions, to truly drive your disk subsystem is a major goal of load testing—it will certainly be exercised in the real world! And because it also tends to represent the primary “bottleneck” for most enterprise applications, properly exercising your disk subsystem is paramount to a successful stress test.

When it comes to data, you can never really have too much. Valid data is crucial, of course, but quantity is equally important as just discussed. But exactly how much is enough? The right answer is simple—try to never use the same core input data twice during the execution of a test run, data like customer numbers, materials, invoices, and so on. In my experience, the following approach has proven successful:

1. I think about how long I will execute each test run. Thirty minutes is pretty typical, with another 15 needed for ramp-up. Thus, my target is to supply data for 45 minutes worth of processing.
2. I next analyze how long each script within each functional area takes to execute, including think time. If we take a relatively fast transaction like an FD32 credit check followed by a credit line increase into account, and then factor in a typical think time, it would be safe to say that the entire business process would consume one minute of wall clock time.
3. Next, I do some math. If my goal (as directed by the business) is to simulate the 20-member financial team’s peak load for this particular transaction, I would need to process 900 credit checks/updates throughout the stress test ($45 \times 1 \times 20$). In the best case, then, I want to have 900 different customer numbers on hand, so that I never pull up the same customer data twice.

This method can work for all transactions, obviously. But it gets a little cumbersome as the numbers get really large. For example, if my 20-member financial team became 200 people strong, I might be hard-pressed to dig up 9,000 customers that have even been set up in the system, much less able to be given credit increases. My suggestion is to work with the business, therefore, to determine what is real. Do your best to simulate the real world with the data available to you. And whenever possible, push to use business processes that enjoy a large array of data.

Updating Your Project Plan

With a basic stress test project plan beginning to take shape, and a host of underlying business processes and data better understood at this point, it’s time to take a closer look at budgets and goals, and to give some careful thought to which business processes must actually be scripted. And we must determine at what point an investment needs to be made in procuring a stress-testing utility that supports virtual users, based on which mySAP components will play a role in your stress test.

Ultimately, all of this will drive intelligent project plan updates as well as underscore the type and amount of scripting required to support the test. Specifically, you must

- Evaluate the need for real or virtual users
- Analyze which stress testing tool is most appropriate for your needs and budget
- Review and consider which business processes need to be scripted

Each of these three requirements is addressed in the following sections.

Real Versus Virtual Users

Some cases in your stress test plan will probably not be worth automating or scripting. Examples might include generating a bit of background noise through launching a couple of long-running reports, or even launching an MRP or other lengthy business process. In these instances, it could save time and expense to launch each of these processes through a dedicated SAPGUI session.

However, if you need to show the impact that more than a few users have on a system, it will be neither practical nor cost-effective to do so using “real” physical desktop or laptop clients, each running multiple SAPGUI sessions. Instead, investing in an SAP testing utility that supports *virtual users* is the ticket. Virtual users are exactly like “real” user sessions, with one core difference—hundreds of virtual sessions may execute on a single client driver, many more than the six concurrent sessions allowed by recent SAPGUI versions. The virtual SAPGUI sessions are executed behind the scenes via direct SAP Business API (BAPI) calls. In this way, the actual graphical user interface is not required to be displayed, as scripts written to support virtual users don’t fill in the contents of a SAPGUI screen by tabbing to it and entering data; they drive business processes through referencing German field names, the APIs for drop-down boxes, and so on.

Tools that support these powerful virtual capabilities must by their very nature support excellent monitoring and reporting capabilities. These features are by no means free, however. An exercise in price versus capabilities could be warranted in some cases, in fact. But in my experience, if you need to drive more than 50 or 60 SAPGUI sessions, a simple ROI study proves you need virtual testing capabilities. Because most stress tests involve hundreds or even thousands of users, your decision should be pretty clear. This is why virtual user sessions are the subject of most of the remainder of this chapter.

SAP-Aware Versus Freeware and Inexpensive Testing Tools

The argument for testing tools that boast an SAP BAPI-certified interface, and those that do not, comes down to price. Using SAP’s BAPI makes a testing tool “SAP-aware,” in that it can communicate with an SAP system through its standard API. In

this way, SAP-aware tools boast the following capabilities unavailable in other software:

- They can “screen-scrape”; thus, they pull information out of an SAP output screen like “transaction completed successfully” or “Order 0006011998 posted for customer 19940425” and use it to populate a variable.
- Similarly, they can read the contents of a field displayed by the SAPGUI even if the field is not actually “visible” on the screen.
- They can take this capability one step further, and execute transactions without even displaying the SAPGUI; the transaction actually runs, but does so “virtually” or behind the scenes without the need for a user interface. The folks at AutoTester call this VDT, or “virtual direct test” capability.
- Finally, SAP-aware tools can build on these capabilities and run many users—called virtual users—on the same physical desktops or laptops. By using a high-powered “client driver” like an 8-CPU server with 2GB of RAM, literally hundreds or even thousands of virtual users can be set in motion.

Tools like this are not cheap, however. The most popular SAP-aware test tool vendors charge between perhaps \$50K and \$150K for the privilege of being able to execute 500 virtual users on a single client driver. I’m most experienced with AutoTester’s products, including their excellent AutoController product for performing SAP load testing. The folks at AutoTester are some of the most flexible and responsive I’ve worked with in the IT industry, too, and their product line boasts being the first to come out with a BAPI-certified stress test product for SAP. That being said, I’ve also worked with Mercury’s WinRunner, QuickTest, and LoadRunner products, and also CompuWare’s TestPartner (which boasts the added distinction of being the first testing product to be certified for use with Web AS’s eCATT product).

Even factoring in the cost of licensing, the savings are still significant in the long run, however, in that 500 networked and managed desktops cost quite a bit more, the cost of paying real users to give up a Saturday or sacrifice normal business hours productivity costs more, and *not* stress testing costs even more. AutoTester allows its partners (like HP Consulting and Integration Services) to lease software licenses to their customers at a fixed price, making them even more attractive for a discrete stress test. In these cases, the customer keeps the scripts and data coming out of the testing, along with the knowledge that their system meets or exceeds expectations, but the software enabling the test is not theirs to keep. All in all, it’s an excellent way to go for the most price-conscious SAP stress-test projects.

But it is still possible to save some money if you’re willing to sacrifice the ability to easily test a lot of users running on a single server or two. Other low-end products can be used that can drive the SAPGUI simply because the SAPGUI is a Windows-based application. Similarly, tools can be used to drive the WebGUI, too. Segue’s

SilkPerformer, products hailing from Empirix, and even SPECweb utilities offer management and monitoring capabilities of a test run, at prices ranging from eight hundred to a few thousand dollars. For projects truly strapped for cash, Hiddensoft's AutoIT scripting utility can be a great way to go, too—not only is it free, but scripts are available over the Internet that have already been customized for various SAP transactions and general functions. So, if you have the client-driver hardware, the infrastructure, and the time but are lacking financial resources, this could be the answer for your particular stress test.

Developing Business Process Scripts

With all of the information gleaned thus far, it should be clear which business processes need to be run during the stress test to represent the typical load on the system. Interview the super users and functional specialists to clarify any questions you may have. Further, it should be clear as to what needs to be scripted to represent a month-end close or seasonal peak—whatever represents the system's busiest time.

In identifying these business processes, the specific mySAP.com components upon which these core business processes depend will be clearly identified. A mySAP financials stress test might exercise R/3 FI, BW, and SRM components, for example, whereas an SAP-enabled Executive Information System might require business processes touching SAP BW, SEM, and perhaps APO and R/3 Production Planning to be scripted. Use this information to begin planning for how and where you will ultimately develop and test scripts. Will you need a dedicated test environment or technical sandbox? Or will you simply (and more likely) need access to test or development clients within each mySAP system landscape? Further, consider how you will ensure that the data in these systems or clients remains static enough to support scripting—there is nothing worse than coding a set of business process scripts one week only to come back the next and find that your data is no longer valid, or the screens in your VA01 sales order transaction have changed!

Finally, not all business processes can be scripted by all tools. Certain mySAP components defy virtual scripting, some are simply not supported by particular test tools, and others present special challenges, all of which are discussed next.

How to Test mySAP Components

Until the last two years, when I thought of SAP stress testing I thought solely of R/3. With the advent of mySAP, however, all of that changed. No longer are interfaces to other systems handled through ALE (Application Link Enabling) or EDI (Electronic Data Interchange); instead, more and more systems are linked leveraging the Internet and protocols like HTTP and XML. And SAP's NetWeaver will push the use of Microsoft's .NET and Java connectivity/interoperability in general, further underscoring the need to stress test across the “extended enterprise.”

To execute a stress test, you fortunately do not need to be an expert in these protocols and initiatives, nor necessarily in the systems and products that these protocols bring together. Rather, if you “drive” the business process correctly, and the functional testers have done their work to ensure that the business process executes as expected, you can (theoretically) relax and focus on scripting business processes and then monitoring the stress test. This is great news to anyone faced with otherwise learning a slew of interfaces and products.

However, you’re not out of the woods yet. Scripting in different mySAP.com components can be demanding in that each component may need to be addressed differently. And every component includes a database server, which not only needs to be closely monitored but also needs to be warmed and restored prior to each test run. Thus, pure coordination activities rise to the surface as unavoidable and resource-consuming must-haves, lest the stress test results be compromised in terms of integrity and therefore value. This is why SAP’s standard benchmarking kits can prove valuable, as discussed in the next section.

SAP Standard Application Benchmarks

In some cases, it may make sense to execute an “SAP Standard Application Benchmark,” using benchmark kits published by SAP AG and available to SAP technology partners. This approach is valuable in that a particular solution stack can be measured or “benchmarked” against configurations that have been tested and published by SAP’s hardware technology partners. The downside is that such an approach does not leverage your own customer-specific data and business processes, though. I therefore find these types of benchmarks most useful when funded (or at least partially funded) by your hardware partner. For example, I have participated in customer-specific stress tests, where I used the custom SAP hardware sizing performed by the SAP Competency Center to guide me in actually building the architected solution. And then I tested this customer-specific solution for a potential customer, who had other hardware partners doing the same work in their respective hardware labs.

From the customer’s perspective, this works out well—they benefit from seeing a solution in action, and because the same benchmark kit is used by each hardware vendor, the results are very much apples-to-apples; resources and processes used in the execution and monitoring of an SAP benchmark are unswerving and consistent, reflecting dependable benchmark test results. Plus, most of the testing fees are often absorbed by the respective hardware sales organizations looking to close a big sale, making it an inexpensive insurance policy for the customer—they provide oversight to some degree, participate in Test Week, and walk away with great data and a sense of the real value proposition that each vendor brings to the table.

As illustrated in Figure 16.4, SAP AG publishes quite a few standard benchmark kits that can be used in customer-specific benchmarking exercises as I just described. A

kit exists for many of SAP's different mySAP components and technology solutions, such as

- mySAP Supply Chain Management, which covers SAP APO, the ATO functionality, and R/3 MM, PP, SD, and WM functional areas
- mySAP Business Intelligence, which focuses on SAP BW
- mySAP Product Lifecycle Management, specifically PLM
- E-Commerce solutions
- mySAP HR, specifically R/3 Cross-Application Time Sheets (CATS) and HR's payroll process
- mySAP Financials, focused on SAP R/3's FI functional area
- mySAP CRM, addressing Internet Sales and the CIC (Customer Interaction Center) functions
- Various industry solutions, like retail, banking utilities, and more

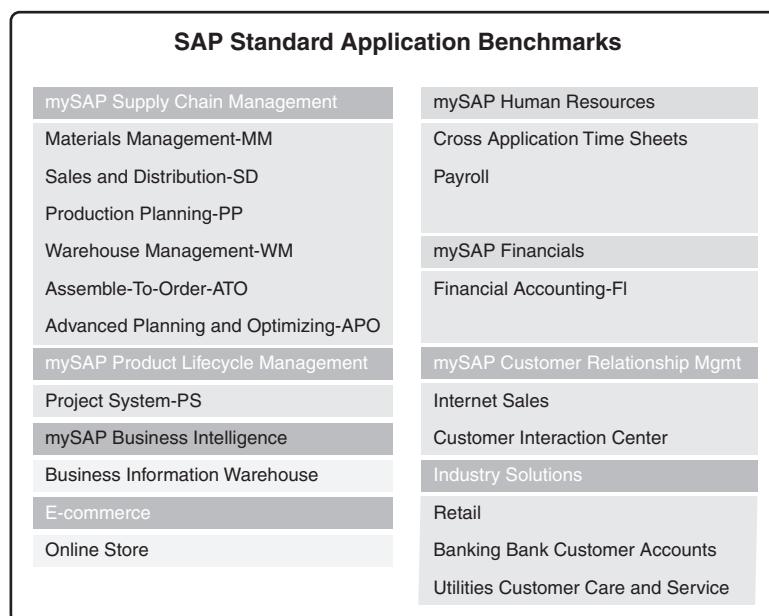


FIGURE 16.4 SAP's Standard Application Benchmarks cover the gamut across the core mySAP offerings.

Where eCATT Fits In

Another valuable tool provided by SAP is eCATT, or *extended Computer Aided Test Tool*, which was discussed in great detail throughout Chapter 15. eCATT is a perfect tool for testing multicomponent mySAP solutions, in that it supports the most common releases of mySAP products on the market or in data centers today, and can leverage its hooks into mySAP to drive complicated back-end processes. But eCATT does not natively support virtual users, relying instead on SAP's testing partners. So although eCATT is a powerful tool for proving functionality and performing regression-testing, it really takes a third-party SAP-aware tool like CompuWare's TestPartner (a Web AS-certified virtual product) or other tools to pull off a true stress test.

SE38—One Answer to Cross-Application Stress Testing

Because of the potential complexities associated with testing business processes that span your enterprise, I find it especially useful to leverage transaction SE38 when I can. SE38 can easily be scripted to run a variety of reports and other jobs, both online and in the background (running as batch processes). Selecting jobs that not only represent your typical workload but that also make calls to other mySAP components is one of the most straightforward methods of driving an enterprise stress test.

Problems exist with this “easy” approach, of course. First of all, I doubt that all of your key business processes can be neatly executed in this manner. Second, you need to consider how many transactions you actually have that can be started from SE38 that *touch multiple mySAP systems* (assuming your goal is a cross-application stress test). My guess is that there won’t be many. Combined, these two challenges tend to make SE38 only a (valuable!) piece of a larger puzzle, then. I like to augment a stress-test load with the heavy load that can come from executing simultaneous batch jobs executed in the foreground, for example. The results may guide you as to when and upon which servers you run your batch jobs, where you place your update work processes, and even help you characterize the degree of parallelism in your custom batch processes. The bulk of the puzzle is then completed by scripting business processes that originate in your other mySAP systems, some of which are discussed next.

Business Information Warehouse and SEM

Scripting BW queries has been a challenge since BW’s inception. Even though a number of stress testing utilities support communications through the SAPGUI or WebGUI (that is, via Internet Explorer), I have not found an easy way for a virtual user to launch a script started in one interface and then pass control of that script to another interface. Specifically, if you start a BW transaction or query through a *virtual SAPGUI session*, and a Web interface or MS Excel screen is subsequently invoked, you are hard pressed to actually “talk” to this new active window.

Conversely, if you script your business process via the Internet Explorer API, you can only “talk” to the initial screen—subsequent screens (used to enter variant data, or to actually view the results of a query) are not accessible via the virtual interface, either. In my experience, I have found five ways around these dilemmas, each of varying application or value:

- First, leverage SAP BW’s inherent capabilities to “trace” its own activity and then to replay it—refer to SAP’s “accelerator” white paper on how to accomplish this task.
- Use a third-party tool from companies like AutoTester or Mercury Interactive, but do NOT leverage virtual client capabilities. Rather, script everything for physical clients. This obviously requires access to the proper number of (adequately configured) clients. Hardware vendors like HP, IBM, and some others maintain a lot of equipment for this kind of use, but such services are almost always fee-based.
- Use a third-party tool’s virtual SAPGUI capabilities, but instead of true business processes, script only simpler “administrative” tasks, like those associated with managing SAP or the database (typical Basis and DBA scripted transactions). This may be all you need, if you achieve the goals set forth in your stress test project plan. For example, if you prove that your hardware platform or solution stack still performs well even after hitting a certain disk queue length, or you achieve metrics in terms of activity or load (like “x” number of logged-in/concurrent users or “x” number of concurrent processes), you don’t necessarily need to script anything else. These tests do nothing to prove that your system can handle the business processes expected to execute on your system, though, and are therefore usually of only limited value.
- Similar to the previous SAPGUI-based solution, script only WebGUI or Web-based activities; the same limitations as just described apply.
- Finally, use transaction RSRT in SAP BW to drive your queries. This transaction can execute any user-based query, but the difference from other queries is that the output—the query results—are kept “inside” the SAPGUI session rather than piped out to Excel or an Internet Explorer session. For stress testing and regression purposes, this is perfect, because you can script your complex BW queries in a virtual and repeatable manner without losing focus between the SAPGUI and the Internet Explorer or Excel interfaces.

Thus, by executing RSRT, you can perform all of your stress testing via the SAPGUI, and still retain the use of virtual client drivers—no need for hordes of physical desktops, or figuring out how to execute and (more to the point) control a combination of SAPGUI and WebGUI scripts at the same time!

Enterprise Portal and SRM/Enterprise Buyer Pro

When it comes to stress testing HTML-based and Java-based mySAP components like Enterprise Portal and Strategic Relationship Management, these products must naturally be driven by Web-based tools. Fortunately, all of the big players in the SAP stress-testing market today have this capability, both in physical and virtual mode. And just as importantly, a host of other testing utilities are available to drive these GUI platforms. The key to making your life simpler is to obtain a product that supports the various controls and other constructs that define a Web browser. Another key is the maker and version of the Web browser upon which you have standardized—not all testing tools support all browsers, for example.

Other mySAP Components and Considerations

Generally, if an SAP component can be accessed via the SAPGUI or WebGUI, you can script a business process using a tool like AutoTester or WinRunner/QuickTest. Of course, exceptions exist, as we just discovered with regard to SAP BW. By and large, though, if the applications supports a standard SAP interface, you're well positioned to leverage it.

If you are unsure of exactly *what* to script, and have no access to super users or functional specialists, I recommend obtaining the mySAP component's completed sizing questionnaire that was shared with each potential hardware vendor. Here, the number of users and transactions, and more importantly, the functional areas and even the types of transactions to be eventually executed will be identified. All of this information can then become a foundation for developing a list of potential stress test transactions and general business processes to execute. For example, if a completed SAP sizing questionnaire tells me that 400 users will execute MM (materials management) functions, and another 50 users will execute in a few other functional areas, it would be safe to assume that MM01, MM02, MM03, MMBE, MD04, ME57, MB1B, and MB1C should be considered first for scripting—these represent typical MM business transactions, and reflect the bulk of activity conducted by many of the future system's online users.

Another thing that has helped me script business processes touching applications with which I am not deeply experienced involves the SAP benchmarking kits. That is, by their very nature, each benchmark kit provides a sequence of transactions that can be executed in support of the benchmarked business processes. If you take a look at what constitutes the CRM benchmark, for example, you will quickly determine *how* to log in to a CRM system, which core transactions to run (like CICO), and other details relevant to typical CRM business processes. This impromptu education is invaluable if you are asked to stress test a system *before* the SAP TSO fully understands all of the business activities that might surround such an effort.

Script Development and Preparation in the Real World

The time and tasks related to developing the business process scripts to be executed during test week are discussed in this section. Before we begin, understand that the term *test week* is used pretty loosely—it's not unusual for this “week” to actually run two calendar weeks. Most often, though, I have found that a calendar week—about seven days—tends to represent the average amount of time a company actually stress tests their production environment. This clearly does not include time spent in planning and preparation.

Unfortunately, we're not talking about eight-hour days and lengthy lunches here, either. For the folks engaged in supporting test week, it seems as if no matter what level of preparation is taken up front, test week consists of something close to 12-hour days. And the days leading up to test week are usually not much better.

Consider the following typical work loads I have scripted:

- In one of the first medium-sized R/3 systems I ever tested, preparation for a single test week consumed eight previous calendar weeks. During this time, I created over 50 online user transaction scripts across six different R/3 functional areas.
- Another stress test engagement required preparation to the tune of four weeks for R/3, and one additional week for BW. I coded 20 discrete online R/3 transactions, three batch processes with over a hundred variants, and five BW queries and a mix of other BW transactions in support of this test.
- Another R/3 system stress test required four weeks to script six end-to-end business processes (approximately 30 online transactions representing processes like order-to-cash and so on), and another week and a half to execute and analyze the test runs.
- Another really simple stress test consisting of only five core business transactions consumed two weeks to script and test, and another three days to execute (a short “test week”). Comparing this particular test to the previous tests illustrates the work involved up front; additional scripts become easier to develop after the core work is completed.

Why such a long time to prepare? Because there's much to do. In terms of the big picture, it's really not as big an investment as it might seem—a single stress test normally supports a production environment that will afterwards stay essentially the same for the next two to three years. So an investment of 2–8 weeks seems pretty reasonable in this light.

I spend much time with both the business and technical teams identifying functional process flows and work flows, and then gathering enough valid data to support a test. After installing the scripting tool on my laptop or customer-provided

equipment, even more time is spent creating scripts that functionally work; like programming, script development is subject to development iterations, including fixing coding bugs, introducing standard subroutines for error handling and reporting, and so on. When I understand the business task to be scripted (such as the steps involved with creating a sales order), I must *record* the business process using the scripting tool. Recording involves tracking each transaction, keyboard entry, mouse-click, and so on that I execute in the SAPGUI (or WebGUI) to a text-based file called a script. This script is then edited to reflect variable input, to run in virtual user mode, and so on. Don't worry too much about this process right now, though. I discuss the details of scripting later in this section.

During the script development process, I like to hold at least twice-a-week status reviews—these therefore consume project time as well. During each one- to two-hour meeting, I ensure that all testing assumptions still hold true, share status updates related to scripting in the various functional areas, identify issues or problem areas, and share successes. If something is changing (for example, I'm losing my test client because it's getting refreshed, or the entire test environment is being refreshed again), we discuss what this means to the test project plan.

Fortunately, some of the tasks associated with a stress test can be performed concurrently, like working through the final test goals and success criteria, determining ways to validate success criteria, making revisions to the test plan and refining a testing methodology, working with the business units or functional teams to understand business processes or verify transaction flows, actually test-executing each transaction, and so on. This gives you an opportunity to *crash* the project schedule, a project management term that implies throwing more bodies at a project to achieve milestones faster; in some cases, one or two of my colleagues have assisted me with data collection and script development, allowing us to complete these tasks faster than I ever could alone. On the other hand, tasks that cannot be crashed imply things that must occur sequentially, one task after the other. A good example of this is testing a script for bugs—each script must be recorded and then set up to run in virtual mode leveraging variable input before any real bug testing can take place.

Core Script Development

In the weeks prior to test week, the script development process commences and matures or evolves as each mini-milestone is achieved. The following list assumes you are scripting in AutoTester ONE, version 2.x (other scripting languages that support virtual direct testing are similar in form and function). All of the following scripts can be found on the Planning CD:

- The correct SAP testing environment must be decided in advance, and then a SAPGUI connection needs to be made to it. To record a script using AutoTester ONE that will later be run in virtual mode (by using AutoController), for example, you need to establish a session with a particular SAP server and

instance's System ID (SID), and perhaps a logon group. I usually set up a standard SAPLOGIN.aof script for this purpose, which is then called by each functional script for the sole purpose of logging in to the system to be tested. The core functionality of this script amounts to executing a command like

```
SapEstablishSession "server_name" "00" front
```

where *server_name* is the name of the application server or central instance, *00* is the instance ID, and *front* tells the script to display the SAPGUI (useful for testing). For a virtual connection to an SAP logon group named FINANCE in the STG system, the SAPLOGIN.aof script might instead execute

```
SapLoadEstablishSession "STG" "message_server_name" "/H/" "FINANCE" front  
groupconnect
```

Of course, you need to ensure that the name of the logon group is correct; the SAP Basis team configures these via transaction SMLG.

- After a session is established with the SAP system, you can then log in with a scripted command like

```
SapLogin "010" "gwanders" "password" "E"
```

where *010* is the client, *gwanders* is your user ID in this particular client, and *E* designates English. Your password can be saved in encrypted form or as shown here, in clear-text (the password is simply “password” in this case). The great thing about establishing a *session* (rather than scripting a process that logs in to the SAPGUI by clicking a SAPGUI icon or in a similar fashion) and then logging in with the SapLogin command, is that many recorded commands are automatically recorded in a manner that allows for later running the script virtually. It is precisely this virtual capability that we want to leverage. Also, establishing a session has the added benefit that the SAPGUI *information messages* and screen names are captured by the script. These can later be used programmatically to drive if-then logic. For example, if screen “Display Material*” does not appear after 60 seconds, then execute ERROR_ROUTINE.aof, a scripted standard error routine that might write an error message to a log and then break the virtual connection to your SAP instance. Note the flexibility of some of these commands, too—an asterisk can be used to denote wildcards. In this case, any screen that is displayed within 60 seconds and starts with the title “Display Material” would *not* execute the error routine.

- Now we are at a point where business processes can be scripted. I assume you have already worked with your super users and functional experts and therefore understand the exact workflow that you need to duplicate via a script. It's also helpful to understand how different business processes work, but it's not

required. Interestingly, as a Basis or technical resource, one of the best ways to gain an “Intro to SD” or “Intro to MM” education is to script in those business areas! In fact, through stress-test scripting I learned how a number of different business areas worked, enabling me to make future stress tests even more pertinent or valuable to my own clients. For example, by understanding that transaction VA01 in R/3 begins a whole slew of business processes, I was able to identify additional end-to-end work flows that needed to be scripted to represent one of my customer’s core business areas—they initially only wanted to create orders, but hadn’t thought about different sales areas, or performing repairs and credit returns, and so on.

- When a basic script has been created, it needs to be tested to ensure that it actually runs with the initial valid data provided by the functional experts tasked with assisting you. Just getting the basic functionality to “work” can be a challenge; more than once, I’ve been provided data that did not actually work in the particular system or client in which I was asked to perform the stress-testing script development.
- With the basic script validated and operational, I either collect more data myself or request more valid data from the experts. Finding this new input data was very tedious for me in the past, before I fully understood that not all data works in all business processes. For example, only specific customers may order specific items defined in a sales group, and only specific materials are actually associated with certain factories, plants, warehouses, and so on. To complicate things further, only specific storage locations within certain plants may house certain materials! It’s the *rules* of various business processes that make stress testing (and functional testing, for that matter) challenging. Once I began to understand that, I was able to intelligently search for valid data combinations on my own, thus freeing up my customer’s people to focus on their own tasks at hand.
- Testing my newfound data is next required. I perform my own “testing” of the validity of this new data by converting hard-coded input data (like the material number in an MM03 script that displays materials) into a variable. Thus, in an MM03 script, material 19690823 becomes variable `MM03.INPUT`, or something similar. I can define `MM03.INPUT` as a text or numeric variable, and point to various sources like an Excel spreadsheet or text file (samples of which are included on the Planning CD). I prefer Excel in that it’s easier to manipulate than a text file, but either works equally well. To test the data, I create a simple loop and inside this also include a “log” command, used to write the pass/fail nature of the data. I can then start the script and walk away; success or failure of the particular data is recorded and noted in the output log file, helping me to simply find good data combinations. More of this type of activity is discussed later in this list and in this chapter. Suffice it to say here, though,

that testing data can be time-consuming in and of itself, and therefore requires as much of an automated approach as possible.

- Next, each script is adjusted or fine-tuned to better support virtual capabilities or `what-if` conditions. Thus, script syntax and commands are changed, necessitating more single-unit testing after these changes are made. For example, the names of SAP screens using AutoTester's "SAPLookWindows" command are shortened and appended with an asterisk. This allows for variations in the screen name—some screens append a customer name or number to it, or a material number or some other unique value is added to it somewhere, otherwise making these not easily scripted. Further, I think about what might happen if a particular screen is not displayed by the SAPGUI when it is "supposed" to be displayed, and how to get around this—maybe a business rule only applies to certain customers, for example. So I add `if-then` logic to the script to address this.
- In the case of errors resulting from business rules, I also write code that directs the script to fail its current loop. In doing so, I force it to write an update to my output file for that particular transaction or business area indicating that the combination of data failed, and then I force the script to pull in a fresh combination of input data and start over. By programmatically addressing these "errors," I avoid running my error subroutine, which ends the SAP virtual session and therefore causes me to lose one of my test users. I also avoid errors that otherwise cause the script to abort, which in turn tends to hold onto client driver resources (eventually requiring a reboot to remedy).
- Next, subroutines are added to each business transaction that cover things like explicit error conditions or failures. These kinds of conditions are displayed by the SAPGUI at the bottom of the screen, the values of which can be easily "scraped" and used in `if-then` statements. I pipe this data to the output file designated for the transaction or business area as well; it's valuable to understand how many, and for what reason, business transactions fail in a particular test run, after all.
- I also build in routines for providing random, unique, or other input data not already addressed. For one of my customers, for example, I needed to provide a unique value for a configure-to-order (CTO) product. Their business rules required a unique serial number (which makes sense!), so I created a variable that concatenated the year, month, day, hour, minute, second, and the virtual client's unique number. Just like that, I had a serial number that would never be duplicated, ever, no matter how many virtual clients ran the CTO process, or when. This and other script subroutine examples are discussed in more detail later in this section, and can be found on the Planning CD.

- I build any additional and necessary loops or do-while constructs. Sometimes, my customers request that a particular set of input files is used to feed their scripts—a loop and if-then condition must therefore be set up to determine when the end of a file has been reached. Similarly, when a virtual user establishes a session with SAP, it makes sense to execute a business process over and over again instead of logging in, executing the functional piece of the script, and then logging out. So a loop and if-then logic is built around the entire business process in this case as well.
- Now, around each core screen in a transaction or function, I build a process that collects response time statistics. This involves capturing data for whatever occurs prior to moving to the next screen in a business process. For example, if a test user presses Enter to advance from an initial VL02 “Change Outbound Delivery” screen to a “Delivery Change Overview” screen, I capture at minimum the associated SAP application server response time. Sometimes, I might also capture the “wall clock time,” or the amount of real time that has elapsed in a particular business process.
- Finally, I capture all of this response time data, error messages generated previously, and any other information captured through variables, and pipe all of this out to an output file designated for the transaction or functional area. For an example of what this entails, see the OUTPUT.aof file on the Planning CD.

When both scripting and associated single-unit testing is completed (including finding more last-minute data, which is not as unlikely as it sounds), the script is ready to be tested by multiple virtual users running concurrently. Exactly how many virtual users is up to the functional team and SAP project manager. Part of the work of stress testing involves understanding how many virtual clients will be needed to emulate the future production environment, including which functional areas and so on. When you have this data in hand, you are ready to progress to the next phase of script development—defining, installing, and configuring your load testing infrastructure.

Stress Testing Client Infrastructure

In stress testing, your goal is to run your tests in an environment that looks like production. In the best of worlds, you run your stress test on your soon-to-be actual production system. This is *always* preferred to any other alternatives, but is sometimes not possible. In that case, an identical staging or disaster recovery system, or something cobbled together that truly represents production, must suffice.

Just as importantly, you need to assemble the “client side” of your production system. It is this client infrastructure that I refer to when I talk about “stress test client infrastructure.” As I mentioned previously, it is rare for a customer to have enough desktop/laptop clients, network infrastructure, and time to actually assemble

a mock client environment—this is precisely why virtual direct testing software products were built in the first place. But you still need some amount of in-place hardware to serve as client drivers. In other words, you will need to install one to perhaps four or five servers to provide the client load on the production system. Further, you'll need one or more computers to control these client drivers. In the world of AutoTester products, both of these functions are performed by AutoController—an AutoController "Virtual Client" product handles the former task, while the "AutoController Console" addresses the latter. For specific and concise AutoController installation instructions, please refer to AutoTester-provided documentation and their Web site: <http://www.autotester.com>. And refer to the following for a high-level task list.

- Size the client infrastructure. For client drivers, I recommend running no more than 100 virtual users per processor (1.4GHz or faster). Extrapolate as necessary, based on the type and number of servers you have available. For the console, something in the neighborhood of 300–400 virtual users is supported per processor. And don't forget to size for RAM. I suggest 512MB for 400 virtual users, which includes 128MB for your operating system and the AutoController console product itself. Much less RAM is actually required, but this gives you space if you need it, thereby ensuring minimum paging to your pagefile.
- Install the client infrastructure hardware and operating system. For an 800-user virtual test, it's common to have two client driver servers and a single console. Often, I'll load the console directly onto the same server running the client driver if I have the requisite CPU horsepower. Regardless, you'll want to load Windows 2000 Advanced Server (to take advantage of all of your processing power, up to eight CPUs), and load Service Pack 2 at minimum. And stay away from /3GB and /PAE memory management, as these options are not supported by AutoController (which is actually fine, as memory beyond 2GB is not needed anyway).
- Install the SAPGUI, being careful in the installation process to include the option to also install SAP development tools, and then copy both the services and SAPMSG.ini file from one of your SAP applications servers to your client drivers and console. I've included samples of both these files, so that you can clearly see the kind of data that must be present.
- Install the AutoController client and console software (in any order); refer to AutoTester's documentation for details.
- Configure each AutoController client, as documented in the "AC Virtual Client Software Notes" document found on the Planning CD. Note that it will make your life easier (and the scripts I provided on the Planning CD more useful) if you limit the number of characters in the "Remote Virtual Client Name" field to three. This is because the virtual name associated with each virtual test user

generated by this particular client driver uses this as part of its name. Longer or shorter names therefore change the length of the name of each virtual test user.

At this point, you can begin building the AutoController Console packages that will execute the individual (or set of) scripts previously developed. There are a number of ways to group scripts or sets of scripts. Initially, you will want to test each individual script in a single package, however, to ensure that a single virtual user can run successfully on this new client-driver platform. This theoretically represents the final iteration of “single unit testing,” though in reality there will still be a need for it every time a change is introduced in a script.

When a single user can run in virtual mode, I like to increment the packages to run 10 virtual users; executing one package therefore starts 10 identical scripts. I verify that the input data used is indeed random, ordered, or as specified by the customer. And then I resolve any issues related to locking/contention (usually by revising a script to check for new “warning” or “error” messages generated by the virtual SAPGUI session and not seen before). If changes have been made to the scripts, I of course fall back to single-unit testing and ramp up to 10 users again, as just discussed.

After 10 virtual users run their scripts to completion successfully, I jump to the maximum required per the test plan, or to 100 virtual users, whichever is less. In an AutoController package responsible for starting many virtual users, it’s very important to ensure that all sessions are *not* started at the same time, but rather are staggered. In other words, ramp-up time must be taken into account—a well-thought-out stress test run should not actually commence until all virtual users have logged in to the system and are ready to “work.” Further, things like built-in delay times, waits and think times need to be carefully reviewed in lieu of the increased load 100 users place on the system, to ensure that any built-in delays still reflect the targeted think times even under load, and that everything still supports the stress test’s success criteria and goals.

Normally, 100 users is where I end my concurrent script testing—if 100 users can run the same transaction concurrently, with different data, and run to completion, in my experience quite a few more users can run concurrently, too. Plus, the number 100 is a very manageable number, as it’s large enough to speed through ramp-up, but small enough to allow me to track the status of individual virtual users as warranted. It’s therefore about the largest number of virtual users I assign to a package that I actually use for stress testing.

I perform 100-concurrent-user script-testing for each functional area, business process, or transaction that will become part of the stress test, working to ensure that the *mix* of online and batch transactions represents what my customer expects to see

in production, therefore reflecting their stress-test goals. For example, I'll build packages in AutoController that consist of a mix of BW, APO, and R/3 transactions, the latter of which may consist perhaps of 40 SD users, 120 financial users, 200 materials management users, and 40 production planning users if the R/3 online-user count has been identified as 400 and the functional mix has been earmarked at 10%/30%/50%/10% respectively.

Before we go on, I need to mention one final testing observation—it is very common for stress-testing client-driver utilities to appear single-threaded; indeed, some are and can therefore execute on only a single CPU. In the case of AutoController, this might appear to be the case, too. However, it's not—I've run this product on 8-CPU machines and have witnessed first-hand a nice distribution of work across all of these processors. The key in this case is to open the Windows Task Manager, navigate to the Processes tab, and find the `vatrun` process. Right-click this process and select the option to change processor affinity. Only the first processor is selected by default. Change this so that the `vatrun` process executes on all processors (you can do this "on the fly," in fact), and enjoy the results.

In the next few sections, I detail some of the specific scripting challenges touched upon thus far, and indicate what these scripts might look like in the real world of AutoTester and AutoController scripting.

Creating Administrative and Other Utility Scripts

Good programming or scripting often focuses on creating reusable code, like subroutines that can be accessed when needed. In the course of stress testing, I have developed a cadre of subroutines that seem to prove useful over and over again, like the following (which can be found on the Planning CD):

- Log in to the SAPGUI and establish a virtual session (`SAPLOGIN.aof`)
- Log out of the SAPGUI (`SAPLOGOFF.aof`)
- Random number generator (`RANDOM_NUMBER_GENERATOR.aof`)
- Unique number generator (`UNIQUE_NUMBER_GENERATOR.aof`)
- SAPGUI screen-scraper, useful in capturing error and warning messages or basic informational messages like "Credit limit for customer 320030613 changed" (`SCREEN_SCRAPER.aof`)
- A compare routine, useful for incorporating into `if-then` conditional statements (`COMPARE_ROUTINE.aof`)
- Read from an input text file (`READ_FILE.aof`)
- A reporting/statistics-capturing subroutine, including writing this information out to a text file (`OUTPUT.aof`)

Many times I actually build these subroutines directly into my script, rather than calling them—it just depends on the likelihood of having to change something, or the need for a specific subroutine, versus my willingness to manage separate subroutines. Regardless, I’m sure you will agree that these subroutines save you coding time.

Login and Establishing a Virtual User Session

There is more to running a good stress test than simply logging in a few hundred virtual users and letting them run loose. To control the ramp-up of a stress test such that all users actually have the opportunity to log in successfully, the users need to be logged in over a period of time. Why? Because logging in 500 users at once results in many of those login scripts aborting for timeout reasons.

At one time, I used to simply create small AutoController packages, and then manually release these in a controlled fashion, thereby executing the AutoTester scripts within each package in a controlled fashion as well. But I always admired the way that the SAP Benchmark Kits could log in a new benchmark user every second, and in doing so thus automatically control login “behind the scenes.” With the help of one of my very best friends and colleagues, we developed a similar programmatic approach for AutoTester ONE that I call *Staggered SAPGUI Login*. The core logic behind this approach is to manipulate the unique number assigned to each virtual user. Because this is held in a system variable, it is available for use even before a script is initialized or a variable file is identified. The scripted subroutine looks something like this:

- First, it’s good practice for the SAP virtual test user to check whether it is already logged in to a mySAP instance—if so, no other work is required, and the real purpose behind the script can commence (like displaying a purchase requisition, creating a sales order, and so on). Otherwise, the staggered login portion of the script must be executed. This is accomplished by the following code:

```
Compare $SAP_SESSION "0"
```

If this system variable is “0”, you are not logged in. At this point the rest of the subroutine in the script takes over (I use an if-then argument to control this), and therefore the lines discussed in the next few bullet points are executed. Note that if you *were* already logged in, execution of the script would drop out of the if-then clause and continue with the meat of the script.

- Next, I display and compare the value of the system variable \$MACHINE (explained in more detail shortly) to the word “Machine.” This is because the machine variable is only set to something other than “Machine” when running a script virtually through AutoController. So if the variable is simply

“Machine”, I am obviously not running a stress test—I’m probably busy coding and fixing bugs—so I hard-code my wait time to log in to the SAPGUI to a 1, for one second. The code looks like this:

```
Compare $MACHINE "Machine"
If Equal
>Assign LOGIN.COUNTER = "1"
```

- The key to the whole Staggered SAPGUI Login subroutine lies in the next two lines of code.

```
Otherwise
>SubString $MACHINE 17 2 LOGIN.COUNTERTXT
```

In this case, if the client is indeed a virtual client, it will have a unique number assigned to it (rather than simply the label “Machine”). This number can then be extracted from the complete machine name of the virtual user. For example, if you installed the AutoController client driver software with the remote virtual client name of “GLO,” virtual client 197 would have a full name of “GLO.VIRT.USER.0197.” Performing the SubString function “17 2” would extract the last two characters “97”. Similarly, “16 3” would extract the last three “197”—the SubString function works from left to right, counting to the character position indicated by the first number in the command, and extracting the equivalent number of characters specified by the second number in the command. Because these are alphanumeric characters, I assign the value to a text variable (`LOGIN.COUNTERTXT`).

- After the text variable `LOGIN.COUNTERTXT` is assigned, I can then manipulate the data in any way I choose, including converting it to a numeric value as shown here:

```
>Assign LOGIN.COUNTER = LOGIN.COUNTERTXT
>Message LOGIN.DELAY wait 2
>Delay LOGIN.COUNTER
```

The `LOGIN.COUNTER` variable is a numeric variable that can subsequently be used in comparisons, or as a starting point in a countdown, or used in a simple “delay” command as depicted in the preceding example. Note that the “message” command is optional—I like to see for myself that the subroutine actually worked while I perform my script testing. But this is commented out in the actual code finally used for scripting because it’s not necessary and in fact detrimental as it consumes video resources on the client driver machine. Think about the screen refreshes necessary for 400 virtual users running on a single client driver simultaneously, and you’ll begin to understand that any “extra” message boxes need to be avoided.

It's important that this scripted subroutine be executed *before* you log in to SAP (that is, before you execute `SAPLOGIN.aof`). Why? Because you are executing this subroutine to control when the SAP login process should commence. So in keeping with good coding standards, it cannot exist as part of your SAP login script. Instead, it needs to be executed by the actual functional script, or an “umbrella” script used to execute multiple discrete transactions.

You might wonder why I used a system variable (`$MACHINE`) and not a user-defined variable. This is because most of my scripts tend to leverage different variable files, each with perhaps different variables defined. System variables, on the other hand, are consistently available even when a variable file has not been defined. And the `$MACHINE` variable in particular will always be unique to a particular virtual client, making it an indispensable part of the Staggered SAPGUI Login approach.

Ramping Up Users and Processes

With specific AutoController packages created for each functional area or business process to be tested, ramping up virtual users (and therefore active work processes) is pretty easy. As I said before, I define a set number of users to a package. If I need to execute 300 Sales and Distribution users on an R/3 system, I create three packages, each defined to run 100 virtual users. To control the ramp-up, at the prescribed time I simply release the first package, and monitor the SAP logins via SAPGUI transaction AL08 (as shown in Figure 16.5, which displays “Current Active Users”) or ST07 (the “Application Monitor: User Distribution”) transaction. It's also possible through the AutoController console to monitor the activities of the individual virtual users themselves, but I tend to shy away from this functionality (I disable this feature) because it consumes a great amount of additional processing power on the AC console.

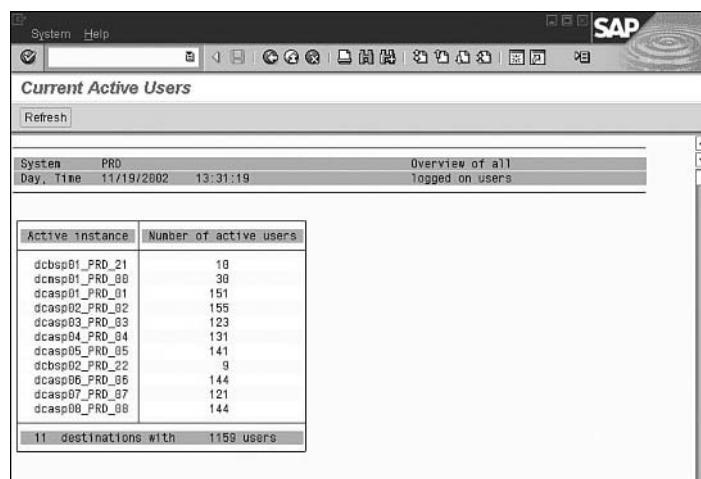


FIGURE 16.5 CCMS Transaction AL08 provides information as to the number of active users running on a system, sorted by SAP application server.

After the first 100 users are logged in, I then release the second package and follow the same monitoring process just discussed. At 200 users, I release the third and final package. When all 300 users are logged in and executing transactions, I note the specific “stop time” for ramp-up, which doubles as the “start time” for the actual stress test.

Later, when analyzing the data associated with the test run, I toss out all transactions and data associated with ramp-up. Doing so allows me to focus only on the statistics associated with the actual run. This underscores why it is so critical to track the start and stop times for each individual transaction that is executed—recording start and stop times (also called “wall clock” time) is performed programmatically in the body of the script’s loop, and written to the output file by virtue of the output subroutine called by the script.

Collecting Statistics

Statistics are critical to the stress test; failing to capture the proper source data is therefore akin to throwing away most of the budget money earmarked for stress testing. I normally collect the following data and statistics:

- In regard to housekeeping, I capture the virtual client’s machine name (\$MACHINE), the test case ID associated with the script (a high-level ID, like “SAP SD” or “BW Queries”), a “Run Name” associated with the particular script (useful for versioning), and the host name of the SAP server in which the test user logged in.
- General statistics, such as the current date, the time that the script began, the time that the functional loop began executing, when the functional loop completed (which usually coincides with when the script wrote its reporting/statistics data to an output file), and when the entire script ended.
- Specific statistical data, including the name of the SAPGUI screen and how long (response time) it took to display the screen, the screen’s input data (like customer number, material number, PO number, and so on), and the screen’s output data (like the order number created, or an information message noting the fact that a record was updated, or an update failed, and so on).

Other statistical data proves useful, too, though this data may not necessarily be collected via the scripts being executed but instead through other means. For example, I like to collect test-run data that represents a snapshot in time (like every two or three minutes) over the course of a test run. The snapshot data might include the number of concurrent processes executing, the number of users actually logged in and actively using a work process, the disk queue length associated with each of the database partitions, average CPU utilization of each application, database and

central instance server, and so on. This information is captured in a variety of ways—through another SAPGUI session used for monitoring the stress test run, through Performance Monitor or other OS-based performance utilities, and even from the output generated through the testing tool itself (for example, AutoController can generate a comprehensive “results” file for your stress test run, if configured and enabled to do so).

I've also created specific AutoTester scripts designed to capture some of the source data just discussed. These are intended to run concurrently with my stress test scripts (another example of scriptable and repeatable background noise), and include powerful CCMS transactions like ST07, which is also shown in Figure 16.6. I am fond of ST07 because it gives me a snapshot of not only how many users are logged in, but how many are actually active simultaneously, how many are actively using a work process at that point in time, how many average user sessions are logged in, and even how many application servers are servicing various functional areas (which verifies that any logon load balancing is working as expected). Automating this transaction gives me one less thing to worry about when executing the actual stress test runs. A sample ST07 data collection script can be found on the Planning CD.

Application Monitor: User Distribution						
		Choose		Sort		SAP buffer
				DB accesses		DB memory
				Response time		Quantity structure
				History		
Database	Name	PRO		Time	13:38:13	
Server	DCDBP20			Date	12/19/2002	
System	NSQL					
User	12744			all clients		
Number of servers	11			Work processes		313
Application	Number of users			sess. per		
logged on	active	In VP		User		App1. Server
Basis Components	104	12	4	2.32		10
Controlling	5	2	0	1.60		5
Cross-Application Components	28	6	0	1.80		6
Environment, Health & Safety	6	1	0	1.50		4
Financial Accounting	59	12	1	2.83		8
Logistics - General	56	8	0	2.75		8
Logistics Execution	78	8	1	2.09		6
Materials Management	168	23	1	2.74		8
Personnel Management	19	1	0	1.26		2
Personnel Time Management	7	0	0	1.00		3
Plant Maintenance	221	34	2	2.36		8
Production Planning and Control	3	2	0	3.00		3
Project System	5	2	0	1.00		3
Quality Management	9	0	1	1.00		6
Sales and Distribution	40	5	0	3.27		5
Treasury	1	0	0	3.00		1
Other	339	14	27	1.50		11
total	1,124	130	37	2.15		11

FIGURE 16.6 The CCMS Application Monitor invoked by ST07 provides active-user data, sorted by functional area.

Regardless of how you collect your source data, it's critical to understand all of the statistics you want to analyze later, as you must actively capture this data during the run. Even the simple task of starting a PerfMon log must be actively performed; I suggest creating a simple checklist of things you want to capture, so that you

remember to do so during the actual test runs—without a checklist, I know personally that I would have spent more time *rerunning* test-runs to collect forgotten data, instead of getting to the business of analyzing that data.

Logging Out—Gracefully Ending Your Test Session

When you have completed your test run, or have decided to end or otherwise abort the run, you need to gracefully stop the test. The key here is “gracefully”! It’s pretty simple (though time-consuming) to kill all of the SAP sessions via transaction SM04. Faster still is to abort the actively running AutoController packages. But from a data-collection perspective, this is not the best answer. Why? Because correctly logging out of the virtual session is critical; an unsuccessful logout can get in the way of collecting statistics associated with the run, and skew the test results, too.

In the best of cases, simply aborting a script only causes you to lose the statistics associated with the last loop in the test run. It doesn’t sound like much, but for a test of 1,000 users, losing the results for something between 1 and 1,000 transactions will skew your results and, depending on the number of loops, actually represent a wide margin of error. Some of the 1,000 transactions may just have started, some might be halfway through a complete business process (which then might invalidate the entire business process for the test, given that it never completes successfully), and other transactions may be in the middle of writing output statistics (some of which may be actually written, while others are lost). And worst case, a whole slew of backed-up transaction output that is queued to be written might be lost. This is next to impossible to track, making the whole test run suspect. In fact, none of these cases reflect sound business process or transactional integrity, and therefore should be avoided.

Aborting the activities of a virtual user might be required in some cases, however. For example, if after executing a business process 30 or 40 times, your virtual user 197 gets hung or “stuck” in a transaction (due to poor scripting that did not take an error or other condition into account), you could very well lose all of your statistical data pertinent to that particular user—the orders created, materials moved, and so on. For one or very few users, this is easily dealt with. The output files reflecting activity for virtual user 197 can be purged or simply not considered in the final analysis, for example.

But more often than not, you will need an orderly and repeatable method of ending scripts. A few possibilities exist:

- Code each script, or set up each package of scripts, to only execute a finite number of times. This represents a method of testing in and of itself, in fact, in that the test becomes more of a race against time than an exercise in throughput.

- Code each script to check the status of a system-wide or other variable. If the contents of the variable in a special variable file are “true,” end the script. To make the contents of the variable true, and therefore end the test, I suggest copying a new variable file (with the variable already set as true) out to the file share or other location where it currently resides and is accessed by the various scripts (in effect, copying “over” the current variable file). This approach may be faster than trying to edit the contents of a variable via AutoTester’s Variable Panel (or a similar tool’s method).
- Similar to the approach described in the preceding paragraph, code each script to read the contents of a “status” file; if the contents say “stop,” then execute the SAPLOGOFF.aof script to stop the test.

I’ve used all three approaches with varying degrees of success. The last method can be troublesome, however, if a shared file is used for more than something like 100 users. This is because of the open and read activity that takes place, and resulting “lock” that one user places on the file. Subsequent users can’t get to the file, and thus sit around and wait for access. If the wait is too long, these scripts can abort. In my experience, the *best* way around these locking issues is to create a unique “status” file for each virtual user, like status.0197 for virtual user 197. Doing so eliminates file-contention issues, but brings up new issues including how to manage the process for quickly changing the contents of 1,000 status files. I have used simple batch files to rename the current status file (one that does not contain the word “stop”) to another name, and then renamed a file containing the word “stop” to status.0197—this works out quite nicely, avoiding every problem that I have ever come across in this regard. Other methods are certainly possible as well.

Additional Scripting Tips and Tricks

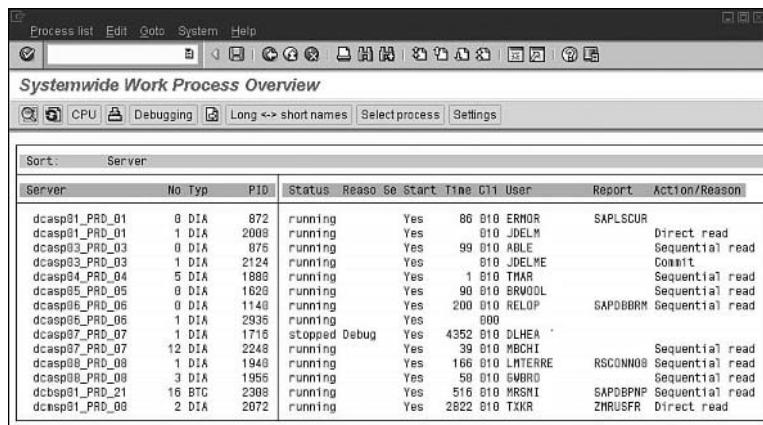
Throughout the years, I have come across or developed a few handy scripting tips and tricks that I believe will be beneficial to others. For simple “hardware delta” stress tests (where I want to compare the throughput of one SAP Solution Stack to another), one of my favorite practices is to avoid running transactions that *create* anything. It’s still very important to run transactions that issue *writes* to the database (like transactions that change sales orders, purchase requisitions, customer credit limits, and so on). But by avoiding inserts, you avoid making the database larger and therefore eliminate much of the need for restoring the database to a known state before each stress test run. This saves a lot of time on a couple of fronts—you don’t have to devise a process for warming/populating the database, nor do you have to take up time restoring the database to a known state before each test run—and makes a lot of sense for certain load-testing scenarios.

I’ve also learned to use the random number generators that ship with different load testing tools. Some of these are quite good, in that they generate a different number

or “seed” as expected. Others tend to generate the same sequence of numbers, which is therefore not random in my eyes, but can be used for predictably unique sequences. I highly recommend developing your own random number generation process, too. Like the serial number example I gave earlier in this chapter, a random number can be easily created by simply concatenating a number of values that change constantly. I often use the virtual client’s unique number along with the numbers associated with the current time and date to create a truly random number.

I’ve also become a big fan of scripting Basis transactions that can help me automatically gather test statistics. Like transaction ST07 discussed previously, transaction ST03 is another excellent tool for gathering post-run response time and throughput metrics associated with a particular test run. In the past, I’ve coded ST03 in combination with SM51; using SM51, I can select a specific application server (like the top one in the list). I then follow this up with an ST03 to gather specific dialog steps processed for the run as well as average response time, wait time, load time, roll time, database request time, enqueue time, and so on. After collecting this application server-specific data, I simply run SM51 again and choose the next application server in the list. In all, this is just the kind of data that truly proves an SAP system is ready for prime time, or that one SAP system outperforms another.

In running stress tests, I noticed long ago that it took a whole lot of end users to create the load I needed to generate (as observed in SM50 or in system-wide detail in SM66). As explained earlier, these transactions allow you to display the status of dialog, update, and other work processes, as shown in Figure 16.7. The following list describes some of the ways I have artificially and consistently created a greater load on a system during stress testing:



The screenshot shows the SAP Systemwide Work Process Overview interface. The title bar reads "Systemwide Work Process Overview". Below the title bar is a toolbar with icons for Process list, Edit, Goto, System, Help, and various system status indicators. The main area is a grid table with the following columns: Sort:, Server, No, Typ, PID, Status, Reason, Start Time, C11 User, Report, and Action/Reason. The table lists 18 processes across different servers (dcaasp01_PRD_01 to dcaasp01_PRD_08) and types (DIA, BTC). The processes include various SAP modules like SAPLSCUR, SAPDBRM, and ZHRSFR, along with reasons like Direct read, Sequential read, Commit, and Sequential read. Some processes are running, while others are stopped or debugged.

Sort:	Server	No	Typ	PID	Status	Reason	Start Time	C11 User	Report	Action/Reason
	dcaasp01_PRD_01	0	DIA	872	running	Yes	86 010 0100	SAPLSCUR		
	dcaasp01_PRD_01	1	DIA	2009	running	Yes	810 010 0100	DLHEM	Direct read	
	dcaasp03_PRD_03	0	DIA	875	running	Yes	99 010 0100	ABLE	Sequential read	
	dcaasp03_PRD_03	1	DIA	2124	running	Yes	810 010 0100	JOELME	Commit	
	dcaasp04_PRD_04	5	DIA	1882	running	Yes	1 010 0100	TMAR	Sequential read	
	dcaasp05_PRD_05	8	DIA	1629	running	Yes	90 010 0100	BRWOL	Sequential read	
	dcaasp06_PRD_06	0	DIA	1148	running	Yes	200 010 0100	RELOP	Sequential read	
	dcaasp06_PRD_06	1	DIA	2935	running	Yes	800 010 0100			
	dcaasp07_PRD_07	1	DIA	1715	stopped	Debug	4352 010 0100	DLHEA		
	dcaasp07_PRD_07	12	DIA	2248	running	Yes	39 010 0100	MBCHI	Sequential read	
	dcaasp08_PRD_08	1	DIA	1949	running	Yes	166 010 0100	LINTERRE	RSCONN008	Sequential read
	dcaasp08_PRD_08	3	DIA	1956	running	Yes	59 010 0100	SWARD	Sequential read	
	dcaasp01_PRD_21	16	BTC	2308	running	Yes	516 010 0100	MRSNI	SAPDBPNP	Sequential read
	dcaasp01_PRD_08	2	DIA	2072	running	Yes	2822 010 0100	TXKR	ZHRSFR	Direct read

FIGURE 16.7 Arguably one of the most powerful cross-application transaction-codes, SM66 is perfect for monitoring multisystem mySAP landscapes by monitoring active work processes.

- Use “noise” scripts. Some of my clients call this *background noise*; regardless, the function is the same—by executing a core set of display-only transactions behind the scenes of your real stress test, you will realize much more activity in the system and create a more-representative stress test. Noise transactions that I have scripted in the past include VA03 (display an order), MM03 (display a material), PA03 (display an employee record), ME23 (display a purchase order), and quite a few other read-only transactions.
- Similar to noise scripts, execute a predefined set of long-running queries or reports behind the scenes. A customer’s custom “Z” reports can prove beneficial in this regard, and truly help exercise an SAP system.
- End a transaction by executing the initial transaction again. For example, if you execute a VA01, you’ll go through three or four screens that need to be filled in with input data and processed. At the tail end of the transaction, an order will be created. Run the subroutine to gather your statistics followed by incrementing the loop as usual. I then suggest simply running the core VA01 transaction again, only this time refrain from going through the additional screens. Instead, execute a “/n” and repeat the loop as normal (starting *again* with VA01). The end result is that the load on the system is increased in a regular and repeatable manner, without introducing new or unusual transactions.
- For really basic load tests, especially in the case where no user data is yet available, it’s possible to gain a respectable amount of value from stress testing by executing a set of scripted Basis (as opposed to functional) transactions. These kinds of tests best support hardware delta testing, which I spoke of previously. Comparing one system to another can be accomplished by measuring total dialog steps processed for a finite run (of say 30 minutes), or calculating average response time, or simply monitoring the number of times a particular loop of transactions executed. Typical Basis transactions that are available even before any developers customize an SAP system include ST02, ST04, ST06, ST07, OS01, SM50, SM51, SM66, DB02, RZ10, and a host of others—the idea is to find a set of transactions that are not only relatively easy to script, but also that make database calls or CPU requests, like OS01 depicted in Figure 16.8.

For scripting tools that are not SAP-aware, another useful practice is to drive the SAPGUI using “keyboard” commands as much as possible, instead of performing mouse clicks. That is, if you can navigate via the keyboard to the button you need to click, or the radio button you need to click, you tend to get a more reliable script. And because many of the screens in the SAPGUI support function keys as well as “mousing,” it’s not too difficult to find keyboard shortcuts. While scripting, right-click the background of the SAPGUI to see the available function key shortcuts for that particular screen.

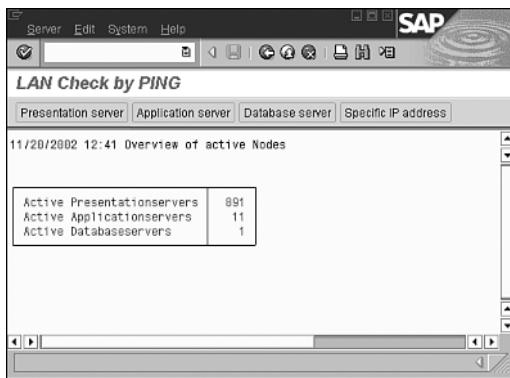


FIGURE 16.8 Though simple, CCMS transaction OS01 provides powerful up-to-date rolled-up user data, and in doing so creates a predictable load.

Finally, yet another handy trick I picked up a few years ago involves using the number of seconds in the current time (0–59) as a way to “randomly” determine which transaction should execute. This is useful in umbrella scripts, which as I mentioned before are scripts that essentially execute other scripts based on a set of criteria. I use umbrella scripts and information provided to me by my customers to set up distributions. For example, if my customer tells me that 10% of all scripts should execute FD31, and 30% should execute VA02, I set up if-then logic in my master umbrella script like “if time = 00 through 05, then execute transaction FD31.” This represents 6 of 60 possibilities, and therefore 10%. Similarly, “if time = 06 through 23, then execute transaction VA02.” This represents 18 of 60 possibilities, of 30%. As you can see, it’s granular enough to handle different percentage loads. And if you create AutoController packages of 60 virtual clients each, you are assured of getting an excellent distribution (assuming you use the staggered login approach I explained previously).

One of the appealing things about scripting or coding is that there are usually many ways of solving a problem. I look forward to hearing about different methods that you will ultimately use to solve your own stress-testing scripting challenges. And I hope that some of the scripts discussed here and included on the Planning CD give you a framework or at least a few ideas as you seek to plan for and complete your own scripting.

Final Preparations Before “Test Week” Commences

By the time “Test Week” draws near, all of the gear required to complete the stress test should be in place. That is, the production system or system to be tested must be in-place and locked down (in terms of change control) as much as can be expected at this time. All virtual test clients—physical desktops or servers, software, and everything associated with driving virtual clients—must be in place. Any special monitors

or consoles to be used for monitoring must also be set up. For example, I require at least two desktops from which to execute SAPGUI sessions, OS-monitoring tools, and so on. Finally, the AutoController (or whatever product you are using) stress-testing console needs to be installed, configured, customized in terms of packages/scripts, and tested. This will help ensure that Day 1 of Test Week is not wasted performing pre-test-week tasks. Without further adieu, on to Test Week!

Stress Test Execution During “Test Week”

With the big week finally upon you, there is much to do, from preparation to final review of both the system to be tested and the supporting test infrastructure, to last-minute assessment of scripts, data, and more. By the time this is done, you probably have time to run a test or two before lunch. I suggest performing a reboot of all servers involved in the test—everything, including the system to be tested and the systems driving the test—prior to officially starting a “run.” In this way, all caches are flushed, everything comes up “clean,” and a very repeatable test foundation is established. Bring down the test infrastructure first, followed by the test system’s ITS or other Web servers, and then the test system’s application servers, and finally the database server. Start everything up in the reverse order—database up first, then application servers, then ITS and other Web servers, and then all of the test infrastructure client drivers, and finally the test consoles.

Next, start the various monitors you will use to collect data on the run. Start two different SAPGUI sessions, kick off PerfMon or start your UNIX utility of choice, and so on. Note the wall clock time on your test run checklist, give the run a name, and then start the actual stress test run by releasing the first AutoController package. This starts your ramp-up period. Be certain to release each package in the same sequence and using the same time intervals each time you start a run, and note any deviations. I strongly suggest recording the sequence and timing in your checklist, to make sure there is little room for error. Why? Because even a couple of minutes difference in starting hundreds of test users will result in different throughput numbers, and generally skew the test run when compared to other test runs.

Leveraging Your Testing Tools

With the first test started, you need to begin monitoring the run immediately. I recommend focusing first on the testing tool’s console (AutoController, for instance), to ensure that the scripts are indeed running and that virtual clients are indeed being generated and making a connection to your SAP system(s). While monitoring this activity, don’t forget to continue the ramp-up process, though! In my experience, it’s common to release between 5 and 10 different packages in the course of 15 to 30 minutes, while simultaneously monitoring the test run.

Monitoring the Stress Test via SAP Transaction Codes

With your SAPGUI and a number of dedicated console sessions per *mySAP component*, you will monitor the activity generated by each virtual user. The initial thing to look for is the system activity spikes associated with each virtual client logging in to each system. CCMS transactions ST07 and AL08 are perfect in this regard, with the latter transaction providing the added benefit of showing you how well your logon load balancing system is working (AL08 displays end users sorted by the application server each has logged into).

When the logon spikes have subsided and the test run has approached the end of the ramp-up period, another set of CCMS transactions become valuable:

- SM66, to monitor how active a particular system is across all application servers and the Central Instance. SM66 displays all active work processes—database updates, dialog, background, and so on.
- ST06, to monitor disk queues, processor/system utilization, and memory/paging statistics.
- ST04, to verify that procedure and data cache hit rates are high or otherwise acceptable.
- ST22, to monitor ABAP dumps.
- ST11, to review error logs/traces as necessary.
- SM21, to review the system log.
- ST03, though don't worry about collecting too much data during the actual test run. Instead, I execute one or two ST03 transactions to simply ensure at a high level that actual response times are in the ball park.

While monitoring your stress test via the SAPGUI and CCMS, don't forget to continue to look at the status of the stress test as identified by the tool's monitor. It's not uncommon to "lose" a couple of virtual front ends, for example. However, this needs to be noted so that its impact can be factored in after the stress test run has completed.

Operating System Utilities

Although the virtual test tool and the SAPGUI/CCMS are vital to monitoring a test run, simple utilities like Microsoft's PerfMon and UNIX's command-line utilities are equally important. PerfMon will gather much of the statistical source data that will later prove useful in analyzing the real performance of your test runs, for example. And this PerfMon data also helps you troubleshoot and explain anomalies that might crop up during your stress test.

The easiest way to ensure that you gather the statistics and data you might need is to collect *everything*—all performance counters associated with each server. But if it becomes necessary to minimize your data collection, I suggest gathering all CPU, system, disk, and network-related information. And don’t forget to be consistent across the board. The same data should be collected for and available from each test run. Not doing so risks being unable to answer questions or drill down into specific data in a specific test run. And more importantly, running different collectors actually impacts the performance of the system—collecting more statistics impacts the system more, underscoring the fact that the same process should be employed and data collected for each and every test run.

Using Test Output for Continuous Improvement

One of the neat things about a test run’s output is that it can be leveraged to tune your pre-production system “on the fly.” Further, the data can then be used for comparisons, allowing you to measure the real value gained in the tuning/changes made to the system simply by executing another test run. It’s a great process, and certainly one of the best ways to tune a system simply because the load placed on the system is so predictable and repeatable.

For this reason, collecting data expressly derived from a stress test run is a must—you cannot afford to “forget” to collect your data, or to lump the test results of many test runs into a master file or another general repository. Do your best to keep both your test runs and your test output data discrete. To this end, I suggest creating a directory structure where each test run is given a directory, and underneath each test run directory are separate folders for input, scripts used, and output data. Copy all of the relevant test-related figures into their appropriate folders. I also create a simple checklist like the one that follows to help me remember what data to collect:

- Details related to the complete “load tested” solution stack configuration. Of critical importance is to note any changes (compared to the previous run) that were made to the stack, test tool, or approach.
- Average ST03 response time statistics for the test run (with all low-level details), especially dialog and update work process response times. Include background response times if applicable.
- Dialog steps processed per application server (also available from ST03).
- The output files created from each script, which should record input data and output/results for each transaction executed during the test.
- Total number of transactions executed (sorted by whatever criteria makes sense in your particular case). This should correlate pretty well to the number of transactions processed.

- The number of virtual users that executed on *average* throughout the test, as well as the number of virtual users at the end of the test. Note that there is usually a drop-off of a few users during any test; however, I void a test run if I lose more than a few percentage points.
- OS-collected statistics identified previously. And don't forget to copy the PerfMon or other utility's logon to your test run directory structure. In some cases, you might even want to add event or error logs to the output test directory.
- Add to this any data collected manually, like regular "spot checks" recording CPU utilization or average disk queue length and so on.

With all of this data collected, it is then necessary to go through the process of preparing for the next test run. Sometimes this requires restoring the "test" databases for each mySAP component. In all cases, I recommend rebooting and restarting each system as previously described to clear all cache and start the next test run with a clean slate—in this way, caches will be predictably populated during the ramp-up period of the next test.

Additional Stress-Testing Goals

When you have made the initial investment in a stress test, and have the infrastructure and scripts in place to accurately simulate how your production environment will behave after Go-Live, it's very tempting to take this investment one step further and begin doing some additional value-add or just plain out-of-scope stress testing. In my experience, the most common additional test-runs performed outside of the core goal-driven testing include

- Playing "what if," like losing an application server to see the impact that its absence has on load balancing
- Verifying that your failover solution works as advertised, and ensuring that specific failover scenarios operate as expected.
- Ramping up the system to reflect excessive loads; this might include increasing the number of online users, reducing the think time of these online users, increasing the number or intensity of batch processes, and so on.

Certainly other scenarios can be tested. I suggest that these be discussed well before test week if possible, in case any prior preparation or research would prove beneficial in setting up or executing these test runs.

Playing “What If”

Taking a “what if” approach to additional stress testing is a lot of fun. If your company is considering an acquisition, in the middle of bringing more users into the system, or simply interested in the impact an MRP run has on the daily production load, these tests are valuable. I believe the key to a good “what if” test is guidance and some level of assistance or preparation from the business units. In other words, you want to understand *exactly what* might be valuable to script, simulate, or test manually. This implies working with the functional or business teams to understand their business processes. You can then add additional “layers” of testing by folding in some of the tests discussed in the next few sections. For example, understanding the impact of an MRP run during a typical day might just be the beginning for you. To really gain valuable insight, you might further want to see the system fail over from one database node to another, or from the primary data center site to the DR site, while in the middle of all of this processing activity. Additional hardware-centric or high-availability-specific “what if” scenarios are discussed next.

Verify System Redundancy and Failover Perform as Expected

I have had a lot of personal experience with this type of value-add stress testing. The possibilities are endless, but the following scenarios are typical:

- Knock out power to a database cluster node, redundant application server, no-single-point-of-failure disk subsystem, or a dual-redundant network switch, to verify that the system remains up and available, or reacts as expected.
- Simulate a failed disk drive. Speak to your hardware partner for supported ways to perform this test, keeping in mind that unplugging an actively running disk drive, even if it’s redundant or protected via RAID, may not be the best or supported method of testing disk drive failures.
- Actively fail your system over to the DR site, and then back to the primary site.
- If supported by your solution stack, hot-replace a “failed” component like a power supply or network card to ensure the process is both flawless and that it is documented clearly.

Finally, test the impact of losing an entire application server, which reflects how well your logon load balancing scheme works, how long users take to reconnect to the remaining application servers, and other performance and availability metrics.

Ramp Up to Excessive Loads

Ramping up your work load to a point beyond what you “expect” to see in production after Go-Live is a common goal of stress testing after the core goals of Go-Live viability have been met. Many of my customers look at this in terms of month-end,

quarter-end, and year-end processing. In each case, additional online users and/or batch processes are added to the test's distribution mix.

Measuring specific technology metrics like disk queue lengths or average CPU utilization can be useful too. For one of my customers, for example, we determined that the average database CPU load would be between 20–40%. Our goal was then to see how many users the system could host before average CPU utilization across the production landscape exceeded 80%. Other tests were crafted, focused specifically on driving the load on the database server, and then the central instance (which ran on a dedicated server by itself), then specific application servers, and finally a pair of dedicated batch servers.

In another large-scale SAP stress test, I monitored the disk queue lengths observed by different disk subsystem designs. Their legacy FCAL disk subsystem served as the baseline. A new SAN-based disk subsystem located at a different customer site acted as a target of sorts. Finally, after they received their new virtual array, we configured it consistent with different recommendations pulled from SAP's SAP Notes and from white papers published by the hardware vendor's SAP Competency Center. All of this resulted in three different configurations that we finally tested. We settled on the design and configuration that resulted in the lowest average disk queue lengths while still meeting the customer's high-availability and budget requirements.

In yet another case, I executed a set of stress test runs that pushed the customer's pre-production environment almost 180% beyond what would be observed in their typical day-to-day system load. Specifically, their stress test target was to hit 80 concurrent processes. After we achieved this goal, I reduced the think times in the functional scripts and then increased the number of virtual users executing these scripts. By the time I was done, SM66 revealed that nearly *all* (220 of 240) of their dialog work processes were actually being used (an amazing sight, by the way), and the quantity of work being performed was tremendous by all measurements.

The business value provided in each of these scenarios was the same, in that I proved that each customer's SAP Solution Stack was not only optimally configured (through iterations of tuning and testing), but also that each system exhibited a certain amount of scalability, or head room. Further, the testing revealed which subsystems or layers of the technology stack enjoyed the head room, and which didn't. By doing so, I also identified future performance bottlenecks that might one day become problematic. Certainly, as funds became available to perform upgrades over the next few years, each client would be in a position to more judiciously allocate those funds to reduce or eliminate future performance bottlenecks.

Extracting the Last Drop of Value out of Testing

True, the common goals of stress testing seem to reflect the immediate needs of the production user community—things like average response times, number of

supported users, speed with which batch processing is completed, and so on. However, I applaud those companies that take advantage of stress testing their pre-production environment to verify:

- That their backup and restore processes works as advertised
- That their disaster recovery plans are not a disaster waiting to happen!

Other value can be gained as well. For example, testing the ability of your newly selected Systems Management applications or utilities (and other monitoring tools and processes) to effectively meet the needs of the SAP TSO can represent an excellent testing investment, too.

Verify That Backup/Restore Meets Requirements

Certainly, the backup/restore process has already been tested somewhere in the SAP system landscape. What is important here, however, is that the actual B/R solution implemented for production SAP be tested. The significance of this can be easily demonstrated by looking at the real-world situations that have been thrust upon my colleagues and myself in the past, including:

- A new tape library was introduced in the production environment. Contrary to best practices, though, no other system in the SAP landscape utilized this specific model of tape library (or any tape library, for that matter). Instead, all other environments relied on direct-attached single SCSI 35/70 DLT tape drives. Had the library been tested during test week, it would have been clear that the combination of the backup application, OS drivers, and tape library was not yet ready for “prime time.” As it turned out, the customer found this out during an already painful week of Go-Live.
- In another case, a customer new to SAP also had to deal with learning the intricacies and differences inherent to managing and backing up data residing in a new Storage Area Network, or SAN environment. Maintaining at least one other environment identical to production was not a priority for these folks either. They learned their lesson the hard way when an issue cropped up during high SAN utilization—the database server simply lost its connection to its data files, corrupting the entire production database as it failed. A firmware upgrade resolved the issue, but a quick 30-minute stress test run would have uncovered this issue weeks before Go-Live.
- A long-time customer of mine decided to implement a clustered database/central instance scheme while going through a database upgrade. They clustered their technical sandbox, documented the process for both the database upgrade and the underlying clustered-supported hardware upgrade, and sent two key SAP support members to cluster training. What they failed to ever do, though, was run a backup to completion in their clustered sandbox

environment until a week before cutover. At that time, they discovered a bug in the backup software when installed in a cluster. The bug only manifested itself under load, and then only intermittently. A disaster was avoided, but confidence in both the backup solution and the clustering approach to high-availability was shaken.

If you are paying attention, you probably noticed at least two embarrassing oversights per issue. First, a stress-test would have revealed the flaws that were eventually found either too late or too close to actual Go-Live. Second, better change control practices would have brought these issues to someone's attention in a more timely manner, too.

Ensure That Your Disaster Recovery Plan Is Effective

As I discussed throughout the last few sections of Chapter 6, testing your disaster recovery plan is an important task that needs to be performed prior to Go-Live and then once a year afterwards. Most everyone agrees with this in principle, but when it really comes down to spending the money to test your DR system, too many CIOs and IT directors have *other* plans, and the DR test is "talked through" rather than actually executed. Talking through it is better than nothing, I suppose, but I have a better suggestion—take advantage of stress testing to actually test your DR process *in action* before you Go-Live. Later, when you are in production mode, take advantage of the downtime normally associated with quarter-end or annual change control waves to again test your DR process. This is appealing because it kills two expensive birds with one moderately priced stone. That is, the downtime is already scheduled, so adding another day or two of DR testing only takes advantage of the resources in place, and in my book equates to excellent return on investment.

Lessons Learned in the Real World

In the last few years, I have had the honor to assist countless SAP customers with designing and executing their SAP stress tests. It's an area of work that I really enjoy, and one that is as valuable to my clients as it is fun for me. The hours are long, but the rewards are great. So, too, have been the lessons I've learned—I hope the following list helps you not to repeat some of my own mistakes and oversights!

- I quickly learned the value of data in one of my first stress tests—more data is always better. In this particular case, I had so little transactional and master input data that within a few minutes of a test run, all of it was accessed and just as quickly cached by the disk subsystem. With next to no load on the disk subsystem, the entire test was tremendously skewed; user response times were phenomenal, transactions flew to completion, and the whole effort was wasted until we uncovered 100x more data.

- Don't try to execute a stress test too early in your SAP project implementation. Good data can be hard to find when you begin testing *too* early. SAP clients change as data is brought in or created, functional areas are updated, and business processes are revamped. This fact manifested itself in scripts that no longer worked a few days before Test Week, due to data combinations being invalid and SAPGUI functional transaction screens actually changing as a result of development. This brings up another lesson learned: As much as possible, lock down your client and configuration throughout the script development and test execution phases.
- Not all virtual users are created equally. Different testing tools exhibit different tendencies and strengths. Some force all users to log in to the system simultaneously, whereas others exhibit a pattern of repetitive activity. This can all be resolved programmatically, but requires time. Thus, do not wait until the last minute to actually test your scripts in a load-testing manner—follow the process I outlined earlier in this chapter, where scripts are tested for functionality, then single-user tested, 10-user tested, and 100-user tested.
- Think times need to be considered and *verified*. If a system is designed or sized to host 1,000 concurrent medium users (therefore implying a 30-second think time), you need to be sure to include the appropriate think times in your scripting efforts, and then verify that actual "wall clock" time and your programmed think times don't contradict each other. This is especially true as a load is put on your system. Consider the following example—after scripting a VA01 for a customer, I noted in my single-unit testing that the end-to-end process took three minutes to complete. With wall clock time of three minutes, and direction by the customer to execute 12 sales orders every hour (or one every five minutes), I figured I needed to hard-code two minutes worth of think or delay time. When I had 200 of these SD virtual users executing, however, the average wall clock time jumped to more than four minutes. I therefore had to scale back my think times in a big way; otherwise, I would have missed my target of 12 orders per hour per user by 2 orders or so. At 200 SD users, that would have equated to 400 missing orders!
- The performance and monitoring data that is derived from a stress test run needs to actually be *looked at and analyzed before another test run is performed*. My customer and I wasted almost two days performing a series of tests only to discover that more and more of the core transactions were not completing as the user load was increased. Had I looked at the data earlier, it would have been clear that we had a problem (as I had coded an obvious "transaction unsuccessful" message in the error routine used by the scripts; the test's output files were filled with evidence of this irritating little fact).
- In one of my first stress tests, we proved that a customized R/3 Sales and Distribution implementation was poorly coded from an ABAP perspective. In

another, it was uncovered that the customer's reporting system was misconfigured (it defaulted to pulling in all data from all business units by default), bringing the entire system to its knees. Only under load were these problems obvious, though. Both problems were easily rectified (in the latter case, the users were forced to select specific parameters like profit center, cost center, and plant). But uncovering these issues was not the explicit goal of load testing in these cases, thus underscoring the unexpected extra value that can come out of a properly executed stress test.

Some of my favorite stress tests have been in support of my own long-time customers performing upgrades and updates to their in-place SAP production systems. These are usually the least time-consuming to perform, and often revolve around the introduction of new server or disk subsystem platforms, or a fundamentally new database release. I tend to see a lot of activity in this regard when new or substantially updated processors are released, or a new wave in disk subsystem technology drives massive database upgrades, or a new RDBMS is finally made public. And of course, with a new version of a core mySAP component or enabling technology introduced quarterly, I expect to enjoy many additional stress tests in the years ahead!

Tools and Techniques

In addition to the various screen shots and other figures from this chapter, I have also included actual working scripts in a number of different scripting languages. Some of these scripts serve specific purposes, and others simply demonstrate how to achieve a particular outcome. Further, status reports, checklists, and other documents facilitating the installation, management, and execution of a stress test are included as well.

Summary

Throughout Chapter 16, I focused on the tasks related to planning for, scripting, executing, and monitoring an SAP stress test. Key testing tools, approaches to stress testing, methods of gathering input data, and the work of identifying and gathering post-run statistics were covered, as were lessons learned, coding tips and tricks, and more.

In the next and final chapter, we will look at the remaining tasks that ultimately prepare the production environment for its critical role in supporting a company's end users. Thus, verifying output management, locking down the system, preparing for the first change control wave post-Go-Live, ironing out support and service level agreements, preparing the SAP technical support organization for the big day, and so on are all covered in Chapter 17.

17

Preparing for SAP Go-Live

Overview—Preparing for Cutover

The big day of Go-Live is finally close at hand. Go-Live is simply a point in time, however, or the milestone of greatest importance in your implementation or upgrade project. But it's the *process* of transitioning from one system to a new one that is the final challenge related to going live on a new system. This process is often referred to as *cutover*, and is the focus of this final chapter.

Critical cutover-related tasks occur well before the big day of Go-Live, and the planning and preparation associated with these activities occurs even further back. Project milestones related to locking down the system from a technical change management perspective, or preparing the SAP Technical Support Organization for its new role and responsibilities, and even rolling out the SAPGUI to all future end users are all key to achieving Go-Live success. And of course, the huge task of completing the functional customization of the system, synchronizing data between systems, and many other tasks and activities must all be accomplished. These high-level technical and business tasks, and other specific activities, make up your *cutover plan*.

The Cutover Plan

Planning, preparing, and executing cutover should all be documented and managed through the use of your project plan. Why so formal an approach? Because there's so much to do, and little room for error. Nearly every

IN THIS CHAPTER

- Overview—Preparing for Cutover
- Preparing for Technical Go-Live
- Final Administrative Technical Details
- The Changing Role of the SAP TSO
- Preparing SAP Operations and the Help Desk
- Addressing Future Service and Support
- The First Week of Go-Live
- Tools and Techniques

member of the SAP TSO is involved, and all of them need to be coordinated in terms of tasks, milestones, and working through critical-path objectives. If we take a quick look at the SAP Operations team alone, the responsibilities are huge in that even this low-profile team must

- In keeping with the company's change control processes, work with the technical and functional teams to lock down the production system (described in more detail later in this chapter).
- Work with the enterprise management group and basis team to update or reconfigure the enterprise management tools to be used for monitoring production.
- Obtain any last-minute training regarding ongoing operations.
- Review and refine/update all systems-related operations procedures and processes, and ensure that they are clearly defined, understood, and documented. For example, backup/restore, system monitoring, event management, and escalation of issues are all core subtasks that fall under the umbrella of systems-related operations.
- By the same token, ensure that SAP functional operations tasks are addressed as well. This might include defining your "daily routine" procedures, monitoring and managing batch jobs, addressing on-call support needs, and so on.
- Assign ownership of all of these processes to individuals within the operations organization; allow SAP operations to be self-sustaining in the future rather than relying on system administrators to do what actually amount to operational functions.

Event escalation in particular needs to be clearly understood. SAP operations must document which team is contacted (and in what order) for each failure scenario. Typically, an on-call rotation list needs to be disseminated by each SAP TSO group to the operations team. And within each team, or specific to a particular error or condition, the "event management process" needs to be developed. This includes developing an incident-reporting process, a mechanism or tool for tracking and documenting incidents, and a method of reporting. At one of my SAP customer sites, the event management process has been broken down into discrete steps, such as escalating the event to the correct team or person, establishing a conference call with all relevant parties, documenting and tracking an action plan, and building a custom communication process into the company's management/stakeholder teams.

But SAP Operations actually has it easy compared to many of the other groups within the technical support organization! For example, consider the following high-level milestones that must be managed by the Client SAP project manager, each

milestone consisting of anywhere from a handful of tasks to 10 or 20 discrete supporting activities touching multiple SAP TSO groups:

- The SAP Help Desk and other core support teams must be established, staffed, trained, and prepared for their role in supporting Go-Live if this has not been done already.
- After all post-stress-test tuning changes have been implemented, the production system must be frozen in terms of technical or solution stack changes.
- At a functional level, all transports, customizing settings, and so on related to your SAP component's business processes must be completed and then frozen. Business cutoff dates need to be established as well, and signoff on these dates/milestones by each functional group needs to be obtained (the signoff process keeps anybody from later complaining that the cutoff date was too early and they "didn't know" or had no control).
- Any new/remaining and necessary master data needs to find its way into the new system; creating the data or performing bulk load inserts from a source or legacy system are two ways of handling this.
- Further, this master data may need to be transformed or otherwise changed in some cases (for example, in support of populating SAP BW cubes, or preparing the data for use by SAP APO).
- It is necessary to "freeze" all legacy systems some time prior to Go-Live, so that the final *transactional* data from these existing systems can be captured and migrated to the new system. This can be done once, prior to retiring the old system, or performed in parallel to keep the old and new systems tightly synchronized for a period of time. Anything from scripting bulk data loads to performing manual data entry may be necessary. And of course, all of this activity must be carefully communicated to end users, too.
- All mySAP end users must be trained and knowledgeable as to the support resources available to them (for example, their super-user colleagues within their own functional organizations, the SAP Help Desk, key "backup" or after-hours folks in other SAP TSO organizations, and so on). See the Planning CD for a document that covers how end users should leverage the SAP Help Desk/Business Support Organization in preparation for Go-Live.
- A process for actually measuring the system's availability and performance against its Service Level Agreements needs to be developed. Sometimes called *availability tracking*, it includes defining what does and does not constitute downtime, what constitutes acceptable performance, how to track planned and unplanned downtime, how to measure availability, how to schedule downtime, and finally who will own the tracking process. A number of methods useful in organizing and communicating this data can be found on the Planning CD.

- Coordinating and validating an *operational review* that determines the readiness of the system is key, too. An operational review seeks to confirm technical, functional, operational, and other readiness factors. For example, an operational review of the data conversion process must be performed to ensure not only that the data itself is good, but that the process used to populate the system is auditable. And the readiness of each system process or component must include signoff based on agreed-upon readiness criteria.
- Tasks related to vendor and partner support and maintenance/management contracts must be carried out. For example, maintenance contracts, repair processes, access to consulting resources post-Go-Live, and so on all need to be addressed.
- Finally, a review of the project plan against your implementation methodology of choice must be performed to ensure that no stone has been left unturned. The SAP Solution Manager maintains roadmaps like the *Enhanced Solution Management Roadmap*, which details the technical implementation. In the same manner, the *Enhanced Implementation Roadmap for mySAP.com* targets project managers and your functional teams. The specific tasks associated with older and still valuable implementation methodologies—ValueSAP and ASAP—can be reviewed as well.

Of course, none of these activities can occur in a vacuum. Cutover must be underpinned by sound communication and reporting processes, not only for the obvious coordination requirements but also so that no stakeholder is surprised at the last minute by a condition or issue that has existed for some time but has yet to be resolved.

In a nutshell, then, these tasks can be summarized as follows: communicate with your stakeholders, lock down the system, address SLAs, prepare your support personnel, plan for contingencies, work through last-minute issues, train everyone, make sure you didn't miss anything, and do your best to control the chaos bubbling underneath the surface of your project. The details relevant to these tasks are discussed throughout the remainder of the chapter.

Preparing for Technical Go-Live

At least a month before Go-Live, all of the technical tasks related to updating the system earmarked for production need to be addressed. The most critical of these tasks surrounds the issue of change control; any change that must be implemented into production needs to advance down the change management path *now*. You want to ensure that each change has ample time to settle in and prove itself within your SAP system landscape. And just as importantly, you need time should one of the changes lead to an unstable SAP Solution Stack; backing out a change, or researching new updates/changes and updating the stack, takes time.

As outlined in the ASAP methodology, you'll also need to focus on last-minute *production support*, which is another way of saying that last-minute issues need to be managed and resolved, and the results of live business processes need to be validated. ASAP never allotted much time for these activities, however; in my experience, nothing less than a calendar month makes sense.

Of course, a great many additional tasks need to be accomplished prior to Go-Live, too, from fine-tuning your "golden" production business client to completing any final DR site updates, to reviewing the system from top to bottom, to finalizing documentation, and so on. The bulk of this chapter is focused on these activities and more.

SAP GoingLive Check and Other Review Processes

A key task associated with locking down the system from a technical perspective involves leveraging SAP's "proactive and predictive services" offerings. The first of these, the *GoingLive Check*, is usually performed a number of times before Go-Live. Normally, one is performed prior to your stress test (in support of pre-tuning and basically ensuring a sound solution stack), and another a few weeks before Go-Live. This final Go-Live check ensures that you have SAP AG's "blessing" to bring your system live; indeed, it is in SAP's best interests as well as your own to smooth the way to post-Go-Live and the continuous improvement phase of your mySAP implementation.

After Go-Live, you'll still want SAP AG to audit and review your system occasionally. This is performed remotely, onsite, or as a self-service procedure, and is termed an *EarlyWatch service* or *session*. EarlyWatch services have been a mainstay of SAP AG (and select technology partners) for years, and help to ensure that your solution continues to run well even in the wake of changes to your business processes or infrastructure. The latest development regarding EarlyWatch involves the SAP Solution Manager. Specifically, SSM's proactive "solution monitoring" capabilities are designed to prevent critical situations before they occur. Conditions surrounding poor performance, high workloads, and low database free space will trigger an EarlyWatch alert, drawing the SAP TSO's attention to areas it may need to watch more closely. This alert information can be collected and later analyzed to support real-time analysis (where time intervals vary from minutes to hours) to SLA reporting (where time frames span days to weeks and longer).

Today, included with your mySAP system maintenance, SAP AG provides two EarlyWatch sessions per year. Most of my customers have these executed via "remote delivery" by a SAP service engineer working over an RFC connection to your system. I recommend that you use the first of these shortly after your first post-Go-Live change control wave. Use your second EarlyWatch session as you think necessary; in support of performance issues, or after major changes to your system, are both appropriate uses.

SAPGUI Rollout Mechanism

Prior to Go-Live, each future end user of the new SAP system needs to enable their desktops, laptops, and other access devices to communicate with the SAP system. I use the term SAPGUI generically; as we have discovered, there are a number of ways to access a mySAP component, including Java-based and HTML-based WebGUI interfaces.

The simplest method for small or phased implementations is to make the SAPGUI Front-End CD available via a file share or similar method, and point your users to it. You can leverage logon scripting to automatically install the SAPGUI, too, though I generally prefer to leave this up to the users as the installation takes upwards of 30 minutes depending on your network link and the processor speed of the front-end client.

I have seen more and more SAPGUI deployments enabled through Citrix, Windows Terminal Services, and similar means. Where applicable, hand-helds, palm computers, and other wireless devices need to be enabled for SAP as well.

On the other end of the technology spectrum, it's still not uncommon to see PC Support organizations tasked with visiting the desktop of each user, and performing the SAPGUI installation based on a standard installation procedure. It's amazing, but certainly not uncommon. If this is your preferred method of rolling out the SAPGUI, I also suggest that SAP default printing and faxing services be set up during this visit as well.

Before we leave the topic of rolling out the SAPGUI, you also want to ensure that your standard desktop or laptop "image" includes the SAPGUI by default. In this way, as your IT organization deploys new or replacement computers throughout the company, your users will not be forced to install the SAPGUI manually. Like imaging your office productivity applications and so on, including the SAPGUI in your master image saves time and reduces complexity and support costs in the long run.

Setting Up Batch Housekeeping Jobs

Though you have probably already set up the core housekeeping jobs as part of your unique SAP component post-installation process, a certain number of other "house-cleaning" or housekeeping jobs must be scheduled. Review the jobs you currently have scheduled, including the execution status of any job, by running transaction SM37. To add standard housekeeping jobs to new instances, execute transaction SM36 and then click the Standard Jobs button (see the Planning CD for a more detailed procedure). You can verify that no single job is scheduled to execute twice at the same time from here as well.

Additional jobs beyond the standard ones can now be set up. These can always be executed manually, but I find SM36 helpful when it comes to scheduling housekeeping jobs that should run regularly, or jobs that don't necessarily map well to a traditional calendar. For example, if you need to run a job every Monday unless Monday is a holiday, you can select a more restrictive execution option, such that if the day falls on a non-working day (according to a factory-based or other calendar), you can configure the system to "Move job to next workday."

What exactly are housekeeping jobs, though? The answer depends on the particular SAP product that has been implemented:

- For Knowledge Warehouse, job RSIRIBUF needs to be scheduled to delete buffered documents that have expired, and RSTIRIDX indexes flagged documents and logs errors.
- For CRM, data sent by a mobile client device is not actually updated in the back end until job WAF_MW_MAPPING is executed.
- For R/3 4.6C basis layers and higher, Smart Forms may terminate with a "time out" during execution unless SF_BATCH_GENERATE is run (note that Smart Forms replaced SAPscript a while back).

In other cases, jobs may be more generic. For example, job RSBDCSUB may be executed to process a batch input session. And ZTERRDEL should be set up to run periodically, to clean up application data left undeleted in table TERRD—these are both good examples of cleanup jobs. Other jobs may be used to prepare for executing another particular job, or to report against the status of a specific job, component, condition, and so on. I suggest that you carefully review the Installation Guide, Master Guide, and other core documentation for each mySAP component you have installed. Pay particular attention to the SAP Notes referenced by each guide, and review these as well. You will also want to look at SAP Notes that are specific to the Basis release on which your mySAP components reside (4.6C or 6.30, for example). And in all cases, refer to SAP Note 16083, "Standard jobs, reorganization jobs" to determine whether any new housekeeping jobs are recommended—as SAP's NetWeaver platform/initiative grows to enable greater enterprise connectivity and interoperability, I speculate that more and more standard housekeeping jobs will become the norm.

Final System Updates and Review

A number of weeks prior to the estimated technical lock-down date, the SAP TSO Basis or Technical Infrastructure team needs to carefully review the system that "mirrors" production (for example, Staging or Test/QA). When the mirrored solution stack has proven itself to be reliable, only then can the following final technical changes be applied to production:

- Promote the final support packages, SPAM, SAINT, and kernel updates, database patches, OS service packs/patches/hot-fixes, and so on into the production environment.
- Similarly, this is the time to perform any updates to add-ons, such as the R/3 Plug-ins. As with everything else, take care to ensure that add-ons are not rushed through the promote-to-production process, as only specific releases of add-ons are supported by specific mySAP components.
- Perform an additional client copy of your golden client, and keep it available “just in case” and in keeping with best practices regarding data safekeeping.
- Any final tweaks to your SAP Profiles need to be made now, too.

Remember, I recommend obtaining your final SAP GoingLive Check at this time. I also strongly suggest that you quickly follow up these changes with a cold or offline tape backup, and ship the backup offsite for safekeeping. For Windows-based systems, back up the registry of each server in your production environment(s) as well. And remember one of the goals of making these final changes to production—you seek consistency between two or more different systems. That is, by the end of this final change management process, your production environment should be *identical* to your staging, test/QA, technical sandbox, or some other environment within your SAP system landscape. The less identical these environments, the more risk you incur should additional updates or changes be required by the system one day (which is inevitable!), as you will not truly have the benefit inherent to testing changes in an apples-to-apples manner.

Locking Down the System

When your offline backup is safely offsite, take yet another tape (or disk-based) backup and keep this one onsite and close by, in case you need it prior to Go-Live. And lock down the system. In other words, do not make any more technical changes to it. This includes applying last-minute patches and updates, flashing firmware, restringing your SAN fibre cables, swapping out network cables for color-coded ones, and so on. You should even refrain from creating new clients or doing anything else that could potentially impact available disk space. I've seen all of these little things impact the system on the day of Go-Live. As we were nicely reminded in the Marine Corps Recruit Depot in San Diego after being instructed to halt, “You’re done. Lock it up! Lock it up!” In essence, communicate to everyone on the technical team the same thing—No Changes Allowed. Feel free not to scream.

Preparing for the First Change Management Package

It might seem strange to be already thinking about making changes to the system, but it's actually very normal; when the system is locked down in preparation for Go-Live, all potential updates need to be queued for the first change control wave. In

my experience, changes still seem to get put in even days before Go-Live, but this practice is certainly not a *best* practice, and it is to be avoided.

Instead, the SAP TSO needs to begin embracing a mindset that says only absolutely critical changes will ever have the opportunity to be implemented rapidly, and then only after having been promoted in an orderly fashion through the SAP system landscape. Some changes will grudgingly be promoted quickly, if they greatly impact profitability or customer satisfaction, for instance. But the bulk of changes will be stored up, to be applied as part of a package or “wave” on some kind of periodic basis after Go-Live. It’s not uncommon for the first change management package to be applied to the system only a few days or weeks after Go-Live. When things settle down, however, these carefully scheduled mass updates should be applied on a monthly or quarterly basis, and sometimes even less often, depending on the amount of downtime you can afford, if indeed downtime is actually required to implement a particular change.

In the end, you need a very rigorous process for screening any “must have” changes, whether functional or technical. The requests to make changes before Go-Live are inevitable, and therefore need to be dealt with. If you find yourself a week before Go-Live, in my experience a resounding “NO” applies most of the time. However, the value of a process for reviewing and validating a critical “STOP THE PRESSES!” change—one that could potentially delay Go-Live—cannot be underestimated.

- ▶ To review change management release strategies and more, see “An Overview of Change Management,” p. 465 in Chapter 13.

Final Administrative Technical Details

Though the system is locked down from a technical perspective, this does not mean that administrative and process details related to *operating* the system cannot be fine-tuned. Nor does it mean that the system is locked down in terms of user-based administration activities, such as creating and modifying roles, reworking SSO (single sign-on) or central user administration via SAP CUA, Workplace, or Enterprise Portal. And refining administrative tasks related to daily operations, enterprise management, and so on are all fair game at this point.

Backup/Restore Processes

The enormous amount of data that can be maintained and generated within your production system is remarkable. Most of my customers support single production databases ranging from 300GB to over a terabyte, for instance. Factor in multiple mySAP components, synchronizing database copies between test, staging, DR, and technical sandbox environments, and the need to at least occasionally back up production file systems other than the core database and logs (like SAP and database executables, the OS partition, and so on), and the raw amount of production data is staggering.

The need to back up all of this data is critical. Further, the need to actually test whether the tape backup is good is key as well. I applaud those few customer sites that truly embrace regular tape *restore* procedures to ensure true recoverability; their due diligence will pay great dividends one day. In addition to testing the integrity of your backups, though, you must also develop and refine the processes associated with backup and restore, such as

- Implementing a grandfather, father, child approach to tape backups, including how often to recycle tapes (for example, every 5 weeks for daily backups, every 5 months for weekly backups, and every 15 months for monthly backups).
- Determining the best “window” of time in which to run full, incremental, or other backups.
- Determining how often to perform an offline (cold) database backup versus an online (warm) backup. Often, offline database backups are performed weekly and online backups are performed daily.
- How often to dump log files to tape, and whether a disk dump should be used for staging.
- How often to ship log files to your DR system, if applicable.
- How often to back up other file systems to tape (many of my clients perform this during their weekly offline backups, so as to have a complete “image” of the system no older than a week at any given time).
- Determine when a DB consistency check might be appropriate, and schedule it through the SAP or database scheduler.
- Finally, assemble a schedule for rotating backup tapes offsite, and bringing older ones back “in” to the data center to be reused again.

Be sure to document all of the procedures and processes that come out of the tasks noted in the preceding list. Documentation details are covered in depth later in this chapter.

Output Management

Just as backup/restore needs to be addressed before Go-Live, so too do the processes associated with determining which users need which output devices, and then creating and maintaining their relevant printers and fax devices. I suggest creating a step-by-step document, complete with screen shots, when it comes to documenting your how-to output management processes. You also want to document *why* you have chosen a particular approach to printing. For example, if you are setting up all users to print through their own distributed NT printers, you want it to be clear to anyone following in your footsteps years later why this approach made sense at the time.

The same should be noted for special faxing approaches, SMTP-based communication applications, and so on.

Tweaking Your Systems Management Approaches

Your systems management applications should have been installed for some time now, and the SAP TSO should be quite familiar with whatever has been implemented. Now is the time, though, to “add” any final servers to your systems management console. The process of adding a server to a management utility is performed in different ways for different applications, but in essence it just means enabling a server to be managed. For instance, it might include

- Hard-coding or identifying a range of names or IP addresses to discover, thereby adding servers to your system console, so that they can then be monitored and managed.
- Updating the system console in other ways, such as updating its version control database (used for managing and sharing data relevant to software and hardware updates), updating the database of known bugs and issues, revising performance or other thresholds, defining key performance indicators (KPIs, discussed in detail later in this chapter), and so on.
- Training or retooling the SAP Operations and Help Desk teams to monitor your production system. This could include any number of systems management tools, such as SAP Solution Manager, HP OpenView, the Microsoft Management Console, SAP CCMS/CEN, and so on.
- Or it may simply mean teaching the team how to use an enterprise systems management checklist to verify that the systems are available and performing within tolerances.

The key this late in the deployment is to *not* change any system-level software. Thus, although you may need to enable an underlying service or capability to support systems management (SNMP and DMI are two common management protocols), now is not the time to make these kinds of changes to the solution stack. Similarly, it is also too late to load new, or *update the version of*, systems management agents on your production servers—save these changes for the first change wave.

More Details—Managing the SAP Enterprise

A number of miscellaneous tasks need to be performed by the SAP Basis or Operations team. For example, it’s usual to forward or collect your SAP administrative alerts in a public folder. The contents of this folder should therefore be periodically reviewed and used to feed reports and other status and service-level needs.

Other tasks must occur, too; a discussion as to which tools each operations staff member should have on his desktop should culminate in installing and configuring these tools, for example. The team then needs to be trained to monitor the production systems when it comes to availability, performance, virus alerts, and more.

Operations will need certain rights (SAPGUI as well as OS-specific) on each SAP server it has been directed to monitor, too. I suggest creating a global group for operations, and providing it with the custom access rights necessary to monitor and manage the system. For specific rights, refer to the tasks outlined in the daily, weekly, and other operations checklists discussed earlier in the book and provided on the Planning CD.

Determining Realistic SLAs

Before you can track and measure your performance or availability, you of course need some kind of metrics. These metrics are often called service levels or key performance indicators (KPIs), and apply to most every team within the SAP TSO. For example, your SAP Operations group will be charged with meeting the following KPIs:

- Production job completion, or the percentage of time that job runs must complete successfully as scheduled. This can range from 99–99.99%; typical is 99.5% in my experience.
- Backup completion, as a percentage of time that backups must complete successfully. Not surprisingly, the minimum acceptable service level is usually 100%.
- Application system administration, which is the length of time between receiving a request (to create a user ID, or reset a password, and so on) and actually completing the request. This might be broken down between “normal” 9–5 business hours and “after hours,” and vary based on the critical nature of the task, too. For example, the target for creating a new user ID might be 90% within 24 hours, whereas a request to change a password might be expected to be done 90% of the time within an hour.
- Monitoring service levels relates to how long it takes to act after an alert has been received. Some of my customers publish KPIs that promise a 15-minute response or escalation process, for example, after an alert is received by their systems management consoles.

Similarly, the SAP Help Desk will be expected to quickly respond to help desk calls and resolve them efficiently; just how quickly or efficiently is described by the KPIs assigned to this team:

- Help Desk Time-to-Answer, which is the amount of elapsed time (such as 30 seconds for 99% of all calls) it takes for the Help Desk customer to reach a human voice.
- Call-back Response Time, or the time that elapses between entering the trouble ticket and calling back the customer to acknowledge the problem and provide some kind of estimate as to the time it will take to get a technician or analyst onsite.
- Help Desk Resolution Time, which is the time to resolve trouble tickets that do not require service dispatch and can further be easily classified as level 1 (basic issue) or level 2 (moderate) severity. In my experience, 85% of all calls should be completely resolved with 30 minutes or so, and 99% within 4 hours.

The SAP Data Center team is tasked with maximizing network and hardware uptimes, and can therefore be tasked with achieving service levels such as the following:

- System landscape production hardware might be required to be available during normally scheduled uptime periods 99.9% of the time. This would not include *planned* outages, therefore.
- Network availability is usually broken down into external and internal service levels. For example, users accessing the system over the Web might expect these network links to be available 99.99%, whereas internal (and therefore more controllable) access might be expected in the neighborhood of 99.999%.
- System requests, or the time required to evaluate system or service requests and respond to them with schedule and cost estimates. These can be fairly long-term in nature, covering many days perhaps.
- Service completion, which is the time between the service request response and actually completing the service call.
- New system installation, which is similar to system requests but is specific to installing a new server rather than responding to a service need on a current system. It's not uncommon to install 95% of all requested new systems with 14 business days, for example.

Of course, hundreds of other service levels need to be defined and recorded for the remaining SAP TSO teams—database administration, SAP basis/infrastructure, change control, project steering committee, programming staff, and so on are all subject to meeting KPIs. Even generic “customer service” and “service evaluation metrics” need to be defined for the entire project.

And although I have focused on measuring key performance indicators inherent to discrete SAP TSO teams, it is also common in the first 60 or 90 days after Go-Live to look at system availability more simply or holistically. For example, one of my favorite customers on the East Coast identified what they called *realistic and conservative* system availability and performance targets. Specifically, after factoring in all layers and components within their solution stack, their goal was to achieve 90% uptime in the first two months, and then improve this to 95% uptime in the third month. After the third month, they would strive for their target three nines of availability. They took a similar approach to measuring performance, too; their goal for the first two months was a modest two- to three-second average response time across all modules, followed in the third month by an average that did not exceed two seconds.

So how do *you* create pertinent KPIs? One sure way is to identify the goals of each SAP TSO team, or to take a closer look at each team's customer. If a team is responsible for performing a certain task, that task is subject to being evaluated in terms of timeliness, quality, and so on. I therefore suggest using this approach as the foundation for creating realistic KPIs by which your solution can be measured. When you understand the tasks, you can determine how easily the data might be collected to prove how well you are performing the tasks. If a task does not lend itself to being easily measured, or requires data that is difficult to obtain, you probably want to pass on using this task as a key performance indicator.

Tracking System Performance

To help ensure that service and performance levels are tracked and met, I suggest *at minimum* dumping your performance and availability data into a database or spreadsheet. Such an approach serves as a data repository and therefore allows for historical reporting. Many of my customers collect this data manually and use it to plot simple graphs that reflect performance statistics and more. A sample "Basic Historical Performance" worksheet has been included on the Planning CD. It's basic, as the name implies, but effective.

Other customers prefer to collect this data automatically, or have no choice but to automate the process due to their complex mySAP system landscapes. I've seen or developed a number of automated approaches that rely on scripted CCMS transactions, similar to the stress testing scripts discussed in Chapter 16. The sample "Detailed Historical Performance" worksheet found on the Planning CD can be easily populated by leveraging similar scripts.

Collecting and analyzing performance data has long been a strength of enterprise management tools like those discussed in Chapter 14, too. HP OpenView, BMC Patrol, and others allow performance data to be easily collected and reported against. This is effective, but given the lack of support by many enterprise applications for all mySAP solutions, may be incomplete in your particular case.

However, the absolute *best* approach that I've seen by far for validating new mySAP implementation KPIs is to use the SAP Solution Manager. With its ability to collect and analyze real-time data across your mySAP enterprise, SSM

- Can help you identify clear and measurable service-level goals
- Can measure and communicate how well you achieve your service goals, based on agreed-upon measurements and key performance indicators
- Finally, can communicate all of this data through easy-to-understand Service Level reports—data is provided for individual servers as well as complete solutions

Creating a Service Level report is simple. As illustrated in Figure 17.1, just open the service level management screen (in the lower-left corner) in SSM, and then start the “Setup SL Report” session in the middle of the Service Level Management pane.

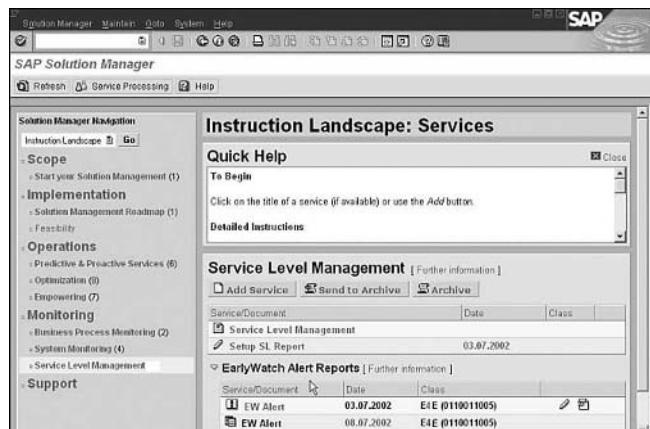


FIGURE 17.1 The SAP Solution Manager elevates KPI validation to the next level, making it easy to create standard service level reports.

Regardless of how you collect and report against your service-level agreements, the key at this stage in your implementation is to ensure that the data being collected is truly useful in proving that SLAs are being met. Thus, *now* is the time to verify that your service-level reporting approaches not only *work*, but *work well*.

The Changing Role of the SAP TSO

As the SAP system evolves into the butterfly (or “moth,” if you’re going with the classic gray SAPGUI!) it is destined to become, so too will the technical support organization tasked with the system’s implementation and upkeep. The very nature of

the jobs held by many SAP TSO members will change in a flash, after Go-Live comes and goes. And just like that, other positions will completely disappear, to be supplanted by new or morphed roles and responsibilities.

Updating Responsibilities and Roles

A few weeks before Go-Live marks a very interesting time in the state of your SAP system landscape. For the first time ever, it is truly static—it is locked down. Nothing at a technical level is changing. Even the functional teams are wrapping up their activities, and although their hours are long, and the transports into production never seem to stop, the nature of the activities taking place at this level are changing, too.

The big day of Go-Live marks the beginning of a new phase in the life of your mySAP solution focused on continuous improvement, a phase that some of my customers call “maturity.” I like the term maturity, as it denotes the change in mindset previously discussed. By placing a system into production, you turn the corner from implementation to operations and continuous improvement. This new phase in the mySAP component implementation life cycle is the longest by far, and requires a support organization that plans to stick around for the long haul. But as I discussed in Chapter 8, the long haul equates to “steady state,” and this in turn equates to perhaps a different type of personality requirement, one focused on achieving SAP *operational* excellence. Along with this operational mindset must come the desire to manage the system, too, rather than allowing changes to control you.

New Roles, New Personality Types

Earlier in the book we discussed two different personality types, one characterized by a need to climb steep learning curves, and one characterized by the need to achieve operational excellence. I labeled these respectively NPs and SMs, or New Project types and Systems/Maintenance types. Everyone tends to exhibit a little bit of both, of course, but with the changing nature of your SAP solution, you’ll want to seek out or develop the latter characteristic in the people charged with *maintaining* the system.

- ▶ To understand how staffing with new project and support/maintenance types impacts the success of your project, see “Understanding the Two Key SAP Support Staff Personalities,” p. 298 in Chapter 8.

Besides, with the system beginning its new productive role in servicing your internal and external customers, it’s inevitable that some of the NP folks assigned to the project will ultimately find less and less of a learning curve to conquer, and will leave. Conversely, the true SM people on your project will completely enjoy the next few years, as things settle down and they play a role in ensuring that the system delivers the performance and availability that your SAP technology partners promised.

My point here is two-fold. First, as attrition takes its toll on your project, be sure to find the right people for backfilling the various roles left vacant (if indeed the role needs to be backfilled). Second, proactively seek to *relocate* your new project types before they leave; the current project may be winding down, but it's my experience that other mySAP component, technology, or third-party integration projects are often on the horizon. So do your best to retain the people that you and your company have invested so much time in—the payoff can be huge when these same people are leveraged in sister projects.

Preparing SAP Operations and the Help Desk

At this point, the operations team should already be experienced with supporting SAP. Over the last few months, they've monitored the system from a performance and availability perspective, assisted pre-production users (like developers and testers) in resolving issues, and so on. The help desk should have been established and staffed by now, and should be currently focused on learning how to report, escalate, and resolve issues. In rare cases, however, I've seen SAP customers hold off on putting together the help desk until a few weeks before Go-Live. It's by no means a best practice, but with everything else going on, it happens. In these cases, it is most often the operations team (or a subset of this team) that is expected to get up to speed quickly. In other cases, I've seen the help desk function outsourced. Regardless, this team needs to be prepared for some pretty unusual things, and armed with troubleshooting tools, escalation processes, and above all a customer-service attitude.

In my experience, curious, customer-driven folks comfortable in the role of "jack of all trades" tend to be most successful. This is because of the issues with which they will be faced. For example, I saw one case where a help desk analyst and the basis team were troubleshooting a recurring performance issue. During the lunch hour every day over the course of a few weeks, the system's response time was rising significantly, consistent with an ever-growing workload. It got to the point where things just didn't add up, however—most of the users would be out of the office enjoying lunch, but the system was nonetheless taking a beating. An enterprising help desk analyst decided to quit focusing on the numbers displayed in CCMS and his performance reports, and instead walked over to the building housing these users. What he found was pretty amazing—lying on top of many users' keyboards were staplers, tape dispensers, and in one case a large claw hammer (how many people have *hammers* in their desk?). As it turned out, one of the users had discovered that his session would not time out if the keyboard's Enter key remained depressed. To this day, I have never quite understood their reluctance to log back in to the system after lunch. The huge transaction load they generated explained everything, though, and this help desk analyst rather than his more experienced basis colleagues solved the problem (and had an interesting story to share for many years).

Final Preparations

Enterprising operations and help desk teams will focus their efforts a few weeks before Go-Live on ironing out things like the following:

- Expectations when it comes to performing their job. To this end, a “general overview” document of what each shift looks like in terms of tasks and responsibilities is useful.
- Shift turnover procedures, including when to escalate issues that may have cropped up on a previous shift, ownership of issues, who is responsible for performing various tasks, and so on.
- Gathering and publishing contact information for the entire SAP TSO.
- End-of-shift daily, weekly, monthly, or other recurring special procedures (for example, how to execute and manage Friday night weekly backups that cross over two different shifts).
- The role they play in terms of protecting the SAP resources from viruses.
- To what extent they will leverage an operations checklist.
- How the team will manage changes to documented procedures; where they will reside, who will update them, and how these changes will be validated.
- How to provide new SAP operations and help desk members access to their resources; access to tools, distribution lists, collaboration sites, and so on needs to be granted, and training needs to be made available as well.

The team may also be tasked with managing both the old and new systems, too, if the decision was made to run parallel systems (a decision that in some respects mitigates the risk of Go-Live but complicates the business while creating a lot of additional work for many). Preparation must therefore take into account the time that will be required to monitor and manage both production systems during this time period, especially with regard to staffing. Finally, central to many of these monitoring tasks is the importance of maintaining “as-is” and process documentation, discussed next.

Updating “As-Is” Documentation

As the technical state of your SAP environment changes and impacts operational activities, these changes need to be tracked. I have found recording the following data quite useful not only in getting new SAP implementations started off on the right foot, but also in tracking changes throughout the implementation:

- All hardware-centric data, such as server and disk subsystem models, serial numbers, drive layouts, PCI slot details, and so on.

- Logical diagram of your unique mySAP environment, illustrating the various system landscapes, server topology, functionality hosted by each system, level of business criticality, and so on.
- Logical diagram illustrating the network topology that interconnects the servers, and provides access to front-end clients, back-end resources, the Internet or company intranet/extranet, and any other important details.
- Physical diagram (sometimes called a “floor map” or data center layout) showing where all of the SAP hardware resides in the data center.
- Physical diagram that details the contents of each rack.

With the current “rolling” state of the system documented, the operations and help desk teams can then turn their attention to process documentation.

Updating Procedural Documentation

Unlike point-in-time documentation, procedural documentation seeks to walk its reader through how to perform a task or activity. Thus, I often call these “how-to” documents. A how-to document needs to be referenced whenever possible, and created when necessary. Fortunately, many how-to’s exist in the form of SAP white papers, SAP Notes, installation and master guides, partner-provided documents, and so on. In the past, though, I have had to create specific process documents for the following:

- Server hardware and OS-related documents, such as “How to Upgrade Server Firmware,” “Installing a Server through Scripting,” “How to Set Up an OS Software Stack for SAP,” and “Installing MSCS for Technical Sandbox Instances”
- SAN hardware How-To’s, such as “How to Create SAN Disk Volumes” and “How to Map SAN Volumes to Logical Drives”
- Database How-To’s, including “How to Install Microsoft SQL Server 2000 and Service Packs for SAP”
- How to implement specific mySAP components, and how to install options such as CUA for Workplace, or the Requisite Catalog for Enterprise Buyer Professional installations, or “How to Download BW Web Queries to MS Excel”
- How to perform specific post-installation procedures, such as setting up a printer or the Transport Management System, installing and uninstalling the SAPGUI, configuring operation modes, executing a client copy, applying support packages, using SMLG to set up logon groups, and so on—many of these processes are documented by SAP AG, but my versions include screen shots and other details to leave little to chance. I’ve included some of these procedures on the Planning CD.

- Additional bolt-ons, such as “How to Install the SAP Library” and “How to Install the Microsoft Exchange Connector for SAP”
- How to use various utilities and applications to monitor performance and availability, such as CIM, PerfMon, the SAP MMC, and more. Some of these documents can be found on the Planning CD.
- How to perform various tape backups, database backups, and more
- Problem analysis processes, such as “How to Identify and Analyze Connectivity Issues,” “How to Identify and Analyze SAP Performance Issues,” and “How to Identify and Analyze SAP Printing Issues”

Updating your process documentation is critical because these processes can be complex and therefore subject to error, and because these processes are typically not static; they change depending on the modifications made to the solution stack. Thus, I recommend that you never postpone updating your documentation. In my experience, when documentation is postponed, it rarely ever gets done—other projects and “emergencies” invariably continue to crop up, pushing the documentation out further and further along your timeline. Thus, by the time you find yourself a spare moment, you might very well have forgotten the processes to be documented.

- ▶ To look over best practices and other approaches useful in creating practical documentation, see “Documentation Best Practices in the Real World,” p. 509 in Chapter 14.

Addressing Future Service and Support

One day, your system will fail. It’s inevitable; even five nines of availability still equates to a small amount of unplanned downtime. And even if your unplanned downtime numbers are small, you will still require *planned* downtime to implement changes to your business processes, technology infrastructure, or any number of elements or components inherent to your solution stack. It is for these reasons that, well before you need it, you need to arrange for mission-critical support services for your SAP environment. As you will see in the next few pages, this support umbrella hails from deep within your SAP TSO as well as from traditional external sources.

At a minimum, you will need to collect contact and escalation information for each technical element or component in your solution stack—from top to bottom. I recommend obtaining not only the usual “1-800” vendor support information, but the deep technical links as well, such as contacts at each partner’s SAP Competency Center or SAP second-level and third-level engineering groups. Add to this *local* vendor contact information, such as pre-sales and post-sales team members, field service, account management, and any special consulting organizations with SAP

experience that are within a few hours of your data center. Finally, augment this external technical contact data with your internal organization charts and on-call lists, and you'll be well prepared when unexpected issues crop up in your system. A sample "Service and Support Contact List" has been provided on the Planning CD.

As I alluded to before, this information is typically maintained by the SAP operations or help desk organizations. My favorite way to maintain all of this service and support contact information is through an easily accessible "Quick Reference Support Matrix" document or Web page kept on your company intranet (along with the rest of your as-is, process, and other documentation). Many of my customers keep their lists of on-call company-internal resources separate from external contact information due to privacy issues—it's not uncommon to include home and personal cell numbers in your rotating on-call lists, for example.

Support Agreements

Each mySAP customer needs to draft a support agreement between it and SAP AG, its hardware partners, database partners, and any other SAP technology vendors. Normally, service is tiered at different levels; more comprehensive service levels, or faster service response times, equate to higher monthly or annual support fees. A support agreement will also address how the various companies represented in the agreement will collaborate with other partners to provide technical and business process support to their mutual customers. Things like processes for problem escalation, expectations during problem resolution, and so on need to be documented and understood between all parties.

Beyond standard service and support agreements, many customers also choose to execute "Consulting on Demand" or other reactive services agreements. Most of the time, these are SAP-specific in nature, focused on basis support, functional support, or integration troubleshooting services. I've seen these executed for particular system components though, like access to deep SAN expertise or e-commerce skills. The key here is that the customer has the paperwork done to quickly access a specific resource, so that in the event of a critical issue, no time is wasted putting a service agreement in place.

Leveraging Joint Escalation Centers

Most everyone is familiar with working through basic technology issues with vendors and partners—typically, a partner's first-level support center is contacted, and a trouble ticket or "case" is created to help document the information and process used to solve the problem. For single-vendor solutions where the support center is well-versed in a solution, a resolution to an issue can be obtained pretty fast. Unfortunately, it's impossible to implement a monolithic solution stack for SAP; various hardware platforms, disk and database subsystems, operating systems, and of course mySAP components equate to multiple partners, each with a first-level

support center. Thus, it might seem that the most typical support model employed by a customer would require no less than three or four trouble tickets, one with each partner. All of these partners would be provided the same redundant data (the nature of the issue, how it manifests itself, details relevant to the issue in terms of technology or business process, and so on), subject to the individual interpretation and biases of each partner. In doing so, coordination between the different partners and SAP AG would be cumbersome at best, and finger-pointing would be an inevitable by-product of such an inadequate support model.

What mySAP customers really need is a support model that melds a hardware partner's expertise in IT management and infrastructure with application and business process expertise inherent to SAP AG. This is where *Joint Escalation Centers*, *Global Solutions Centers*, or *Joint Solution Services* come into play. For our purposes here, I'll refer to these simply as the JEC. Regardless of the name, technology partners who are serious about supporting their SAP customers will participate in, or create, an organization or service offering that

- Takes ownership of escalated first-level issues
- Addresses the needs of both technology and business process issues
- Is capable of reproducing integration and interoperability issues in a lab environment
- Can access, stage, and test the specific solution stack gear deployed in one of their customer's SAP environments
- Has deep ties into both SAP and their own back-end product engineering organizations
- Performs their work in a manner that promotes communication between the customer and the JEC, as illustrated in Figure 17.2, and ultimately facilitates rapid problem resolution

All of this troubleshooting and testing is performed through a joint-staffing model, where the engineers and consultants hail from both SAP and its technology partners (HP, IBM, Dell, Fujitsu, and so on). SAP's PartnerPort in Walldorf meets this criteria, and houses many a "global" SAP competence or support center. Other implementations of the JEC model exist as well. For example, most of the major ISVs, such as Microsoft, Oracle, HP, and IBM maintain support organizations that are tied to various hardware organizations and SAP AG resources. And some of the hardware partners have assembled JEC resources or service offerings in various key geographies as well.

With these kinds of capabilities, a day in the life of a JEC revolves around diagnosing and resolving interoperability issues, addressing performance issues, modeling business process failures, advising and directing the solution-sizing efforts of their

respective SAP Competence Centers, and more. The JEC is uniquely positioned to not only work with back-end engineering resources to help them understand how issues manifest themselves, but also to implement the fixes in a lab environment to *prove* it. And the JEC further proves their work in testing regularly published ISV-provided or hardware-specific fixes, patches, updates, service packs, and so on. As a customer of SAP, you want to be sure to understand and take advantage of the PartnerPort and other JEC-like organizations tasked with supporting mySAP solutions.

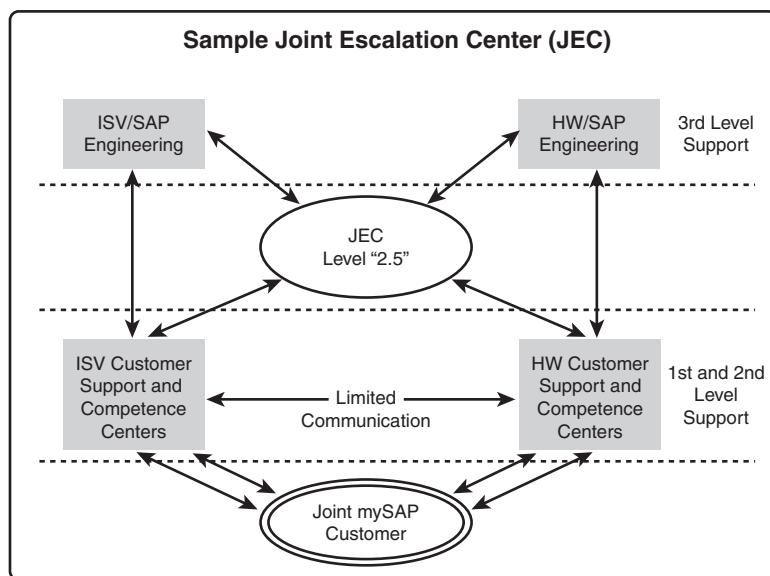


FIGURE 17.2 A Joint Escalation Center represents a key support component, facilitating communication and problem resolution between SAP AG, itself, and the customer.

Thus, if we review the goals of the previously discussed Support Agreements, clearly the JEC represents where the rubber meets the road. It's the JEC that addresses filling in the support holes left by multiple first-level support organizations or competence centers staffed in a one-sided manner. Even a modest investment of two or three engineers can pay off in a big way in terms of improved customer satisfaction and therefore longer customer relationships. As most of us are aware, it takes a lot more effort to win a new customer than it does to maintain an existing one; the JEC serves this purpose, and creates a best-in-class support environment that will attract *new* customers as well.

The First Week of Go-Live

When the system is finally turned on for use by its end users, you have achieved Go-Live. This is very different than technical Go-Live, when the system was locked down and no changes were allowed from a technical perspective. Go-Live is actually synonymous with “functional” Go-Live. However, unlike its technical counterpart, it’s unlikely that the system will indeed be locked down for long after the big day of Go-Live comes and goes. Big changes will of course require downtime, and will therefore be relegated to regularly scheduled change waves. But smaller changes and tweaks to the system will continue to be made and promoted to production as usual, leveraging the transport system.

Turning the system “on” is by no means the end of the road, though—we’ve covered a few of the activities that must occur post-Go-Live throughout this book. In the concluding sections of this chapter, I will take a closer look at this critical week, including how to

- Obtain feedback on the solution, to promote the idea of continuous business process and technical improvement
- Monitor the system during the first critical week of Go-Live
- Evaluate how well the SAP TSO performed during the implementation
- Celebrate!

Go-Live should have been a “non-event.” That is, well-planned and executed mySAP implementations merely make a new productive system available to end users, who in turn are well prepared to begin using it. The transition from the old way of doing things to the new solution should therefore be seamless. I’ve been fortunate to play a part in countless SAP Go-Live weeks, and I want to share how the various teams accomplished this greatest of goals—an uneventful Go-Live—next.

Monitoring During Go-Live Week in the Real World

The foundation of a smooth Go-Live rests on monitoring critical performance, availability, and other system characteristics, and rapidly communicating this information throughout the team. To this end, I have seen all of the following approaches prove useful:

- During the first week of Go-Live, twice-daily status meetings between the technical, functional, operations, and help desk teams keep everyone on the same page.
- Developing an anticipated list of “Most Frequently Asked Questions,” updating it daily through help-desk feedback, and publishing this list to the end-user community in an easily accessible format (for example, a Web site) is key. Basic

topics like explaining how to access the system, login, how to reset passwords, and so on are more common than you think and save everyone time.

- Ramping up the help desk to handle more calls, as discussed previously, is also key. Further, communicating the top-10 questions and resolutions throughout the team (to update the FAQs) and escalating critical issues are important as well. Both of these will serve to reduce help-desk call volume.
- Monitoring SAP CCMS throughout the day (every hour or two the first few days, stretching out to snapshots a few times a shift by the end of the week) is the most important thing the SAP Basis/Infrastructure team can do. The idea is to proactively identify impending performance issues. I suggest regularly running through SM66, ST06, ST04, ST03, and ST02 at minimum this week. Add ST11 (to display error logs) as necessary, and use the Daily Operations checklist found on the Planning CD to monitor additional parameters and conditions, too.
- The database administrator(s) need to monitor how well the RDBMS and disk subsystem are performing, from a hardware, operating system, database, and SAP CCMS perspective.
- The Disaster Recovery team must examine how well the DR solution appears to work. That is, they need to verify that log shipping works as expected, or that the cluster is operational—whatever DR solution was implemented. There should be no surprises because all of this should have been performed previously, but it's still key to perform the due diligence during Go-Live week.
- The network infrastructure/data center team needs to monitor the performance of all internal and external links binding the mySAP system landscape together, and to other systems.
- The data center or operations team also needs to monitor events generated by the system and captured by the systems management applications and tools. In addition, special attention must be paid to basic infrastructure power and cooling during this first week, to catch any trends in utilization of these key assets before they become a problem (this first week will find *all* of your SAP hardware and supporting infrastructure up and running, which may not have been typical in the past).
- The functional or programming teams need to remain available to discuss business process issues with the help desk (and end users, perhaps). This also includes monitoring the number of ABAP dumps created (ST22), to determine where rapid fixes can be implemented.
- Reviewing the solution stack, to verify that no changes have been made unbeknownst to the team, is still a good idea at the end of day one and again at the end of the week. In other words, it's important to verify that no changes are being made behind the scenes, circumventing change control.

While the core technical and functional constituents of the SAP TSO are busy collecting, monitoring, and analyzing the performance of the system, the project management team needs to address related tasks, as communicated next.

Planning for Feedback and Continuous Improvement

Though I use the term lightly, a number of knowledge “databases” exist at this point. For example, the Knowledge Repository should contain all project definition and success criteria, TCO analyses, sizing data, staffing plans, testing scripts and results, project deliverables, and so on. Partners will also maintain system-level or other data relevant to your project. Additionally, the master project plan and all of its iterations (as it was revised over the course of the project) will reflect changing needs and the impact of scope creep. Enterprise systems management tools and utilities will contain a wealth of data by the end of the first week after Go-Live, and this data will continue to grow in terms of both volume and its ability to chart historical trends. The change control system will house similar data, though from a functional perspective as well as a technical one. Finally, the cases created using the help desk’s tracking tool will serve as a database of issues, resolutions, and trends. All of these “databases” can be leveraged to provide project feedback to the team, and help to focus the efforts of the entire SAP TSO in the weeks to come.

Much of this documented “feedback” will highlight what the team did right, and what could be improved upon. It will underscore what cannot be changed, what should change, what worked, and what did not. And performance data collected during the first week of Go-Live will serve as a baseline for the system, a starting point against which updates to the system will be measured. As time passes and the players in the SAP TSO change, this initial collection of information will help level-set newcomers, too.

But perhaps more importantly, this information will serve as the foundation for continuous improvement in the system. Like a baseline in a stress test, all of the data that describes the system’s current state will be used to measure the impact of future state changes. Thus, everything from the technical infrastructure to the actual business processes that execute through the system will play a role in continually improving the system across the board.

Simply *having* all of this information is only the beginning, though. To put continuous improvement into action, the team needs to embrace action as well:

- The SAP and client project managers need to perform a final review of the Master SAP Implementation Project Plan, to ensure that all milestones have indeed been completed, and to communicate project management lessons learned with the steering committee and senior project staff.
- A final high-level “Project Lessons Learned” document, consisting of individual feedback documents created by each SAP TSO team, needs to be assembled and

distributed in *electronic format*. Because this approach ties all organizations together, take care to request only a page or two of lessons learned from each group or team leader—this will allow you to create a document of manageable size and phenomenal value, especially useful if you are preparing to launch similar projects or add new mySAP components to your enterprise environment.

- The core SAP TSO team will need to review the top 10 or 20 help desk calls every week, to update or fine-tune MFAQs, automate more assistance, and ultimately reduce call volumes by working through these issues.
- Similarly, the team responsible for systems management needs to review their toolsets regularly. It's often possible to automate responses to certain business and technical events, or automatically escalate issues to the appropriate teams. In the same way, you will find issues that pop up as events that really don't qualify and can be summarily noted or even ignored; the better you fine-tune your enterprise management application, the fewer near-worthless events you will be forced to manage.
- The SAP Basis/Infrastructure team needs to push many of the standard performance monitoring responsibilities over to the systems management team, by helping them to identify typical performance thresholds and to characterize trends. They also need to push standard systems administration to the operations team, and User ID maintenance/SSO responsibilities to the help desk. In doing so, the basis team will not only be developing "backups," but they will also free themselves to work more closely with the functional, development, and change management teams going forward.
- Finally, the functional and programming teams need to focus on developing, testing, and implementing business-driven user-motivated changes. A testing process that encompasses built-in functional and stress-test milestones will help promote continuous improvement in the code that they author. Regularly reviewing system dumps, revising work flows as directed, and in general embracing change management will consume the rest of their time, while promoting continuous improvement as well.

And though not precisely a continuous-improvement *process per se*, a high-level appraisal of the team's performance needs to be communicated verbally via the final team meeting of Go-Live week. I have seen these meetings used to underscore the important roles held by key team players, and to communicate how well the team worked together to achieve the project's objectives. Normally, one and a half to two hours is about right, though the best of these are wrapped around something like a long catered lunch of prime rib and lobster. Don't misunderstand the importance of this verbal appraisal (or *Go-Live Lunch*, if applicable), though; the feedback from the project leadership team to the team itself can be huge in terms of improved morale and ultimately future performance.

Post-Implementation Evaluation

Although continuous improvement will play a role in ensuring that the system continues to better meet the needs of its end users, a one-time *Post-Implementation Evaluation* is commonly used to obtain formal concise feedback shortly after Go-Live. It is normally the product of the client SAP project manager, though many organizations and individuals play a role in shaping it. After this evaluation form has been crafted, it is shared with the following organizations:

- The manager of each key business unit or other customer organization. If different from the manager, the lead person within each business unit responsible for working with the SAP TSO also needs to complete the evaluation.
- Each manager or team leader within the SAP TSO tasked with any level of responsibility for the project. For small to medium implementations, I've seen every member of the IT group that supported an SAP deployment given these; because feedback is the goal, that's not a bad idea.
- Key implementation and technology partners, especially core project managers and senior technologists.

After the results are collected and tabulated, they should be used to provide feedback to the entire team. And I've found these useful when it comes time to writing employee reviews or to justify that a consultant's contract be extended, too.

Proper Congratulations

If you remember back to Chapter 8, I recommended a number of compensation alternatives in addition to pay increases, promotions, cash and other bonuses, training opportunities, and so on. Proper congratulations for Go-Live should certainly include one or more of these, too, depending on an individual's role and success in the project. But I still maintain that the most important thing, the thing that will help you to retain your top talent and key steady-state personnel, is judicious and honest use of the "attaboy" or "attagirl."

- To read more about how to provide positive feedback to your SAP technical support organization, see "The 'Most Important Thing,'" p. 302 in Chapter 8.

Especially during the week of Go-Live, or shortly thereafter, the project management team, and company management team in general, have a unique opportunity to make a lasting impact on its key employees and long-term contractors. I use the term "key" loosely, too, as I believe that a mySAP implementation or upgrade is only successful when a team of people prove that they work together exceedingly well. Thus, I look at attaboy and general congratulations in terms of layers (thankfully, not solution stack layers, of which I'm sure you're thoroughly tired by now!). Rather, I like to see a performance bonus doled out to the *entire team*, no exceptions. In the

past, I have seen these range from fixed cash bonuses of \$5K to \$10K, to percentage bonuses of 5, 10, and even 25%.

On top of this, though, I feel that there is great value in activities like the Go-Live lunch I discussed earlier, or getting together for a formal banquet or other event after hours to dole out more targeted awards and sincere words of thanks. It is at these celebrations that public attaboy carry the most weight as well. These are especially heady events when the budget can absorb the cost of hosting both its SAP project team members and their spouses or significant others; indeed, the attaboy become that much more meaningful, too.

As I said before, on top of the meal I like the idea of handing out awards. Some of these should be light and humorous, to keep the event a bit more celebratory. Others should be targeted at the top performers, though, typically the various project managers, team leaders, senior solution architects, and other senior technical staff. Cash bonuses are always nice, but gift cards and the like are appreciated as well.

Though not nearly as frequent as they should be, I've also seen gifts handed out to family members; gift cards to nice restaurants, enclosed within handwritten cards personally thanking them for allowing the project to "steal away" their spouse during the project, are excellent. Remember the family, I say, and you'll go a long way in building long-standing bridges to your valuable employees and other team members.

Tools and Techniques

This closing chapter includes a number of spreadsheets aimed at collecting and helping you to analyze system performance. I've also included a number of checklists related to Go-Live, an executive PowerPoint presentation useful in preparing and discussing the business cutover, three different spreadsheet-based approaches to tracking SLAs/availability/performance, and a slew of one-time and administrative/recurring how-to documents that will prepare you for what I hope proves to be an uneventful Go-Live week. And the figures found in this chapter are also included in either PCX or Microsoft PowerPoint format.

Summary

Here we are, more than 700 pages down the road, and you've read about what it takes to plan for, execute, and bring live an SAP system. In some cases, you may have actually implemented or upgraded to a new mySAP solution, or perhaps integrated a new mySAP.com component into your existing enterprise. Regardless of the scope, congratulations are in order! I genuinely hope that I was able to help you avoid some of the land mines that my colleagues and I have stumbled over in the past.

In this final chapter I walked you through the process known as cutover. We learned that cutover is much more than Go-Live, or the instant in time when your new mySAP solution is finally turned over to its anxious end users; instead, it refers to the entire transitional process, from planning for the big day to administration and operations, to running parallel systems, to synchronizing or executing last-minute data loads, and much more. I also addressed the final administrative and operational details necessary to prepare you for the big day, and discussed how the SAP Technical Support Organization will need to change. A quick discussion of service and support agreements followed by a detailed description of how to monitor the new system, address continuous improvement, and reward your SAP TSO team during the first critical week of Go-Live wrapped up the chapter.

Index

Symbols

- 2-tier architectures, 256
- 3-system landscapes, 14
- 3-tier architectures, 230-231, 255
- 9iRAC, 198-199

A

- Academies (SAP), 321
- Accelerated SAP (ASAP), 44, 381-383
- accessibility
 - documentation, 510
 - system landscapes, 79
- acquisition costs, 124
- active users, 227
- administrative scripts, 599-600
- AGate, 403
- analysts, Change Management Analysts, 496-497
- APO (Advanced Planner and Optimizer), 10, 204
- Application Consultant certification, 337
- application layer, 145
 - distributed architecture, 146-147
 - heterogeneous system landscapes, 147-148

SPOFs (single points of failure), 206
stacking, 147

Application Servers, 230
InQMy Application Server, 257
Web AS, 251

Application Service Providers (ASPs), 77, 89

applications. *See* software

AppManager for R/3, 527-528

architected solutions, 265-266

archiving, 420-422

“as-is” documentation, preparing for Go-Live, 638-639

ASAP (Accelerated SAP), 44, 381-383

ASPs (Application Service Providers), 77, 89

“attaboy” (positive feedback), 302-303

attendance certificates, 325

authorizations, 416-417

AutoController, 584, 597-599

AutoIT, 515

automated business process testing, 551-553
BC-eCATT interface, 559
business process constraints, 558-559
eCATT differentiators, 554-555
number of test runs, 559
SAP system landscape requirements, 556-557
staffing issues, 557-558
third-party tools, 553-554

automated data collection (CCMS), 513-515

AutoTester tools, 553
AutoController, 584, 597-599
ONE, 514-515
Web site, 597

availability. *See HA (high availability)*

B

background noise, 608

backups
B/R (backup/restore) solutions, 419-420
backup generators, 345
backup processes, 616-617
Go-Live, 629-630
TCO analysis, 149

Bart’s Boot Disk Web site, 385

baselines (performance), 572

batch jobs
cross-application batch scheduler, 522
housekeeping jobs, 626-627
stress testing, 581

BC-eCATT interface, 559

benchmarks, 586-587

best practices, 7

BI (Business Intelligence)
installation, 406-407
sizing, 255-257

“bigger than me” staffing approach, 437-438

BMC Patrol, 524-525

bonuses, 303

bootable floppy disks, 385

BPR (business process re-engineering), 29-30

B/R (backup/restore) solutions, 419-420

Breakout Software MonitorIT, 531

bstat tool, 532

budgets, 219, 232
initial implementation budget, 46-47
refining, 273

SIPP (SAP Infrastructure Implementation Budget), 84

TCO (total cost of ownership) analysis, 119-120, 123-124
 acquisition costs, 124
 backup/restore capabilities, 149
 defined, 124-126
 delta analysis, 126
 disk subsystems, 140-141
 distributed architecture, 146-147
 documenting, 158-159
 DR (disaster recovery), 130-133
 HA (high availability), 128-130
 heterogeneous system landscapes, 147-148
 leasing versus purchasing, 153
 maintenance costs, 152-153
 manageability, 136
 management tools, 149
 operating systems, 141-142
 operations, 136, 153-154
 outsourcing, 126
 people costs, 125, 151-152
 performance delta analysis, 133-134
 Planning CD resources, 159
 recurring costs, 125
 relational databases, 142-145
 risk factors, 154-158
 SAPGUI, 150
 scalability, 134-135
 security, 136
 servers, 138-140
 stacking, 147
 standardization, 137-138
 Systems Management, 153-154
 TCO-driven system landscapes, 76-77
 technology costs, 125
 upgrades, 148-149

Bull SAP Competency Center, 223

Business Intelligence. *See BI*

business processes

BPR (business process re-engineering), 29-30
 documentation, 509
 scripts, 585, 591-592
 administrative scripts, 599-600
 client infrastructure, 596-599
 core script development, 592-596
 scripting tips, 606-609
 utility scripts, 599-600
SPOFs (single points of failure), 208-209
testing, 543-544
 BC-eCATT interface, 559
 CATT, 544, 546
 cautions, 564
 data tracking, 562
 eCATT, 544-546
 functional testing, 546, 549-551
 integration testing, 546, 550
 negative testing, 558
 number of test runs, 559
 per test scenarios, 547
 planning and preparation, 551-553
 Planning CD resources, 564
 post-execution tasks, 562-563
 process constraints, 558-559
 real-world example, 563
 regression testing, 546, 550-551
 “saptp - Compuware TestPartner and SAP eCATT” (white paper), 564-565
 SLA support, 563-564
 staffing issues, 557-558
 system landscape requirements, 556-557
 test organizers, 544

- Test Workbench, 561-562
third-party tools, 553-554
when to use, 547-548
- Business Sandbox**, 52, 72
- business warehouse**. *See BW*
- business unit buy-in**, 49-50
- buy-in**, 49-50
- BW (business warehouse)**
implementation, 37-38
SPOFs (single points of failure), 204
stress testing, 588-589
- C**
- CA Unicenter Application Management for R/3**, 525-526
- cables**
color-coding, 342
Data Center facilities, 354-355
fibre cables, 372
power cables, 345
- Campus Clusters**, 186-188
- Candidate Status Matrix (CSM)**, 282-284
- capacity planning**, 237, 263
- career pathing**, 304
- case studies**
business process testing, 563
BW (Business Information Warehouse) implementation, 37-38
Change Management, 497-500
consultants, 118-119
DR (disaster recovery), 214-217
HA (high availability), 214-217
interviews, 291-293
stress testing, 617-619
- TSOs (Technical Support Organizations), 103-104
all under one roof, 106-107
DBA cubbyhole, 105-106
Fortune 500 global SAP TSO, 459-462
hardware versus SAP groups, 104-105
leveraging computer operations, 107-108
medium business TSO, 458
multiple production instances, 108-109
small business TSO, 456-457
- CATT (Computer-Aided Test Tool)**, 544-546
- CCMon (Cluster Consistency Monitor)**, 538-539
- CCMS (Computer Center Management System)**
automated data collection, 513-515
CEN (Central Monitoring System), 515-517
manual processes/checklists, 512-513
transaction monitors, 515-517
transactions, 611
- CD**. *See Planning CD resources*
- CEN (Central Monitoring System)**, 515-517
- Central Instance (CI)**, 14, 231
- Central Monitoring System (CEN)**, 515-517
- central systems**, 360-361
- certification programs**, 321, 336-337
- Certified SAP User program**, 337
- CFS (cluster file system)**, 193-194
- Chair of Steering Committee**, 47
- Change Management**, 465-467
change controls, 33-36, 56
Change Management Analysts, 496-497
Change Management Managers, 495-496
communications, 477-480
database layers, 488-491
documentation, 472-473
feedback, 484-485
hardware, 488-491

- operating systems, 488-491
- planning, 493-494
- Planning CD resources, 502
- practices to avoid, 500-501
- preparing for, 628-629
- real-world examples, 497-500
- release strategy, 474-476
- Review Boards, 495
- SAP implementation phases, 485-487
- stakeholders, 468-469
- standards, 473-474
- summary of goals, 468
- Technical Sandbox Change Management Checklist, 485
- testing, 471-472
- tool sets, 482-484
- transport strategies, 491-492
- upgrades, 492-493
- workflow, 481-482
- characteristic combinations, 260**
- characterization testing, 234**
- CheckIt Utilities, 531, 576**
- checklists.** *See also documentation*
 - custom checklists, 391-392
 - quick-reference checklists, 508
- chkdsk command, 577**
- choosing SAP Solution Stack Partners, 264-268**
 - production references, 266
 - risk level, 266
 - sizing data, 265
 - specialized vendors, 268
 - support for architected solutions, 265-266
 - TCO analyses, 266-267
- CI (Central Instance), 14, 231**
- classes (training)**
 - Consultant Education, 321-322
 - custom training curriculum, 323-325
 - onsite training workshops, 323
 - Project Team Training, 320-321
 - SAP Academies, 321
 - training/attendance certificates, 325
 - User Education, 322
- clients**
 - client strategy, 415-416
 - defined, 14
 - stress testing, 596-599
 - three-tiered client/server architecture, 230-231
- cloning, 182-184**
- close followers, 64**
- Cluster Administrator, 538-539**
- Cluster Consistency Monitor (CCMon), 538-539**
- clusters, 179-180, 353**
 - 9iRAC, 198-199
 - Campus Clusters, 186-188
 - CCMon (Cluster Consistency Monitor), 538-539
 - CFS (cluster file system), 193-194
 - Cluster Administrator, 538-539
 - Continental clusters, 188-189
 - Linux clusters, 194
 - MC/ServiceGuard, 192-193
 - Metropolitan Clusters, 186, 188
 - MSCS (Microsoft Cluster Server), 191-192
 - Oracle 8i clustering, 197-198
 - SQL Server clustering, 197
 - Tru64 cluster file system, 193-194
 - UNIX cluster approaches, 192-193
- Cockpit (mySAP SEM), 413**

- collecting statistics, 603-605**
- color-coding cables, 342**
- commands**
 - chkdsk, 577
 - format, 577
 - ping, 403
- communication, 300-301**
 - Change Management, 477-480
 - Communication Liaisons, 209
 - communication stations, 257
- Comparative Analysis, 428**
- compensation alternatives, 304-307**
- Competency Centers, 223-224**
- Component Basis specialists, 438-440**
- Component Functional specialists, 311, 440**
- component installation (mySAP), 10, 14, 404-405**
 - authorizations, 416-417
 - liveCache installations, 412-413
 - mySAP SRM (Supplier Relationship Management), 414
 - mySAP BI (Business Intelligence), 406-407
 - mySAP CRM (Customer Relationship Management), 407-408
 - mySAP PLM (Product Lifecycle Management), 408-410
 - mySAP SCM (Supply Chain Management), 410-413
 - mySAP SEM (Strategic Enterprise Management), 413-414
 - mySAP SRM (Supplier Relationship Management), 414
- Planning CD resources, 422-423
- post-installation tasks
 - archiving, 420-422
 - backup/restore solutions, 419-420
 - client strategy, 415-416
 - faxing, 417-419
- printing, 417-419
- transport strategy, 415-416
- R/3 Enterprise, 405-406
- SAP Knowledge Warehouse, 408
- security, 416-417
- trust relationship management, 416-417
- Component specialists, 311**
- Computer Associate's Unicenter Application Management for R/3, 525-526**
- Computer Center Management System. *See CCMS***
- Computer Operations. *See Systems Management***
- Computer-Aided Test Tool (CATT), 544-546**
- concurrent users, 227, 230, 253, 573**
- configuration**
 - Configuration Assistant, 393-395
 - MMC (Microsoft Management Console), 537-538
 - networks, 360-361
 - pagefiles, 365-366
 - servers, 362-364, 369
- Configuration Assistant, 393-395**
- conservative technology perspective, 64**
- Consultant Education, 321-322**
- consultants, 116, 294, 434-435**
 - budget considerations, 117-118
 - case studies, 118-119
 - compared to internal resources, 118-119
 - Consultant Education, 321-322
 - interviewing, 294
 - LA (letter agreement), 295-296
 - obtaining training feedback from, 333-334
 - organizational balance, 118
 - SOW (scope-of-work), 295-296
- Content Integrator, 414**
- Continental clusters, 188-189**

continuous improvement

- Continuous Improvement Services, 533
- during Go-Live, 646-647
- stress testing, 612-613

Continuous Improvement Services, 533**contractors**

- interviewing, 294
- LA (letter agreement), 295-296
- SOW (scope-of-work), 295-296

cooling Data Center facilities, 349-350**core script development, 592-596****Correction and Transport System (CTS), 491****costs. *See also budgets***

- acquisition costs, 124
- downtime costs, 169-170
- recurring costs, 125
- TCO (total cost of ownership) analysis, 123-124
 - acquisition costs, 124
 - backup/restore capabilities, 149
 - defined, 124-126
 - delta analysis, 126
 - disk subsystems, 140-141
 - distributed architecture, 146-147
 - documenting, 158-159
 - DR (disaster recovery), 130-133
 - HA (high availability), 128-130
 - heterogeneous system landscapes, 147-148
 - leasing versus purchasing, 153
 - maintenance costs, 152-153
 - manageability, 136
 - management tools, 149
 - operating systems, 141-142
 - operations, 136, 153-154
 - outsourcing, 126

people costs, 125, 151-152

performance delta analysis, 133-134

Planning CD resources, 159

recurring costs, 125

relational databases, 142-145

risk factors, 154-158

SAPGUI, 150

scalability, 134-135

security, 136

servers, 138-140

stacking, 147

standardization, 137-138

Systems Management, 153-154

TCO-driven system landscapes, 76-77

technology costs, 125

upgrades, 148-149

courses. *See classes***Crash Kits, 210-212****crashing project schedules, 592****critical paths, 271****CRM (Customer Relationship Management)**

installation, 407-408

sizing, 257-259

SPOFs (single points of failure), 205

cross-application batch schedulers, 522**CSM (Candidate Status Matrix), 282-284****CTS (Correction and Transport System), 491****current-state documentation, 505-506****custom checklists, 391-392****custom training curriculum, 323-325****customer interaction center users, 258****Customer Relationship Management. *See CRM*****cutover plans, 621-624**

D

daily checklists, 376-377

daily operations documentation, 507-509

daily task meetings, 57-58

Data Center facilities

availability considerations, 339-340

cable management, 354-355

cooling, 349-350

Data Center Infrastructure team, 334

Data Center Lead, 112

Data Center specialists, 311

development system deployment, 379

effect on SAP Solution Stack, 340-341

network infrastructure, 355-356

central systems, 360-361

fault tolerance, 356-359

optimal configuration, 360-361

network server configuration, 362-364, 369

operating systems, 364

best practices, 367-368

logical drives, 365

pagefile configuration, 365-366

partitions, 365

paper-based checklists, 376-377

physical requirements, 343-344

Planning CD resources, 379-380

power supply

backup generators, 345

PDUs (power distribution units), 345

power cables, 345

power oversights, 347-349

power requirements, 344-346

redundant power supplies, 344

UPS sizing, 346

racks

air flow, 351

best practices, 352-353

clustered servers, 353

defined, 350

design considerations, 350-351

mounting, 353

production servers, 354

rack layout, 350-351

spacing, 351

SANs (storage area networks), 369-371

best practices, 371-373

database growth, 373

drives, 372-373

fibre cables, 372

GBICs (gigabit interface connectors), 371

HBAs (host bus adapters), 371

hop counts, 372

selective storage presentation, 372

SV (Storage Virtualization), 373-375

zoning, 372

SPOFs (single points of failure), 172-173

entire Data Center, 177-178

network infrastructure, 174-175

power, 173-174

rack infrastructure, 175-177

standardization, 342-343

Technical Sandbox, 52, 71, 378

Technical Sandbox Change Management Checklist, 485

technical training, 315-317

Data Center Infrastructure team, 334

Data Center Lead, 112

Data Center specialists, 311

Data Conversion specialists, 446

- Data Replication Managers (DRMs), 189**
- data warehousing systems, 421**
- Database Administrators, 311**
- databases**
- archiving to, 420-422
 - Change Management, 488-491
 - Database Administrators, 311
 - installation, 385-386
 - management tools, 532
 - MCOD (Multiple Components, One Database), 249
 - sizing, 249-250
 - SPOFs (single points of failure), 194-195
 - 9iRAC, 198-199
 - log replication solutions, 195
 - ODR (Oracle Disaster Recovery), 196
 - Oracle 8i clustering, 197-198
 - Oracle Advanced Replication, 197
 - OSF (Oracle Failsafe), 197-198
 - SQL Server clustering, 197
 - standby databases, 195-196
 - stress testing, 576-577
 - TCO (total cost of ownership) analysis, 142-145
- Dell Open Manage, 529**
- delta analysis, 126**
- Delta Guides, 391, 508**
- delta sizing, 233**
- delta training, 310**
- Demand Planning versions, 260**
- density of disk drives, 140**
- dependency resolution, 535**
- deployment**
- development system, 379
 - SAP Go-Live
 - “as-is” documentation, 638-639
 - backup/restore processes, 629-630
- batch housekeeping jobs, 626-627
- Change Management package, 628-629
- continuous improvement, 646-647
- cutover plans, 621-624
- EarlyWatch services, 625
- feedback, 646-647
- final system updates, 627-628
- first week of Go-Live, 644
- future service/support plans, 640-641
- GoingLive service, 533, 625
- Help Desk, 637-638
- JECs (Joint Escalation Centers), 641-643
- KPIs (key performance indicators), 632-634
- monitoring, 644-646
- operations, 637-638
- output management, 630-631
- performance tracking, 634-635
- planning, 270
- Planning CD resources, 649
- post-implementation evaluation, 648
- procedural documentation, 639-640
- rewarding team members, 648-649
- SAP Enterprise management, 631-632
- SAPGUI deployment, 626
- support agreements, 641
- system lockdown, 628
- Systems Management, 631
- TSO responsibilities/roles, 635-637
- design**
- racks, 350-351
 - system landscape, 67-68
 - system landscape design, 236
- developers, training, 312**
- development consultants, obtaining training feedback from, 333**

development phase (SAP implementation), 486

Development System, 52

- deploying, 379
- documentation, 509
- training, 315-317

Dialog Steps, 227, 229

differentiators (eCATT), 554-555

disaster recovery. *See DR*

disaster tolerance, determining, 167

- costs of downtime, 169-170
- return on investment calculations, 168-169

disk controllers, 184

disk subsystem

- Campus and Metropolitan Clusters, 186-188
- cloning, 182-184
- Continental Clusters, 188-189
- density of disk drives, 140
- disk controllers, 184
- logical drives, 365
- naming conventions, 342
- partitioning, 365
- physically moving drives, 185-186
- RAID, 181-182, 243-244
- sizing, 244-246
- spindles, 244
- SANs (storage area networks), 372-373
- SPOFs (single points of failure), 180
 - Campus and Metropolitan Clusters, 186-188
 - cloning, 182-184
 - Continental Clusters, 188-189
 - disk controllers, 184
 - disk subsystem infrastructure, 184-185
 - DRMs (Data Replication Managers), 189
 - physically moving drives, 185-186
 - RAID, 181-182
 - triple mirroring, 182-184

stress testing, 576-577

virtual storage arrays, 246-248

TCO (total cost of ownership) analysis, 140-141

triple mirroring, 182-184

disk-to-disk copies, 190-191

disks, floppy, 385

distributed architecture, 146-147

Document Review Checklist, 511

Document Specialist (DS), 241

documentation, 330-332

- accessibility, 510
- best practices, 509-511
- Change Management, 472-473
- current-state documentation, 505-506
- custom checklists, 391-392
- daily operations, 507-508
- Delta Guides, 508
- development, 509
- Document Review Checklist, 511
- Documentation specialists, 312
- importance of, 33, 387-388
- installation procedures, 507-508
- InstGuides
 - new installation guides, 389
 - post-installation tasks, 390-391
 - pre-installation tasks, 390
 - upgrade guides, 390
 - Web site, 389
- maintenance, 510
- Master Guides, 388-389
- preparing for Go-Live
 - “as-is” documentation, 638-639
 - procedural documentation, 639-640
- process documentation, 509
- publishing, 510
- quick-reference checklists, 508

- regularly scheduled procedures, 508-509
SAP Operations Manual, 504-505
screen shots, 510
standards, 509
TCO (total cost of ownership) documentation, 158-159
- Documentation specialists, 312**
- Documentor, 531**
- DP (Demand Planning) versions, 260**
- DR (disaster recovery), 53, 161-163, 487**
- application-layer SPOFs (single points of failure), 206
 - best practices, 207-208
 - case studies, 214-217
 - causes of downtime, 163
 - compared to HA (high availability), 163-165
 - database SPOFs (single points of failure), 194-195
 - 9iRAC, 198-199
 - log replication solutions, 195
 - ODR (Oracle Disaster Recovery), 196
 - Oracle 8i clustering, 197-198
 - Oracle Advanced Replication, 197
 - OSF (Oracle Failsafe), 197-198
 - SQL Server clustering, 197
 - standby databases, 195-196
 - disaster tolerance, 167
 - costs of downtime, 169-170
 - return on investment calculations, 168-169
 - disk subsystem SPOFs (single points of failure), 180
 - Campus and Metropolitan Clusters, 186-188
 - cloning, 182-184
 - Continental Clusters, 188-189
 - disk controllers, 184
 - disk subsystem infrastructure, 184-185
 - DRMs (Data Replication Managers), 189
 - physically moving drives, 185-186
 - RAID, 181-182
 - triple mirroring, 182-184

people-related SPOFs (single points of failure), 208-209
Planning CD resources, 219
process SPOFs (single points of failure), 208-209
researching HA/DR options, 170-171
SAP Data Center SPOFs (single points of failure), 172-173
 entire Data Center, 177-178
 network infrastructure, 174-175
 power, 173-174
 rack infrastructure, 175-177
SAP Server SPOFs (single points of failure), 178-180
stress testing, 617
system landscape, 70-71
TCO (total cost of ownership) analysis, 130-133
testing, 213-214
training, 312, 316
DR (disaster recovery) specialists, 312, 445-446
drives
 Campus and Metropolitan Clusters, 186-188
 cloning, 182-184
 Continental Clusters, 188-189
 density of disk drives, 140
 disk controllers, 184
 logical drives, 365
 moving, 185-186
 naming conventions, 342
 RAID, 181-182, 243-244
 partitioning, 365
 SANs (storage area networks), 372-373
 sizing, 244-246
 spindles, 244

SPOFs (single points of failure), 180
 Campus and Metropolitan Clusters, 186-188
 cloning, 182-184
 Continental Clusters, 188-189
 disk controllers, 184
 disk subsystem infrastructure, 184-185
 DRMs (Data Replication Managers), 189
 physically moving drives, 185-186
 RAID, 181-182
 triple mirroring, 182-184
stress testing, 576-577
virtual storage arrays, 246-248
TCO (total cost of ownership) analysis, 140-141
triple mirroring, 182-184
DRMs (Data Replication Managers), 189
DRO (Disaster Recovery Organization), 209-210
DS (Document Specialist), 241
Dynamo utility, 576

E

EAI (Enterprise Application Integration) specialists, 440-441
EarlyWatch service, 533, 625
EBP (Enterprise Buyer), 205
eCATT (extended CATT), 545-546, 588
 BC-eCATT interface, 559
 eCATT differentiators, 554-555
 online resources, 565
Ecora Documentor, 531
EIP (Enterprise Information Portal), 400-402
e-Learning, 328

- Emergency Change Releases**, 476
- employees**. *See staffing*
- end users**, 227-230, 256-258
- active users, 227
 - concurrent users, 227, 230, 253, 573
 - CRM Internet sales users, 258
 - customer interaction center users, 258
 - high users, 228
 - low users, 228
 - medium users, 228
 - mobile sales users, 258
 - named users, 227
 - normalized SD users, 230
 - online user analysis, 581
 - ramping up, 602-603
 - think time, 223, 227
- User Education, 322
- virtual users, 583, 600-602
- ending test sessions**, 605-606
- engines, Integration Engine/Server**, 252
- Enhanced Implementation Roadmap**, 624
- Enhanced Solution Management Roadmap**, 624
- EnjoySAP initiative**, 577-579
- Enterprise Application Integration (EAI) specialists**, 440-441
- Enterprise Buyer (EPB)**, 205
- Enterprise Information Portal (EIP)**, 326-328, 400-402
- sizing, 253-254
 - stress testing, 590
- SPOFs (single points of failure), 205
- E-Procurement**, 414
- ERROR_ROUTINE.aof script**, 593
- estat tool**, 532
- estimating ROI (Return on Investment)**, 42, 44
- eTestingLabs.com Web site**, 576
- evaluating**
- Go-Live, 648
 - resumes, 281-284
- training programs**
- certification programs, 336-337
 - Data Center Infrastructure team, 334
 - development consultants, 333
 - functional consultants, 333
 - Help Desk team, 335
 - Operations team, 335
 - Security team, 334
 - specialists, 335-336
 - technical consultants, 333-334
- excessive work loads, stress testing**, 614-615
- Exchange Infrastructure (XI)**, 414
- IB (Integration Builder), 398
 - installation, 399-400
 - IS (Integration Server), 398
 - MarketSet Adapter, 399
 - SAP XI Add-On, 399
 - SAP XI Connectivity, 399
 - sizing, 252-253
 - SLD (System Landscape Directory), 399
- executing business process testing**
- cautions, 564
 - data tracking, 562
 - post-execution tasks, 562-563
 - real-world example, 563
 - SLA support, 563-564
 - Test Workbench, 561-562
- expanding staff**, 430-432
- extended CATT**. *See eCATT*
- external consultants**. *See consultants*

F

failover, stress testing, 614

fault tolerance, 356-359

faxing, 417-419

feedback

Change Management, 484-485

Go-Live, 646-647

training

certification programs, 336-337

Data Center Infrastructure team, 334

development consultants, 333

functional consultants, 333

Help Desk team, 335

Operations team, 335

Security team, 334

specialists, 335-336

technical consultants, 333-334

"fewer is better" staffing approach, 435-436

fibre cables, 372

files

journaling file systems, 190

pagefiles, 365-366

filters, 488

final system updates, 627-628

flat organizations, 109

floppy disks, bootable, 385

format command, 577

Front-End specialists, 312

Fujitsu-Siemens SAP Competency Center, 223

functional consultants, obtaining training feedback from, 333

functional SPOFs (single points of failure), 206

functional testing, 546, 549-551

functions, SubString, 601

future service/support plans, 640-641

JECs (Joint Escalation Centers), 641-643

support agreements, 641

G

GBICs (gigabit interface connectors), 371

general new-product testing, 490

gigabit interface connectors (GBICs), 371

Global ASAP, 382-383

Global Solutions Centers. *See JECs (Joint Escalation Centers)*

GlobalSAP, 45

Go-Live

"as-is" documentation, 638-639

backup/restore processes, 629-630

batch housekeeping jobs, 626-627

Change Management package, 628-629

continuous improvement, 646-647

cutover plans, 621-624

EarlyWatch services, 625

feedback, 646-647

final system updates, 627-628

first week of Go-Live, 644

future service/support plans, 640-641

GoingLive service, 533, 625

Help Desk, 637-638

JECs (Joint Escalation Centers), 641-643

KPIs (key performance indicators), 632-634

monitoring, 644-646

operations, 637-638

output management, 630-631

performance tracking, 634-635

- planning, 270
 - Planning CD resources, 649
 - post-implementation evaluation, 648
 - procedural documentation, 639-640
 - rewarding team members, 648-649
 - SAP Enterprise management, 631-632
 - SAPGUI deployment, 626
 - support agreements, 641
 - system lockdown, 628
 - Systems Management, 631
 - TSO responsibilities/roles, 635-637
 - goals of SAP implementation, 27**
 - application integration, 28
 - BPR (business process re-engineering), 29-30
 - customer benefits, 30-32
 - improved operational reporting, 28-29
 - improved strategic reporting, 29
 - stress testing, 571-572, 613
 - GoingLive service, 533, 625**
-
- ## H
- HA (high availability), 161-163**
 - application-layer SPOFs (single points of failure), 206
 - availability requirements, 166
 - availability testing, 471
 - availability tracking, 623
 - best practices, 207-208
 - case studies, 214-217
 - causes of downtime, 163
 - compared to DR (disaster recovery), 163-165
 - database SPOFs (single points of failure), 194-195
 - 9iRAC, 198-199
 - log replication solutions, 195
 - ODR (Oracle Disaster Recovery), 196
 - Oracle 8i clustering, 197-198
 - Oracle Advanced Replication, 197
 - OSF (Oracle Failsafe), 197-198
 - SQL Server clustering, 197
 - standby databases, 195-196
 - disaster tolerance, 167
 - costs of downtime, 169-170
 - return on investment calculations, 168-169
 - disk subsystem SPOFs (single points of failure), 180
 - Campus and Metropolitan Clusters, 186-188
 - cloning, 182-184
 - Continental Clusters, 188-189
 - disk controllers, 184
 - disk subsystem infrastructure, 184-185
 - DRMs (Data Replication Managers), 189
 - physically moving drives, 185-186
 - RAID, 181-182
 - triple mirroring, 182-184
 - functional SPOFs (single points of failure), 206
 - HA specialists, 312, 445-446
 - mySAP.com SPOFs (single points of failure), 200-201
 - APO Optimizer, 204
 - BW (business warehouse), 204
 - clustering components, 201-202
 - CRM (Customer Relationship Management), 205
 - EBP (Enterprise Buyer), 205
 - EP (Enterprise Portal), 205
 - HP Somersault, 202
 - ITS (Internet Transaction Server), 205
 - KM (Knowledge Management), 205
 - liveCache, 204

- PLM (Product Lifecycle Management), 205
- SAP Replicated Enqueue, 203-204
- WP (Workplace), 206
- “nines” of availability, 166
- operating system SPOFs (single points of failure)
- disk-to-disk copies, 190-191
 - Linux clusters, 194
 - MC/ServiceGuard, 192-193
 - MSCS (Microsoft Cluster Server), 191-192
 - tape shipping, 190-191
 - Tru64 cluster file system, 193-194
 - UNIX cluster approaches, 192-193
- people-related SPOFs (single points of failure), 208-209
- Planning CD resources, 219
- process SPOFs (single points of failure), 208-209
- researching HA/DR options, 170-171
- SAP Data Center SPOFs (single points of failure), 172-173
- entire Data Center, 177-178
 - network infrastructure, 174-175
 - power, 173-174
 - rack infrastructure, 175-177
- SAP Server SPOFs (single points of failure), 178-180
- Steering Committees, 167
- system landscape, 70
- TCO (total cost of ownership) analysis, 128-130
- HA (high availability) specialists, 312, 445-446**
- hard drives.** *See* **drives**
- hardware**
- Change Management, 488-491
 - delta stress tests, 606
- disk subsystem
- Campus and Metropolitan Clusters, 186-188
 - cloning, 182-184
 - Continental Clusters, 188-189
 - density of disk drives, 140
 - disk controllers, 184
 - logical drives, 365
 - naming conventions, 342
 - RAID, 181-182, 243-244
 - partitioning, 365
 - physically moving drives, 185-186
 - SANs (storage area networks), 372-373
 - sizing, 244-246
 - spindles, 244
 - SPOFs (single points of failure), 180-189
 - stress testing, 576-577
 - virtual storage arrays, 246-248
- TCO (total cost of ownership) analysis, 140-141
- triple mirroring, 182-184
- hardware sizings, 44
- hardware TCO (total cost of ownership)
- disk subsystems, 140-141
 - servers, 138-140
- HBAs (host bus adapters), 371, 489
- management tools, 529-531
- HBAs (host bus adapters), 371, 489**
- heating/cooling issues (Data Center facilities), 349-350**
- Help Desk**
- common help desk questions, 454-455
 - end-user perceptions, 456
 - Go-Live, 637-638
- Help Desk team
- obtaining feedback from, 335
 - training, 313

preparation for, 455
role of, 451-452
staffing, 453-454

heterogeneous system landscapes, 147-148

Hewlett Packard. *See HP*

high availability. *See HA*

high-level project plans, 58

high users, 228

hop counts, 372

horizontal scalability, 75

host bus adapters (HBAs), 371, 489

hot add, 142

hot replace, 142

housekeeping jobs, 626-627

how-to documentation, 509

HP (Hewlett Packard)

- CCMon (Cluster Consistency Monitor), 538-539
- Continental Clusters, 188-189
- DRMs (Data Replication Managers), 189
- Insight Manager, 529
- MC/ServiceGuard, 192-193
- OpenView VantagePoint for Windows, 526-527
- ProLiant, 347-349
- RackBuilder, 530
- RILO (Remote Insight Lights-Out) Board, 540
- SAN Management Appliance, 530
- SAP Competency Center, 223
- Somersault, 202
- Survey, 531
- Tru64 cluster file system, 193-194

I

IB (Integration Builder), 398

IBM SAP Competency Center, 224

Identity Management, 416

IDES (Information Demonstration and Education System), 320

IFS (Information File Store), 319

ignoring change control, 33-36

IMG (Implementation Guide for R/3 Customizing), 382

implementation partners, 36-38

incentive bonuses, 303

InfoCubes, 257

InfoDB (Information Database), 319

Information Demonstration and Education System (IDES), 320

Information File Store (IFS), 319

Information Technology Outsourcing. *See ITO*

infrastructure

- Infrastructure/Basis Lead, 112-115
- Infrastructure specialists, 311
- planning sessions, 268-273
 - budgets, 273
 - client contacts, 273
 - Go-Live dates, 270
 - introductions, 269
 - key milestones, 270
 - key risks, 271
 - knowledge transfer, 272
 - landscape assistance, 271
 - landscape milestones, 271
 - project administration, 270
 - project management, 269
 - project sponsorship, 269
 - project structure, 270
 - quality management, 270

- roles, 272
- timelines, 271-273
- vision and definition, 269
- in-house system management applications, 521**
- initial implementation budget, 46-47**
- in-place upgradability, 140**
- InQMy Application Server, 257**
- Insight Manager, 529**
- installation**
 - authorizations, 416-417
 - Configuration Assistant, 393-395
 - documentation, 387, 507-508
 - custom checklists, 391-392
 - importance of, 387-388
 - InstGuides, 389-391
 - Master Guides, 388-389
 - implementation methodologies
 - ASAP (Accelerated SAP), 381
 - Global ASAP, 382-383
 - SSM (SAP Solution Manager), 383-384
 - MMC (Microsoft Management Console), 536-537
 - mySAP components, 404-405
 - BI (Business Intelligence), 406-407
 - CRM (Customer Relationship Management), 407-408
 - Knowledge Warehouse, 408
 - liveCache installations, 412-413
 - PLM (Product Lifecycle Management), 408-410
 - R/3 Enterprise, 405-406
 - SCM (Supply Chain Management), 410-413
 - SEM (Strategic Enterprise Management), 413-414
 - SRM (Supplier Relationship Management), 414
- NetWeaver**
 - EIP (Enterprise Information Portal), 400-402
 - ITS (Internet Transaction Server), 403-404
 - SAP XI (Exchange Infrastructure), 398-400
 - Web AS (Application Server), 397-398
 - Web site, 397
- Planning CD resources, 422-423
- post-installation tasks, 390-391
 - archiving, 420-422
 - backup/restore solutions, 419-420
 - client strategy, 415-416
 - faxing, 417-419
 - printing, 417-419
 - transport strategy, 415-416
- pre-installation tasks, 390
- security, 416-417
- Smart Implementations, 393-395
- System Landscape, 384-385
 - database installation, 385-386
 - OS installation, 385
 - SAPinst, 386-387, 395-396
- trust relationship management, 416-417
- unattended installations, 395-396
- instances**
 - CI (Central Instance), 14, 231
 - defined, 14
 - stacking, 236
- InstGuides**
 - new installation guides, 389
 - post-installation tasks, 390-391
 - pre-installation tasks, 390
 - upgrade guides, 390
 - Web site, 389
- Integration Builder (IB), 398**
- Integration Directory, 252**

Integration Engine, 252

Integration experts, 312

Integration Repository, 252

Integration Server (IS), 398

integration testing, 546, 550

Intel Iometer, 530

internal staff. *See also staffing*

 compared to consultants, 118-119

 training, 116-117

 transferring, 293-294, 432-434

Internet Transaction Server (ITS), 403-404

interviewing job applicants

 advanced skills interview questions, 288-289

 basic sample interview questions, 287-288

 interview techniques and approaches,
 286-287

 real-world examples, 291-293

 setting up interviews, 285-286

Iometer, 530, 576

iostat tool, 532

IS (Integration Server), 398

iSizing for SAP, 224

ITO (Information Technology Outsourcing),
 84-85

 ASPs (Application Service Providers), 89

 benefits of, 86-87

 choosing outsourcers, 88

 limitations of, 87-88

 requirements, 85-86

 UDC (Utility Data Center), 89-90

ITS (Internet Transaction Server)

 installing, 403-404

 SPOFs (single points of failure), 205

J

Java

 JRE (Java Runtime Environment), 386

 JVM (Java Virtual Machine), 263

JECs (Joint Escalation Centers), 641-643

job descriptions, 281

Joint Escalation Centers (JECs), 641-643

journaling file systems, 190

JRE (Java Runtime Environment), 386

JVM (Java Virtual Machine) tuning, 263

K

kernel, 231

key figures, 260

Kintana, 483

KPIs (key performance indicators), 632-634

KVA (Kilo Volt-Amps), 346

KW (Knowledge Warehouse), 218-219, 325-326,
 408

L

LA (letter agreement), 295-296

landscape. *See system landscape*

**large business TSOs (Technical Support
 Organizations)**, 459-462

latency, 73

layers (client/server architecture), 230-231

leading edge technology perspective, 64

Learning Maps, 45

leasing, 153

letter agreement (LA), 295-296
Linux clusters, 194
liveCache, 204, 412-413
locking down system, 628
log replication solutions, 195
logged-on users, 227
logical drives, 365
LOGIN.COUNTER variable, 601
LOGIN.COUNTERTXT variable, 601
logons
 logon groups, 231
 parallel logons, 253
 staggered SAPGUI logon, 600
low users, 228

M

\$MACHINE variable, 600, 602
mainstream technology perspective, 64
maintenance
 batch housekeeping jobs, 626-627
 cost of, 152-153
 documentation, 510
manageability
 system landscapes, 78-79
 TCO (total cost of ownership) analysis, 136
management
 Change Management, 465-467
 change controls, 33-36, 56
 Change Management Analysts, 496-497
 Change Management Managers, 495-496
 communications, 477-480
 database layers, 488-491
 documentation, 472-473
 feedback, 484-485

hardware, 488-491
operating systems, 488-491
planning, 493-494
Planning CD resources, 502
practices to avoid, 500-501
preparing for, 628-629
real-world examples, 497-500
release strategy, 474-476
Review Boards, 495
SAP implementation phases, 485-487
stakeholders, 468-469
standards, 473-474
summary of goals, 468
Technical Sandbox Change Management Checklist, 485
testing, 471-472
tool sets, 482-484
transport strategies, 491-492
upgrades, 492-493
workflow, 481-482
Management Cockpit (mySAP SEM), 413
SAPinst, 386-387, 395-396
Systems Management, 511-512, 518
 best practices, 528-529
 BMC Patrol, 524-525
 CA Unicenter Application Management for R/3, 525-526
 CCMon (Cluster Consistency Monitor), 538-539
 CCMS (Computer Center Management System), 512-517
 CEN (Central Monitoring System), 515-517
 Cluster Administrator, 538-539
 Computer Operations staff, 447-450, 637-638
 database management tools, 532
 evaluating solutions, 522-524

- Go-Live, 631
hardware management tools, 529-531
HP OpenView VantagePoint for Windows, 526-527
in-house applications, 521
MMC (Microsoft Management Console), 536-538
NetIQ AppManager for R/3, 527-528
OS management tools, 531-532
out-of-band management, 540
Planning CD resources, 540-541
potential problems, 520-521
remote management tools, 539-540
requirements, 519-520
SAP Note Assistant, 534-536
SAP OS Collector, 512-513
Solution Manager, 517, 532-534
Solution Operations Services, 534
TCO analysis, 153-154
- Management Cockpit (mySAP SEM), 413**
- managers.** *See also staffing*
- Change Management Managers, 495-496
 - Project Managers, 102-103
- MarketSet Adapter, 399**
- Master Check-Off List (MCL), 19-22**
- Master Guides, 388-389**
- MC/ServiceGuard, 192-193**
- MCL (Master Check-Off List), 19-22**
- MCOD (Multiple Components, One Database), 67, 249**
- measuring progress, 57-58**
- MeatGrinder, 576**
- medium users, 228**
- medium-sized business TSOs (Technical Support Organizations), 458**
- meetings, 57-58**
- MetroClusters, 186-188**
- Metropolitan Clusters, 186-188**
- Microsoft**
- Cluster Administrator, 538-539
 - Cluster Server (MSCS), 191-192
 - MMC (Microsoft Management Console), 536-538
 - SQL Profiler, 576
 - SQLIO, 530
 - WAST (Web Application Stress Tool), 576
- milestones, 270-271**
- mirroring, 243**
- MM03.INPUT variable, 594**
- MMC (Microsoft Management Console), 536-538**
- mobile sales users, 258**
- monitoring, 448-449**
- CEN (Central Monitoring System), 516-517
 - Go-Live, 644-646
 - stress testing, 611
- MonitorIT, 531**
- mounting racks, 353**
- moving drives, 185-186**
- MSCS (Microsoft Cluster Server), 191-192**
- Multi-Channel Interface Server, 258**
- Multiple Components, One Database (MCOD), 67, 249**
- multiple production instances, 108-109**
- mySAP.com, 10-11**
- component installation, 404-405
 - authorizations, 416-417
 - BI (Business Intelligence), 406-407
 - CRM (Customer Relationship Management), 407-408
 - Knowledge Warehouse, 408
 - liveCache, 412-413
 - Planning CD resources, 422-423

- PLM (Product Lifecycle Management), 408-410
post-installation tasks, 415-422
SCM (Supply Chain Management), 410-413
SEM (Strategic Enterprise Management), 413-414
R/3 Enterprise, 405-406
security, 416-417
SRM (Supplier Relationship Management), 414
trust relationship management, 416-417
sizing, 250
BI (Business Intelligence), 255-257
CRM (Customer Relationship Management), 257-259, 407-408
Enterprise Portal, 253-254
InfoCubes, 257
Planning CD resources, 274
PLM (Product Lifecycle Management), 259-260, 408-410
queries, 256
R/3 Enterprise, 254-255
SCM (Supply Chain Management), 260-261, 410-413
SEM (Strategic Enterprise Management), 263-264, 413-414
SRM (Supplier Relationship Management), 261-263, 414
users, 256, 258
Web AS, 251
XI (Exchange Infrastructure), 252-253
SPOFs (single points of failure), 200-201
APO Optimizer, 204
BW (business warehouse), 204
clustering components, 201-202
CRM (Customer Relationship Management), 205
EBP (Enterprise Buyer), 205
EP (Enterprise Portal), 205
HP Somersault, 202
ITS (Internet Transaction Server), 205
KM (Knowledge Management), 205
liveCache, 204
PLM (Product Lifecycle Management), 205
SAP Replicated Enqueue, 203-204
WP (Workplace), 206
stress testing, 579-580, 585-586, 590
- ## N
- named users, 227**
naming conventions, 342
Napkin Plans, 50-51
NAS (Network-Attached Storage), 141
navigation steps per hour, 256
near-line storage facilities, 421
negative testing, 558
NetIQ AppManager for R/3, 527-528
NetWeaver, 397
EIP (Enterprise Information Portal), 400-402
ITS (Internet Transaction Server), 403-404
Web AS (Application Server), 397-398
Web site, 397
XI (Exchange Infrastructure)
IB (Integration Builder), 398
installation, 399-400
IS (Integration Server), 398
MarketSet Adapter, 399
SAP XI Add-On, 399
SAP XI Connectivity, 399
SLD (System Landscape Directory), 399

Network-Attached Storage (NAS), 141

Network specialists, 311

networks

- NAS (Network-Attached Storage), 141
- network infrastructure, 355-356
 - central systems, 360-361
 - fault tolerance, 356-359
 - optimal configuration, 360-361
- Network specialists, 311
- SANs (storage area networks), 369-371
 - best practices, 371-373
 - database growth, 373
 - drives, 372-373
 - fibre cables, 372
 - GBICs (gigabit interface connectors), 371
 - HBAs (host bus adapters), 371
 - hop counts, 372
 - selective storage presentation, 372
 - SV (Storage Virtualization), 373-375
 - zoning, 372
- server configuration, 362-364, 369
- SPOFs (single points of failure), 174-175
- stress testing, 577-579

new hires

- orientation, 297-298
- preparing for, 296-297

New Project personality types, 299

new product testing, 490

noise scripts, 608

normalized SD users, 230

Note Assistant, 534-536

NP (New Project) personality types, 299

O

ODR (Oracle Disaster Recovery), 196

ODS (Operational Data Store), 256

onsite training workshops, 323

Open Manage, 529

OpenView VantagePoint for Windows, 526-527

operating systems, 364

- best practices, 367-368
- Change Management, 488-491
- installation, 385
- logical drives, 365
- management tools, 531-532
- OS specialists, 311
- pagefile configuration, 365-366
- partitions, 365
- sizing, 248-249
- SPOFs (single points of failure)
 - disk-to-disk copies, 190-191
 - Linux clusters, 194
 - MC/ServiceGuard, 192-193
 - MSCS (Microsoft Cluster Server), 191-192
 - tape shipping, 190-191
 - Tru64 cluster file system, 193-194
 - UNIX cluster approaches, 192-193
- stress testing, 575-576
- TCO analysis, 141-142

Operational Data Store (ODS), 256

Operations. *See Systems Management*

Operations Manual, 504-505

Operations staff, 447-448

- day-to-day tasks, 450
- obtaining training feedback from, 335
- potential for advancement, 450
- preparing for Go-Live, 637-638

SAP monitoring, 448-449
training, 313

Oracle

Advanced Replication, 197
ODR (Oracle Disaster Recovery), 196
Oracle 8i clustering, 197-198
OSF (Oracle Failsafe), 197-198

orientation (new hires), 297-298

OS Collector, 512-513

OSF (Oracle Failsafe), 197-198

OSI model, 341

OSs. *See operating systems*

out-of-band management, 540

output management, 630-631

outsourcing

ITO (Information Technology Outsourcing), 84-85
ASPs (Application Service Providers), 89
benefits of, 86-87
choosing outsourcers, 88
limitations of, 87-88
requirements, 85-86
UDC (Utility Data Center), 89-90

TCO (total cost of ownership) analysis, 126

ownerships, costs of, 123-124

acquisition costs, 124
backup/restore capabilities, 149
defined, 124-126
delta analysis, 126
disk subsystems, 140-141
distributed architecture, 146-147
documenting, 158-159
DR (disaster recovery), 130-133
HA (high availability), 128-130
heterogeneous system landscapes, 147-148
leasing versus purchasing, 153

maintenance costs, 152-153
manageability, 136
management tools, 149
operating systems, 141-142
operations, 136, 153-154
outsourcing, 126
people costs, 125, 151-152
performance delta analysis, 133-134
Planning CD resources, 159
recurring costs, 125
relational databases, 142-145
risk factors, 154-158
SAPGUI, 150
scalability, 134-135
security, 136
servers, 138-140
stacking, 147
standardization, 137-138
Systems Management, 153-154
TCO-driven system landscapes, 76-77
technology costs, 125
upgrades, 148-149

P

pagefiles, 365-366

paper-based checklists, 376-377

parallel logons, 253

parallelization, 234

parity, striping with, 244

partitions, 365-366

partners, SAP Solution Stack Partners, 264-268

production references, 266

risk level, 266

sizing data, 265

- specialized vendors, 268
- support for architected solutions, 265-266
- TCO analyses, 266-267
- Patrol**, 524-525
- PDU**s (power distribution units), 345
- people-related SPOFs** (single points of failure), 208-209
- per test scenarios**, 547
- PerfMon** (Performance Monitor), 531
- performance**
 - baselines, 572
 - KPIs (key performance indicators), 632-634
 - latency, 73
 - PerfMon (Performance Monitor), 531
 - performance bonuses, 303
 - performance-driven system landscapes, 73-74
 - Performance specialists, 441-443
 - Performance Stress Test Utility, 576
 - response times, 73
- SAPS (SAP Application Benchmark Performance Standard), 226
- TCO (total cost of ownership) analysis, 133-134
- throughput, 73
- tracking, 634-635
- Performance Monitor** (PerfMon), 531
- Performance specialists**, 441-443
- Performance Stress Test Utility**, 576
- performance-driven system landscapes**, 73-74
- periods of time**, 257, 261
- Persistent Staging Area** (PSA), 256
- personality types**
 - NP (New Project), 299
 - SM (Support/Maintenance), 299-300
- phone screen**, 284-285
- Pilot phase** (SAP implementation), 485
- ping command**, 403
- planning**
 - business process testing, 551-553
 - capacity planning, 237, 263
 - Change Management, 493-494
 - cutover plans, 621-624
 - Demand Planning versions, 260
 - Planning CD resources, 19, 58-59
 - business process testing, 564
 - Change Management, 502
 - component installation, 422-423
 - Data Center facilities, 379-380
 - DR (disaster recovery), 219
 - Go-Live, 649
 - HA (high availability), 219
 - SAP TSOs, 120
 - sizing, 274
 - staffing, 307, 462
 - stress testing, 619
 - Systems Management, 540-541
 - TCO analysis, 159
 - training, 337
- Planning Runs**, 260
- SAP implementation**
 - business requirements, 39-40
 - Business Sandbox, 52
 - business unit buy-in, 49-50
 - change control, 56
 - daily task meetings, 57-58
 - Development System, 52
 - DR (Disaster Recovery) System, 53
 - estimated ROI (Return on Investment), 42-44
 - initial implementation budget, 46-47
 - methodologies, 44-46

- Napkin Plans, 50-51
 - Planning CD resources, 58-59
 - Production System, 53
 - Project Sponsors, 38-39
 - realistic SLAs (Service Level Agreements), 41-42
 - SAP system roles and purposes, 52-54
 - SAP TSOs (Technical Support Organizations), 56-57
 - SIPP (SAP Implementation Project Plan), 57-58
 - solution characteristics matrix, 40-41
 - solution sizing, 54-55
 - Staging System, 53
 - status update meetings, 57-58
 - Steering Committee, 47-49
 - Technical Sandbox System, 52
 - technology drivers, 44
 - Test/QA System, 52
 - timelines, 51
 - Training System, 53
 - SAP infrastructure, 268-273
 - budgets, 273
 - client contacts, 273
 - Go-Live dates, 270
 - introductions, 269
 - key milestones, 270
 - key risks, 271
 - knowledge transfer, 272
 - landscape assistance, 271
 - landscape milestones, 271
 - project administration, 270
 - project management, 269-270
 - project sponsorship, 269
 - quality management, 270
 - roles, 272
 - timelines, 271-273
 - vision and definition, 269
 - stress testing project plans
 - creating, 580-581
 - updating, 582-583
 - Supply Network Planning versions, 260
- Planning CD resources, 19, 58-59**
- business process testing, 564
 - Change Management, 502
 - component installation, 422-423
 - Data Center facilities, 379-380
 - DR (disaster recovery), 219
 - Go-Live, 649
 - HA (high availability), 219
 - SAP TSOs, 120
 - sizing, 274
 - staffing, 307, 462
 - stress testing, 619
 - Systems Management, 540-541
 - TCO analysis, 159
 - training, 337
- Planning Runs, 260**
- PLM (Product Lifecycle Management)**
- installation, 408-410
 - sizing, 259-260
 - SPOFs (single points of failure), 205
- PMD (physically moving drives), 185-186**
- PMs (project managers), 102-103**
- POC (proof of concept), 233-234**
- portals, 158**
- positive feedback, 302-303**
- post-implementation Go-Live evaluation, 648**
- post-installation tasks**
- archiving, 420-422
 - authorizations, 416-417
 - backup/restore solutions, 419-420

client strategy, 415-416
faxing, 417-419
printing, 417-419
security, 416-417
transport strategy, 415-416
trust relationship management, 416-417

power distribution units (PDUs), 345

power supply

- Data Center facilities
 - backup generators, 345
 - PDUs (power distribution units), 345
 - power cables, 345
 - power oversights, 347-349
 - power requirements, 344-346
 - redundant power supplies, 344
 - UPS sizing, 346
- SPOFs (single points of failure), 173-174

power users, 228

PreciseSoft Web site, 580

Predictive and Proactive Services, 533

preparing

- for sizing, 237
- for Test Week, 609-610

Presentation layer (client/server architecture), 231

pre-sizing conference calls, 237-240

pre-tuning, 574

printing, 417-419

probationary period approach, 277

procedural documentation, 639-640

processes. *See* business processes

Product Lifecycle Management. *See* PLM

production, 53, 487. *See also* stress testing

- multiple production instances, 108-109
- production references, 266
- production servers, 354

production support, 625
training, 316

progress, measuring, 57-58

Project Plan. *See* SIPP

project portals, 158

Project Sponsors, 38-39

project sponsorship, 38-39, 269

project teams. *See* staffing

ProLiant, 347-349

promotion from within, 110

proof of concept (POC), 233-234

proxy generation, 252

PROXYCFG.exe, 402

PSA (Persistent Staging Area), 256

publishing documentation, 510

Q

Quality Assurance phase (SAP implementation), 486

queries, 256

questionnaires. *See* Sizing Questionnaires

QuickSizer, 224, 232

quick-reference checklists, 508

R

R/3 Enterprise, 10, 14

- installation, 405-406
- sizing, 254-255

RackBuilder, 530

racks

- air flow, 351
- best practices, 352-353

- clustered servers, 353
- defined, 350
- design considerations, 350-351
- mounting, 353
- production servers, 354
- rack layout, 350-351
- spacing, 351
- SPOFs (single points of failure), 175-177
- RAID (redundant array of inexpensive disks), 181-182, 243-244**
- Ramp-Up Knowledge Transfer (RKT), 383**
- ramping up users, 602-603**
- random number generators, 606**
- ranking prospective candidates, 289-291**
- RDSSP (Rapid Deployment SAP Staffing Process), 278-281**
 - CSM (Candidate Status Matrix), 282-284
 - interviews
 - advanced skills interview questions, 288-289
 - basic sample interview questions, 287-288
 - interview techniques and approaches, 286-287
 - real-world examples, 291-293
 - setting up, 285-286
 - job descriptions, 281
 - ranking prospective candidates, 289-291
 - resume evaluation, 281-284
 - technical phone screen, 284-285
- real application clusters (9iRAC), 198-199**
- real-world examples. *See* case studies**
- realistic SLAs (Service Level Agreements), 41-42**
- Realization, 381**
- Recovery Managers, 209**
- recurring costs, 125**
- redundancy, 110, 614**
- redundant array of inexpensive disks (RAID), 181-182, 243-244**
- regression testing, 546, 550-551**
- relational databases**
 - archiving to, 420-422
 - Change Management, 488-491
 - Database Administrators, 311
 - installation, 385-386
 - management tools, 532
 - MCOD (Multiple Components, One Database), 249
 - sizing, 249-250
 - SPOFs (single points of failure), 194-195
 - 9iRAC, 198-199
 - log replication solutions, 195
 - ODR (Oracle Disaster Recovery), 196
 - Oracle 8i clustering, 197-198
 - Oracle Advanced Replication, 197
 - OSF (Oracle Failsafe), 197-198
 - SQL Server clustering, 197
 - standby databases, 195-196
 - stress testing, 576-577
 - TCO (total cost of ownership) analysis, 142-145
- release strategy, 474-476**
- Remote Insight Lights-Out (RILO) Board, 540**
- remote management tools, 539-540**
- Replicated Enqueue, 203-204**
- reporting**
 - operational reporting, 28-29
 - strategic reporting, 29
 - Top Sessions reports, 532
- Request for Information (RFI), 80-83, 222-223**
- response times, 73**
- restore solutions, 419-420**
 - Go-Live, 629-630
 - stress testing, 616-617

resume-to-interview staffing process, 277
resumes, evaluating, 281-284
retaining employees, 151-152, 298
 career pathing, 304
 communication, 300-301
 compensation alternatives, 304-307
 New Project personality types, 299
 performance bonuses, 303
 positive feedback, 302-303
 salary requirements, 302
 Support/Maintenance personality types, 299-300
 training, 303
return on investment calculations
Return on Investment (ROI), 42-44, 168-169
Review Boards, 495
review process, 63, 241-242
Review/Certification Managers, 209
RFI (Request for Information), 80-83, 222-223
RILO (Remote Insight Lights-Out) Board, 540
risk
 identifying, 271
 risk factors, 154-158
 TCO (total cost of ownership) analysis, 154-158
 verifying, 266
RKT (Ramp-Up Knowledge Transfer), 383
roadmaps
 Enhanced Implementation Roadmap, 624
 Enhanced Solution Management Roadmap, 624
ROI (Return on Investment), 42-44, 168-169
roles (Go-Live), 636-637
RWD Technologies, 328

S

salary requirements, 302
sandboxes, 378
 Business Sandbox, 52, 72
 Development System, 52
 Technical Sandbox, 52, 71
 Change Management Checklist, 485
 technical training, 315-317
SANs (storage area networks), 369-371
 best practices, 371-373
 database growth, 373
 drives, 372-373
 fibre cables, 372
 GBICs (gigabit interface connectors), 371
 HBAs (host bus adapters), 371
 hop counts, 372
 SAN Management Appliance, 530
 SAN specialists, 311
 selective storage presentation, 372
 SV (Storage Virtualization), 373-375
 zoning, 372
SAP Application Benchmark Performance Standard (SAPS), 226
SAP-aware testing tools, 583-585
SAP Collaboration Engine (SCE), 414
SAP Infrastructure Project Plan. *See* SIPP
SAP Notes Web site, 405
SAP Operations Manual, 504-505
SAP Solution Manager (SSM), 383-384, 517, 532-534
SAP Web site, 564
SAPGUI
 defined, 14
 deployment, 626

staggered SAPGUI login, 600
TCO (total cost of ownership) analysis, 150
SAPinst, 386-387, 395-396
SAPLOGIN.aof script, 593
SAProuters, 422
SAPS (SAP Application Benchmark Performance Standard), 226
"saptp - Compuware TestPartner and SAP eCATT" (white paper), 564-565
SAAs (Solution Architects), 112-114, 240
scalability
 horizontal scalability, 75
 system landscapes, 74-76
 TCO (total cost of ownership) analysis, 134-135
 vertical scalability, 75
SCE (SAP Collaboration Engine), 414
scheduling
 crashing schedules, 592
 cross-application batch scheduler, 522
 documentation, 508-509
 job interviews, 285-286
SCM (Supply Chain Management)
 installation, 410-413
 sizing, 260-261
scope creep, 34
scope-of-work (SOW), 295-296
scripts
 business process scripts, 585, 591-592
 administrative scripts, 599-600
 core script development, 592-596
 scripting tips, 606-609
 utility scripts, 599-600
 ERROR_ROUTINE.aof, 593
 "noise" scripts, 608
 SAPLOGIN.aof, 593
 scripted OS installations, 385
SD Dialog Steps, 227-229
SE38, 588
Secure Systems Management, 416
security
 mySAP component installation, 416-417
 Secure Systems Management, 416
 Security specialists, 443-445
 obtaining feedback from, 334
 training, 312
 system landscapes, 77-78
 TCO (total cost of ownership) analysis, 136
Security specialists, 443-445
 obtaining feedback from, 334
 training, 312
Segue's SilkPerformer,, 585
selective storage presentation, 372
SEM (Strategic Enterprise Management)
 installation, 413-414
 sizing, 263-264
Senior DBA (Database Administrator), 112
Senior Solution Architects, 240
servers
 Application Servers, 230
 clustered servers, 353
 clustering, 179-180
 communication stations, 257
 HP ProLiant, 347-349
 InQMy Application Server, 257
 Integration Engine/Server, 252
 IS (Integration Server), 398
 ITS (Internet Transaction Server), 403-404
 liveCache server, 412-413
 logon groups, 231
 Multi-Channel Interface Server, 258
 naming conventions, 342
 production servers, 354

- sizing, 242-243
 - SPOFs (single points of failure), 178-180
 - stress testing, 575-576
 - TCO analysis, 138-140
 - three-tiered client/server architecture, 230-231
 - TRex Server, 257
 - Web AS (Application Server), 251, 397-398
 - workgroup servers, 257
- Service Level Agreements.** *See SLAs*
- Service Marketplace, 401**
- services**
- Continuous Improvement Services, 533
 - Predictive and Proactive Services, 533
 - Service Marketplace, 401
 - SLAs (Service Level Agreements)
 - determining realistic SLAs, 632-634
 - planning, 41-42
 - supporting with business process testing, 563-564
 - Solution Operations Services, 534
- sessions, 227**
- SET (Sizing Evaluation Team), 240-241**
- Setup Manager Wizard, 385**
- setup/installation GUI, 140**
- SilkPerformer,, 585**
- Simple Project Staffing Costing Matrix.xls file, 159**
- simplifying system landscape, 68, 70**
- single points of failure.** *See SPOFs*
- SIPP (SAP Implementation Project Plan), 19-22, 84**
- daily task meetings, 57-58
 - high-level project plans, 58
 - status update meetings, 57-58
- stress testing plans
 - creating, 580-581
 - updating, 582-583
- sizing, 44, 221-226**
- assumptions, 234-236
 - best practices, 230-231
 - BI (Business Intelligence), 255-257
 - budget issues, 219, 232
 - characterization testing, 234
 - CRM (Customer Relationship Management), 257-259
 - databases, 249-250
 - delta sizing, 233
 - disk subsystems, 244-246
 - Enterprise Portal, 253-254
 - InfoCubes, 257
 - infrastructure planning sessions, 268-273
 - budgets, 273
 - client contacts, 273
 - Go-Live dates, 270
 - introductions, 269
 - key milestones, 270
 - key risks, 271
 - knowledge transfer, 272
 - landscape assistance, 271
 - landscape milestones, 271
 - project administration, 270
 - project management, 269
 - project sponsorship, 269
 - project structure, 270
 - quality management, 270
 - roles, 272
 - timelines, 271-273
 - vision and definition, 269

- Knowledge Repository, 218-219
operating systems, 248-249
Planning CD resources, 274
PLM (Product Lifecycle Management), 259-260
POC (proof of concept), 233-234
pre-sizing conference calls, 237-240
preparing for, 217-218, 237
queries, 256
Quick Sizer, 224, 232
R/3 Enterprise, 254-255
RAID, 243-244
review process, 241-242
RFI (Request for Information), 222-223
SCM (Supply Chain Management), 260-261
SD Dialog Steps, 227-229
SEM (Strategic Enterprise Management), 263-264
servers, 242-243
sessions, 227
SET (Sizing Evaluation Team), 240-241
Sizing Evaluation document, 21
Sizing Questionnaires, 222-224
solution sizing, 54-55
Solution Stack Partners, choosing, 264-268
 production references, 266
 risk level, 266
 sizing data, 265
 specialized vendors, 268
 support for architected solutions, 265-266
 TCO analyses, 266-267
SRM (Supplier Relationship Management), 261-263
storage virtualization, 246-248
system landscape design, 236
transaction-based sizing, 232-233
users, 227-230, 256-258
 active users, 227
 concurrent users, 227, 230, 253
 CRM Internet sales users, 258
 customer interaction center users, 258
 high users, 228
 low users, 228
 medium users, 228
 mobile sales users, 258
 named users, 227
 normalized SD users, 230
 think time, 223, 227
Web AS (Application Server), 251
workflow, 221-222
XI (Exchange Infrastructure), 252-253
- Sizing Evaluation document, 21**
- Sizing Evaluation Team (SET), 240-241**
- Sizing Questionnaires, 81, 222-224**
- skillset backups, 313**
- slack time, 271**
- SLAs (Service Level Agreements)**
- determining realistic SLAs, 632-634
 - planning, 41-42
 - supporting with business process testing, 563-564
- SLD (System Landscape Directory), 399**
- SM (Support/Maintenance) personality types, 299-300**
- SM21 transactions, 611**
- SM66 transactions, 611**
- small business TSOs (Technical Support Organizations), 456-457**
- Smart Implementations, 393-395**
- SMEs (Subject Matter Experts), 115, 240**
- SmithMicro's Checklist Utilities, 531, 576**
- SNP (Supply Network Planning) versions, 260**

software

- AppManager for R/3, 527-528
- AutoController, 584, 597-599
- AutoIT, 515
- BMC Patrol, 525
- bstat utility, 532
- CATT, 544, 546
- CCMon (Cluster Consistency Monitor), 538-539
- Change Management tools, 482-484
- CheckIt Utilities, 531, 576
- Cluster Administrator, 538-539
- Documentor, 531
- Dynamo, 576
- eCATT (extended computer aided test tool), 545-546, 588
 - BC-eCATT interface, 559
 - eCATT differentiators, 554-555
 - online resources, 565
- estat, 532
- Insight Manager, 529
- Iometer, 530, 576
- iostat, 532
- MeatGrinder, 576
- MMC (Microsoft Management Console), 536-538
- MonitorIT, 531
- ONE, 514-515
- Open Manage, 529
- OpenView VantagePoint for Windows, 526-527
- Patrol, 524-525
- PerfMon (Performance Monitor), 531
- Performance Stress Test Utility, 576
- PROXYCFG.exe, 402
- RackBuilder, 530
- remote management tools, 539-540
- RILO (Remote Insight Lights-Out) Board, 540
- SAN Management Appliance, 530
- SAP-aware testing tools, 583-585
- SE38, 588
- SQL Profiler, 576
- SQLIO, 530, 576
- Sun Net Manager, 529
- Survey, 531
- Thrasher, 576
- Unicenter Application Management for R/3, 525-526
- vmstat, 532

Solution Architects (SAs), 112-114, 240**solution awareness education, 321****solution characteristics matrix, 40-41****Solution Manager (SSM), 45, 383-384, 517, 532-534****Solution Manager Learning Maps, 384****Solution Operations Services, 534****solution sizing, 44, 54-55****Solution Stack, 9, 12-13, 25, 51-52. *See also solution vision***

- change control, 56

- Change Management

- database layers, 488-491

- hardware, 488-491

- operating systems, 488-491

- transport strategies, 491-492

- upgrades, 492-493

- compared to OSI model, 341

- role of SAP TSOs (Technical Support Organizations), 56-57

- SAP Basis experts, 311

- SAP system roles and purposes, 52-54

- Solution Stack Partners, choosing, 264-268
 production references, 266
 risk level, 266
 sizing data, 265
 specialized vendors, 268
 support for architected solutions, 265-266
 TCO analyses, 266-267
- solution sizing, 44, 54-55
- system landscape
 accessibility, 79
 design, 67-68
 disaster recovery, 70-71
 high availability, 70
 manageability, 78-79
 MCOD (Multiple Components, One Database), 67
 performance-driven system landscapes, 73-74
 requirements, 65-66
 scalability, 74-76
 security, 77-78
 simplifying, 68-70
 TCO-driven system landscapes, 76-77
 training requirements, 71, 73
- TCO (total cost of ownership) analysis, 136-137
 backup/restore capabilities, 149
 disk subsystems, 140-141
 distributed architecture, 146-147
 heterogeneous system landscapes, 147-148
 management tools, 149
 operating systems, 141-142
 relational databases, 142-145
 SAPGUI, 150
 servers, 138-140
 stacking, 147
- standardization, 137-138
upgrades, 148-149
training issues, 318-319
- Solution Stack Partners, choosing, 264-268**
- production references, 266
 risk level, 266
 sizing data, 265
 specialized vendors, 268
 support for architected solutions, 265-266
 TCO analyses, 266-267
- solution vision, 61-63. See also Solution Stack**
- impact on business, 63
 ITO (Information Technology Outsourcing), 84-85
 ASPs (Application Service Providers), 89
 benefits of, 86-87
 choosing outsourcers, 88
 limitations of, 87-88
 requirements, 85-86
 UDC (Utility Data Center), 89-90
 mySAP components, 65
 review process, 63
 RFI (Request for Information), 80-83
 SAP Sizing Questionnaires, 81
 SIPP (SAP Infrastructure Implementation Budget), 84
 solution characteristics matrix, 40-41
 system landscape requirements, 65-66
- TCO (total cost of ownership) analysis, 127
 DR (disaster recovery), 130-133
 HA (high availability), 128-130
 manageability, 136
 operations, 136
 performance delta analysis, 133-134
 scalability, 134-135
 security, 136
- technology perspectives, 63-65

Somersault (HP), 202

SOW (scope-of-work), 295-296

specialists, 438. *See also staffing*

- Component Basis specialists, 438, 440
- Component Functional specialists, 311, 440
- Component specialists, 311
- Data Center specialists, 311
- Data Conversion specialists, 446
- Disaster Recovery specialists, 312, 445-446
- Documentation specialists, 312
- EAI (Enterprise Application Integration) specialists, 440-441
- Front-End specialists, 312
- High Availability specialists, 312, 445-446
- Infrastructure specialists, 311
- Network specialists, 311
- obtaining training feedback from, 335-336
- OS (operating system) specialists, 311
- Performance specialists, 441-443
- SAN specialists, 311
- Security specialists, 312, 443-445
- Storage specialists, 311
- Testing specialists, 446-447
- User Interface Deployment specialists, 312

specialized Solution Stack vendors, 268

spindles, 244

SPOFs (single points of failure), 171

- application-layer SPOFs, 206
- databases, 194-195
 - 9iRAC, 198-199
 - log replication solutions, 195
 - ODR (Oracle Disaster Recovery), 196
 - Oracle 8i clustering, 197-198
 - Oracle Advanced Replication, 197
 - OSF (Oracle Failsafe), 197-198

SQL Server clustering, 197

standby databases, 195-196

disk subsystem, 180

- Campus and Metropolitan Clusters, 186-188
- cloning, 182-184
- Continental Clusters, 188-189
- disk controllers, 184
- disk subsystem infrastructure, 184-185
- DRMs (Data Replication Managers), 189
- physically moving drives, 185-186
- RAID, 181-182
 - triple mirroring, 182-184
- functional SPOFs, 206
- mySAP.com, 200-201
 - APO Optimizer, 204
 - BW (business warehouse), 204
 - clustering components, 201-202
 - CRM (Customer Relationship Management), 205
 - EBP (Enterprise Buyer), 205
 - EP (Enterprise Portal), 205
 - HP Somersault, 202
 - ITS (Internet Transaction Server), 205
 - KM (Knowledge Management), 205
 - liveCache, 204
 - PLM (Product Lifecycle Management), 205
 - SAP Replicated Enqueue, 203-204
 - WP (Workplace), 206

operating systems

- disk-to-disk copies, 190-191
- Linux clusters, 194
- MC/ServiceGuard, 192-193
- MSCS (Microsoft Cluster Service), 191-192
- tape shipping, 190-191
- Tru64 cluster file system, 193-194
- UNIX cluster approaches, 192-193

- people-related SPOFs, 208-209
- process SPOFs, 208-209
- SAP Data Centers, 172-173
 - entire Data Center, 177-178
 - network infrastructure, 174-175
 - power, 173-174
 - rack infrastructure, 175-177
- SAP Servers, 178-180
- sponsorship, 269**
- spreadsheets, 21**
- SQL Profiler, 576**
- SQL Server**
 - clustering, 197
 - TCO analysis, 144-145
- SQLIO, 530, 576**
- SRM (Supplier Relationship Management)**
 - installation, 414
 - sizing, 261-263
 - SRM SCE (SRM Supplier Collaboration Engine), 414
 - stress testing, 590
- SSM (Solution Manager), 45, 383-384, 517, 532-534**
- ST03 transactions, 611**
- ST04 transactions, 611**
- ST06 transactions, 611**
- ST11 transactions, 611**
- ST22 transactions, 611**
- stacking, 147, 236**
- staff-testing approach, 277**
- staffing, 275-277, 425-426.** *See also training*
 - “bigger than me” approach, 437-438
 - Change Management Analysts, 496-497
 - Change Management Managers, 495-496
 - Component Basis specialists, 438, 440
 - Component Functional specialists and programmers, 440
 - costs, 125
 - CSM (Candidate Status Matrix), 282-284
 - Data Conversion specialists, 446
 - Disaster Recovery specialists, 445-446
 - DRO (Disaster Recovery Organization), 209-210
 - EAI (Enterprise Application Integration) specialists, 440-441
 - employee new-hires
 - orientation, 297-298
 - preparing for, 296-297
 - employee retention, 151-152, 298
 - career pathing, 304
 - communication, 300-301
 - compensation alternatives, 304-307
 - New Project personality types, 299
 - performance bonuses, 303
 - positive feedback, 302-303
 - salary requirements, 302
 - Support/Maintenance personality types, 299-300
 - training, 303
 - examples
 - Fortune 500 global TSO, 459-462
 - medium business TSO, 458
 - small business TSO, 456-457
 - external consultants and contractors
 - interviewing, 294
 - LA (letter agreement), 295-296
 - SOW (scope-of-work), 295-296
 - “fewer is better” approach, 435-436
 - Help Desk, 453-454
 - end-user perceptions, 456
 - preparation, 455
 - role of Help Desk, 451-452
 - training, 454-455
 - High Availability specialists, 445-446

- impact of business process testing, 557-558
- internal transfers, 293-294
- interviews
 - advanced skills interview questions, 288-289
 - basic sample interview questions, 287-288
 - interview techniques and approaches, 286-287
 - real-world examples, 291-293
 - setting up, 285-286
- job descriptions, 281
- Operations staff, 447-448
 - day-to-day tasks, 450
 - potential for advancement, 450
 - SAP monitoring, 448-449
- Performance specialists, 441-443
- Planning CD resources, 307 462
- probationary period approach, 277
- ranking prospective candidates, 289-291
- RDSSP (Rapid Deployment SAP Staffing Process), 278-281
- resume evaluation, 281-284
- resume-to-interview process, 277
- rewarding staff after Go-Live, 648-649
- Security specialists, 443-445
- SET (Sizing Evaluation Team), 240-241
- staff expansion, 430-432
- staff size, 427-428
- staff-testing approach, 277
- staffing holes, 426-427
- TCO (total cost of ownership) analysis, 151-152
- technical phone screen, 284-285
- Testing specialists, 446-447
- transferring internal staff, 432-434
- try-before-you-buy approach, 277
- staggered SAPGUI login, 600
- Staging System, 53, 315
- stakeholders, 468-469
- Standard Application Benchmarks, 586-587
- standards
 - Change Management, 473-474
 - Data Center facilities, 342-343
 - documentation, 509
 - Standard Application Benchmarks, 586-587
 - standard test plans, 489
 - TCO (total cost of ownership) analysis, 137-138
- standby databases, 195-196
- statistics, collecting, 603-605
- status update meetings, 57-58
- Steering Committee, 47-49
- stopping test sessions, 605-606
- storage area networks. *See SANs*
- Storage specialists, 311
- Storage Virtualization (SV), 246-248, 373-375
- Strategic Enterprise Management. *See SEM*
- strategic goals of SAP implementation, 27
 - application integration, 28
 - BPR (business process re-engineering), 29-30
 - customer benefits, 30-32
 - improved operational reporting, 28-29
 - improved strategic reporting, 29
- stress testing, 543, 569-571
 - administrative scripts, 599-600
 - background noise, 608
 - backup/restore process, 616-617
 - batch processes, 581
 - business information warehouse, 588-589
 - business process scripts, 585, 591-592
 - client infrastructure, 596-599

- compared to functional testing, 551
component-layer testing, 579-580, 585-586, 590
for continuous improvement, 612-613
core script development, 592-596
costs, 570
data requirements, 581-582
databases, 576-577
disaster recovery plans, 617
disk subsystem, 576-577
eCATT, 588
ending test sessions, 605-606
Enterprise Portal, 590
excessive loads, 614-615
goals, 571-572, 613
hardware delta stress tests, 606
monitoring, 611
network infrastructure testing, 577-579
online user analysis, 581
operating system utilities, 611-612
operating systems, 575-576
performance baselines, 572
Planning CD resources, 619
pre-tuning, 574
project plans
 creating, 580-581
 updating, 582-583
ramping up users, 602-603
real-world examples, 617-619
SAP-aware testing tools, 583-585
SAP Standard Application Benchmarks, 586-587
scripting tips, 606-609
SE38, 588
server hardware, 575-576
- SRM (Strategic Relationship Management), 590
statistics, 603-605
success criteria, 573-574
system redundancy/failover, 614
system-level stress testing, 574-575
Test Week
 preparing for, 609-610
 testing tasks, 610-613
utility scripts, 599-600
virtual users, 583, 600-602
what-if approach, 614
- striping data, 244**
- Subject Matter Experts (SMEs), 115, 240**
- SubString function, 601**
- success criteria, 573-574**
- Sun**
 Net Manager, 529
 SAP Competency Center, 224
- supersizing, 134**
- Supplier Relationship Management. *See* SRM**
- Supply Chain Management. *See* SCM**
- Supply Network Planning versions, 260**
- support agreements, 641**
- Support Center. *See* Help Desk**
- Support/Maintenance personality types, 299-300**
- Survey tool, 531**
- SV (Storage Virtualization), 246-248, 373-375**
- system landscape**
 accessibility, 79
 defined, 14
 design, 67-68, 236
 disaster recovery, 70-71
 high availability, 70

- installation, 384-385
 database installation, 385-386
 OS installation, 385
 SAPinst, 386-387, 395-396
- manageability, 78-79
- MCOD (Multiple Components, One Database), 67
- performance-driven system landscapes, 73-74
- requirements, 65-66
- scalability, 74-76
- security, 77-78
- simplifying, 68, 70
- SLD (System Landscape Directory), 399
- TCO-driven system landscapes, 76-77
- three-system landscapes, 14
- training requirements, 71-73
- System Landscape Directory (SLD), 399**
- system-level stress testing, 574-575**
- system lockdown, 628**
- System Management professionals, 313**
- system redundancy, 614**
- Systems Management, 511-512, 518**
 best practices, 528-529
 BMC Patrol, 524-525
 CA Unicenter Application Management for R/3, 525-526
 CCMon (Cluster Consistency Monitor), 538-539
 CCMS (Computer Center Management System)
 automated data collection, 513-515
 CEN (Central Monitoring System), 515-517
 manual processes/checklists, 512-513
 transaction monitors, 515-517
 CEN (Central Monitoring System), 515-517
 Cluster Administrator, 538-539
- Computer Operations staff, 447-448
 day-to-day tasks, 450
 obtaining training feedback from, 335
 potential for advancement, 450
 preparing for Go-Live, 637-638
 SAP monitoring, 448-449
 training, 313
- database management tools, 532
- evaluating solutions, 522-524
- Go-Live, 631
- hardware management tools, 529-531
- HP OpenView VantagePoint for Windows, 526-527
- in-house applications, 521
- MMC (Microsoft Management Console), 536-538
- NetIQ AppManager for R/3, 527-528
- Note Assistant, 534-536
- operating system management tools, 531-532
- OS Collector, 512-513
- out-of-band management, 540
- Planning CD resources, 540-541
- potential problems, 520-521
- remote management tools, 539-540
- requirements, 519-520
- Solution Manager, 517, 532-534
- Solution Operations Services, 534
- System Management professionals, 313
- TCO (total cost of ownership) analysis, 153-154

T

- T-codes, 512**
- tape shipping, 190-191**

TCO (total cost of ownership) analysis, 123-124

acquisition costs, 124
backup/restore capabilities, 149
defined, 124-126
delta analysis, 126
disk subsystems, 140-141
distributed architecture, 146-147
documenting, 158-159
DR (disaster recovery), 130-133
HA (high availability), 128-130
heterogeneous system landscapes, 147-148
leasing versus purchasing, 153
maintenance costs, 152-153
manageability, 136
management tools, 149
operating systems, 141-142
operations, 136, 153-154
outsourcing, 126
people costs, 125, 151-152
performance delta analysis, 133-134
Planning CD resources, 159
recurring costs, 125
relational databases, 142-145
risk factors, 154-158
SAPGUI, 150
scalability, 134-135
security, 136
servers, 138-140
stacking, 147
standardization, 137-138
Systems Management, 153-154
TCO-driven system landscapes, 76-77
technology costs, 125
upgrades, 148-149

TCO Lease versus Purchase Model.xls file, 159**TCO SAP Outsourcing Analysis.xls file, 159****TCO Simple SAP Landscape Analysis.xls file, 159****TCO Solution Stack Considerations.xls file, 159****TechEd, 328-330****Technical Consultants**

certification, 337
obtaining training feedback from, 333-334

technical Go-Live, 624

batch housekeeping jobs, 626-627
Change Management package, 628-629
EarlyWatch services, 625
final system updates, 627-628
GoingLive Check, 625
SAPGUI deployment, 626
system lockdown, 628

technical phone screen, 284-285**Technical Recovery Teams, 209****Technical Sandbox, 52, 71, 378**

Technical Sandbox Change Management Checklist, 485
technical training, 315-317

Technical Support Organizations. *See TSOs***technology costs, 125****technology drivers, 44****technology perspectives, 63-65****Technology SMEs (Subject Matter Experts), 240****test organizers, 544****Test Week**

preparing for, 609-610
testing tasks, 610-613

Test Workbench, 561-562**Test/QA System, 52, 315****testing, 486**

availability of system, 471
business process testing, 543-544
BC-eCATT interface, 559
CATT, 544-546

- cautions, 564
- data tracking, 562
- eCATT, 544-546
- functional testing, 546, 549-551
- integration testing, 546, 550
- negative testing, 558
- number of test runs, 559
- per test scenarios, 547
- planning, 551-553
- Planning CD resources, 564
- post-execution tasks, 562-563
- process constraints, 558-559
- real-world example, 563
- regression testing, 546, 550-551
- “saptp - Compuware TestPartner and SAP eCATT” (white paper), 564-565
- SLA support, 563-564
- staffing issues, 557-558
- system landscape requirements, 556-557
- test organizers, 544
- Test Workbench, 561-562
- third-party tools, 553-554
 - when to use, 547-548
- Change Management, 471-472
- DR (disaster recovery) process, 213-214
- general new-product testing, 490
- importance of, 34
- standard test plans, 489
- stress testing, 543, 569-571
 - administrative scripts, 599-600
 - background noise, 608
 - backup/restore process, 616-617
 - batch processes, 581
 - business information warehouse, 588-589
 - business process scripts, 585, 591-592
 - client infrastructure, 596-599
 - component-layer testing, 579-580, 585-586, 590
 - for continuous improvement, 612-613
 - core script development, 592-596
 - costs, 570
 - data requirements, 581-582
 - databases, 576-577
 - disaster recovery plans, 617
 - disk subsystem, 576-577
 - eCATT, 588
 - ending test sessions, 605-606
 - Enterprise Portal, 590
 - excessive loads, 614-615
 - goals of, 571-572, 613
 - hardware delta stress tests, 606
 - monitoring, 611
 - network infrastructure testing, 577-579
 - online user analysis, 581
 - operating system utilities, 611-612
 - operating systems, 575-576
 - performance baselines, 572
 - Planning CD resources, 619
 - pre-tuning, 574
 - project plans, 580-583
 - ramping up users, 602-603
 - real-world examples, 617-619
 - SAP-aware testing tools, 583-585
 - SAP Standard Application Benchmarks, 586-587
 - scripting tips, 606-609
 - SE38, 588
 - server hardware, 575-576
 - SRM (Strategic Relationship Management), 590
 - statistics, 603-605
 - success criteria, 573-574

- system redundancy/failover, 614
- system-level stress testing, 574-575
- Test Week, 609-613
- utility scripts, 599-600
- virtual users, 583, 600-602
- what-if approach, 614
- Test/QA System, 52
- Testing specialists, 446-447
- volume testing, 544
- Testing specialists, 446-447**
- think time, 223, 227**
- third-party consultants, 33**
- Thrasher, 576**
- three-system landscapes, 14**
- three-tier architectures, 230-231, 255**
- throughput, 73**
- timelines**
 - planning, 51, 271
 - refining, 273
- times**
 - slash time, 271
 - think time, 223, 227
- tolerance for disaster, determining, 167**
 - costs of downtime, 169-170
 - return on investment calculations, 168-169
- tools. *See* software**
- Top Sessions reports, 532**
- total cost of ownership. *See* TCO analysis**
- tracking**
 - performance, 634-635
 - system availability, 623
- trainers, 312**
- training, 53, 71, 309-310, 315-318, 486**
 - benefits of, 310-313
 - Business Sandbox system, 72
 - certifications, 321, 336-337
 - Consultant Education, 321-322
 - custom training curriculum, 323-325
 - delta training, 310
 - Development System, 315, 317
 - Disaster Recovery System, 316
 - documentation, 330-332
 - e-Learning, 328
 - employee training opportunities, 303
 - Enterprise Portal, 326, 328
 - feedback and evaluation
 - certification programs, 336-337
 - Data Center Infrastructure team, 334
 - development consultants, 333
 - functional consultants, 333
 - Help Desk team, 335
 - Operations team, 335
 - Security team, 334
 - specialists, 335-336
 - technical consultants, 333-334
 - Help Desk staff, 454-455
 - IDES (Information Demonstration and Education System), 320
 - IFS (Information File Store), 319
 - InfoDB (Information Database), 319
 - internal staff members, 116-117
 - KW (Knowledge Warehouse), 325-326
 - onsite training workshops, 323
 - Planning CD resources, 337
 - Production System, 316
 - Project Team Training, 320-321
 - RWD Technologies, 328
 - SAP Academies, 321
 - SAP Solution Stack support, 318-319
 - SAP TechEd, 328-330
 - skillset backups, 313
 - Staging System, 315

- Technical Sandbox, 71, 315-317
Test/QA System, 315
Training System, 53, 71, 315-318
training/attendance certificates, 325
User Education, 322
user manuals, 330-332
when to implement, 313-314
- transactions, 611**
- transaction-based sizing, 232-233
 - transaction codes, 512
 - transaction monitors, 515-517
- transferring internal staff, 432-434**
- Transport Management System, 491-492**
- transport strategies, 415-416, 491-492**
- TRex Server, 257**
- triple mirroring, 182-184**
- trust relationship management, 416-417**
- Trusted PCs, 77**
- try-before-you-buy approach, 277**
- TSOs (Technical Support Organizations), 56-57, 93-94. *See also* training**
- best practices
 - flat organizations, 109
 - general teams, 110
 - high expectations, 110
 - promotion from within, 110
 - redundancy, 110
 - self-documentation, 111
 - staffing suggestions, 110-111
 - budget considerations, 119-120
 - consultants, 116, 434-435
 - budget considerations, 117-118
 - case studies, 118-119
 - compared to internal resources, 118-119
 - organizational balance, 118
 - Data Center Lead, 112
- DRO (Disaster Recovery Organization), 209-210
future service/support plans, 640-641
Go-Live, 635-637
Infrastructure/Basis Lead, 112, 114-115
IT job market, 97-100
limitations, 95-96
Planning CD resources, 120
PMs (project managers), 102-103
project team organization, 102
real-world examples, 103-104
 - all under one roof, 106-107
 - DBA cubbyhole, 105-106
 - Fortune 500 global TSO, 459-462
 - hardware versus SAP groups, 104-105
 - leveraging computer operations, 107-108
 - medium business TSO, 458
 - multiple production instances, 108-109
 - small business TSO, 456-457
- roles and responsibilities, 101
SAs (Solution Architects), 112-114
Senior DBA, 112
staffing, 275-277, 425-426
 - “bigger than me” approach, 437-438
 - Change Management Analysts, 496-497
 - Change Management Managers, 495-496
 - Component Basis specialists, 438-440
 - Component Functional specialists and programmers, 440
 - CSM (Candidate Status Matrix), 282-284
 - Disaster Recovery specialists, 445-446
 - EAI (Enterprise Application Integration) specialists, 440-441
 - employee new-hires, 296-298
 - employee retention, 151-152, 298-307
 - external consultants and contractors, 294-296

- "fewer is better" approach, 435-436
- Help Desk, 451-456
- High Availability specialists, 445-446
- internal transfers, 293-294
- interviews, 285-293
- job descriptions, 281
- Operations staff, 447-450
- Performance specialists, 441-443
- Planning CD resources, 307, 462
- probationary period approach, 277
- ranking prospective candidates, 289-291
- RDSSP (Rapid Deployment SAP Staffing Process), 278-281
- resume evaluation, 281-284
- resume-to-interview process, 277
- rewarding staff after Go-Live, 648-649
- Security specialists, 443-445
- SET (Sizing Evaluation Team), 240-241
- staff expansion, 430-432
- staff size, 427-428
- staff-testing approach, 277
- staffing holes, 426-427
- technical phone screen, 284-285
- Testing specialists, 446-447
- transferring internal staff, 432-434
- try-before-you-buy approach, 277
- support agreements, 641
- two-tier architectures, 256**

- U**
- UDC (Utility Data Center), 89-90
- unattended installations, 395-396
- Unicenter Application Management for R/3, 525-526
- Unicode, 388
- Uninterruptible Power Supply (UPS), 530
- Unisys SAP Competency Center, 224
- UNIX cluster approaches, 192-193
- unplanned growth, problems of, 74-75
- updating stress testing project plans, 582-583
- upgrades**
 - Change Management, 492-493
 - TCO analysis, 148-149
- UPS (Uninterruptible Power Supply), 530**
- User Education, 322**
- User Interface Deployment specialists, 312**
- user manuals, 330-332**
- users, 227-230, 256-258**
 - active users, 227
 - concurrent users, 227, 230, 253, 573
 - CRM Internet sales users, 258
 - customer interaction center users, 258
 - high users, 228
 - low users, 228
 - medium users, 228
 - mobile sales users, 258
 - named users, 227
 - normalized SD users, 230
 - online user analysis, 581
 - ramping up, 602-603
 - think time, 223, 227
- User Education, 322
- user manuals, 330-332
- virtual users, 583, 600-602
- utilities. *See software***
- Utility Data Center (UDC), 89-90**
- utility scripts, 599-600**

V

ValueSAP, 45, 382-383

variables

- LOGIN.COUNTER, 601
 - LOGIN.COUNTERTXT, 601
 - \$MACHINE, 600-602
 - MM03.INPUT, 594
- vertical scalability, 75**
- virtual storage, 246-248**
- virtual users, 583, 600-602**
- vision. *See* solution vision**
- vmstat tool, 532**
- volume testing, 544**

W

WAST (Web Application Stress Tool), 576

WBS (work breakdown structure), 481

Web AS (Application Server), 397-398

- sizing, 251
- Web AS experts, 311

Web sites

- AutoTester, 597
- Bart's Boot Disk, 385
- BC-eCATT interface, 559
- BMS Patrol, 525
- Bull SAP Competency Center, 223
- eCATT resources, 565
- eTestingLabs.com, 576
- Fujitsu-Siemens SAP Competency Center, 223
- HP SAP Competency Center, 223
- IBM SAP Competency Center, 224
- InstGuides, 389
- NetWeaver, 397

- PreciseSoft, 580
- QuickSizer, 232
- SAP, 564
- SAP Notes, 405
- SAP Service Marketplace, 401
- SilkPerformer, 585
- Solution Manager Learning Maps, 384
- Sun SAP Competency Center, 224
- Unicode, 388
- Unisys SAP Competency Center, 224
- user manuals and documentation, 331

weekly checklists, 376-377

WGate, 404

what-if approach to stress testing, 614

Windows Performance Monitor (PerfMon), 531

wizards, Setup Manager Wizard, 385

work breakdown structure (WBS), 481

work processes, 231

workgroup servers, 257

workshops, 323

X-Z

XI (Exchange Infrastructure), 414

- IB (Integration Builder), 398
- installation, 399-400
- IS (Integration Server), 398
- MarketSet Adapter, 399
- SAP XI Add-On, 399
- SAP XI Connectivity, 399
- sizing, 252-253
- SLD (System Landscape Directory), 399

zoning, 372