

SQL Server. Part 1

***A Practical Guide to Backup,
Recovery & Troubleshooting***

eBook

© Copyright Quest® Software, Inc. 2005. All rights reserved.

The information in this publication is furnished for information use only, does not constitute a commitment from Quest Software Inc. of any features or functions discussed and is subject to change without notice. Quest Software, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in this publication.

Last revised: October 2005

TABLE OF CONTENTS

PLANNING AND PREPARATION.....	6
WHAT IS DISASTER RECOVERY PLANNING?	6
<i>Disaster Recovery Plans (DRP)</i>	7
<i>DRP for SQL Server Databases.....</i>	9
<i>Example – Disaster Recovery Planning</i>	11
FRAMEWORKS FOR IT SERVICE MANAGEMENT	16
<i>CoBIT (Control Objectives for Best IT Practices)</i>	17
<i>ITIL (Information Infrastructure Library)</i>	17
<i>Microsoft Enterprise Services Framework (ESF)</i>	19
<i>Balanced Scorecard</i>	21
SERVICE LEVEL METRICS	21
<i>The Scale of Nines</i>	21
<i>Other Availability Metrics</i>	22
<i>What Is Achievable with SQL Server?.....</i>	23
RESPONSIBILITY VS. ACCOUNTABILITY	23
BUILDING STAFF CAPABILITY	24
<i>Consider an Emergency Response Team (ERT)</i>	24
<i>DBA Training and Skills (Building ERT Expertise).....</i>	25
CHANGE CONTROL.....	28
MANAGING CHANGE CONTROL BY EXAMPLE	28
<i>Environment Overview</i>	29
<i>Pre-Change Window Resource Meeting</i>	31
<i>Visual Source Safe (VSS)</i>	32
<i>Managing Servers</i>	33
<i>Development Server.....</i>	34
<i>Test Server.....</i>	36
<i>Production Support.....</i>	37
<i>Production</i>	37
<i>Hot Fixes</i>	39
<i>MRAC of IR/Task Completion</i>	40
<i>Summary</i>	41
USING VSS BY EXAMPLE – PART 1	42
<i>Terminology.....</i>	42
<i>Build Phase – Moving to the New Change Control Structure.....</i>	43
<i>First Change Control</i>	44
<i>Moving Code to TEST</i>	47
<i>Overwriting existing code in TEST from DEV.....</i>	48
<i>Taking a Change Control into Production.....</i>	49
USING VSS BY EXAMPLE – PART 2	51
<i>How Do I Move Files to Next Weeks Change Control?</i>	51
<i>What Does VSS Look Like After 2 Iterations of Change?</i>	52
<i>I Forgot to Take a File into Production for a Schedule Change.....</i>	53
<i>I Have a Special Project That Will Span Many Weeks, Now What?</i>	53
USING VSS BY EXAMPLE – PART 3	56
<i>Re-Iterating our Chosen VSS Structure</i>	56
<i>What Does My VSS Look Like to Date?.....</i>	57
<i>What Do You Do with /Development After Each Change Control?</i>	58

VSS Issues.....	58
Welcome to .Net.....	63
VS.Net Solutions and Projects	64
VSS FOR THE DBA	68
THEORY AND ESSENTIAL SCRIPTS.....	69
UNDO & REDO MANAGEMENT ARCHITECTURE.....	69
AUDIT THE SQL SERVER INSTANCE	75
META DATA FUNCTIONS	77
LISTING SQL SERVER INSTANCES	78
INFORMATION SCHEMA VIEWS	78
DATABASE, FILE AND FILE GROUP INFORMATION.....	80
<i>Extracting Basic Database Information</i>	80
<i>Determining Database Status Programmatically</i>	80
USING O/ISQL.....	81
RETRIEVING LICENSING INFORMATION	82
<i>Alter Licensing Mode After Install?</i>	83
ALLOWING WRITES TO SYSTEM TABLES.....	84
COUNT ROWS & OBJECT SPACE USAGE	84
BLACK BOX TRACING	86
SCAN ERROR LOG FOR MESSAGES?.....	88
DATABASE LAST RESTORED AND FROM WHERE?.....	89
WHAT STORED PROCEDURES WILL FIRE WHEN MY INSTANCE STARTS?	90
WHEN THE DATABASE WAS LAST ACCESSED?	90
ESSENTIAL TRACE FLAGS FOR RECOVERY & DEBUGGING	91
<i>Example of Setting and Verifying the Trace Flags</i>	92
“TRACE OPTION(S) NOT ENABLED FOR THIS CONNECTION”?	94
BULK COPY OUT ALL TABLE DATA FROM DATABASE	94
SQLSERVR BINARY COMMAND LINE OPTIONS.....	95
SQL SERVER LOG FILES	95
<i>How and When Do I Switch SQL Server Logs?</i>	97
DETECTING AND DEALING WITH DEADLOCKS	97
ORPHANED LOGINS	101
ORPHANED SESSIONS – PART 1	102
ORPHANED SESSIONS – PART 2	103
CHANGE DB OWNER	103
TRANSFER DIAGRAMS BETWEEN DATABASES	104
TRANSFER LOGINS BETWEEN SERVERS.....	105
KILLING SESSIONS	105
<i>The ALTER Statement</i>	106
<i>How Do I Trace the Session Before Killing It?</i>	106
SETTING UP AND SENDING SQL ALERTS VIA SMTP.....	108
<i>The Stored Procedure</i>	110
<i>Creating the Alert</i>	110
<i>Testing the Alert</i>	112
<i>Recommended Backup and Restore Alerts</i>	113

HIGH AVAILABILITY.....	114
PURCHASING THE HARDWARE	114
<i>So What Hardware Should I Buy?.....</i>	114
<i>What is the HCL or the "Windows Catalog".....</i>	117
HIGH AVAILABILITY USING CLUSTERS.....	118
VMWARE SQL Cluster – <i>by Example</i>	119
<i>Using VMWARE in Production?.....</i>	120
<i>Step 1. Software & Licensing</i>	120
<i>Step 2. Create the Virtual Servers.....</i>	121
<i>Step 3. Build Your Domain Controller.....</i>	123
<i>Step 4. Build member server 1 (node 1 of the cluster).....</i>	126
<i>Step 5. Build Member Server 2</i>	133
<i>Step 6. Install SQL Server 2k in the Cluster (Active/Passive)</i>	134
HIGH AVAILABILITY USING LOG SHIPPING	142
<i>Manual Log Shipping – Basic Example</i>	144
<i>Custom Logshipping – Enterprise Example</i>	148
INDEX	155

PLANNING AND PREPARATION

“..how you develop [it] is at least as important as the final result”

A.M.Schneiderman

The role of DBA is undoubtedly an important one, but many DBAs tend to be somewhat blasé about backup and recovery. This e-book attempts to bridge the knowledge gap and provide working scenarios, policy/procedure and best practice for database backup and recovery.

As with my previous e-book, I assume a reasonable knowledge of DBMS architecture and in particular some general DBA experience with SQL Server.

This first chapter covers a range of planning and preparation strategies that will ultimately define your system design, backups and recovery procedures. We will focus on disaster recovery planning and frameworks for IT service management, then take this further in chapter two and change management (by example) then chapter three with alternatives for high availability.

What is Disaster Recovery Planning?

This is a complex question. It conjures thoughts of business continuity, data and database recovery, network and server availability, staff expertise/training/availability and policy and procedures. So the question is probably not so much “*what is disaster recovery?*” (the title tends to be self explanatory), but “*at what point do you draw the line?*”, and how much time and money are you prepared to spend curbing the multitude of possibilities?

That said; let us define *disaster recovery planning*.

Planning for Disaster Recovery is synonymous with *contingency planning*; it is “a plan for backup procedures, emergency response and post-disaster recovery”. This plan encompasses the who/how/where/when of “emergency response, back operations and post-disaster” procedures to “ensure the availability of critical resources and to facilitate the continuity of” business “operations in an emergency situation” (2).

It is very interesting reading the variety of thoughts in this space (3), one particularly interesting one was the division of the DR from that of business continuity planning:

1. Disaster Recovery (DR)—the *process of restoring* systems [including manual and automated business process] to an operational [state] after a catastrophic systems failure leading to a complete loss of operational capability. (3a)
2. Business Continuity (BC)—is the *forethought to prevent loss* of operational capability even though there may be a catastrophic failure in some parts of the system. BC includes DR plans to restore failed system components. (3a)

Here the two key elements are *forethought to prevention* and the *process of recovery/resumption of business*—both are essential components partners in *building, maintaining and sustaining* capability at a technical and business services level (we understand the risks, decrease the risks, and manage the risks). Only at this point can we, through a fine balance of money and persistent capability, be confident in our ongoing DR planning (DRP).

Disaster Recovery Plans (DRP)

Disaster recovery is divided into two distinct processes:

1. IT recovery planning or IT system recovery planning
2. business continuity planning – business and IT risk assessment, mitigation planning, both manual, automated, physical and logical. This is a overarching feeder to a) in terms of where a bulk of the focus will be for IT disaster plans, based upon known *business imperatives* (i.e. we only doing what is relevant to the business and its overarching strategy).

For simplicity sake, we will use the acronym DRP to encompass 1). Although important, 2) will not be covered any further in this eBook.

The focus of DRP is in the “recovery of the essential [IT] functions and [IT] systems of an organization”, and “emphasizes recovery within the shortest possible time” (51). The DRP provides the *road-map* that details the actions performed before, during and after a disaster. The plan(s) are a comprehensive set of statements that address any IT disaster that could damage the business and its [core business] services. (51)

The process of planning is an iterative one, base upon:

1. efficiency : doing the right thing at the right time, before, during and after a disaster (speed, sustainability and thoroughness)
2. effectiveness : cost-effective resumption and business service recovery, effective coordination of recovery (cost, coordination, end-game achieved)
3. [professional, legal] obligations, [formal] contractual commitments and accountabilities

In order to effectively write and measure IT performance against SLA's; or underpinning contracts with external providers and operational level agreements (between your IT sections, i.e. comms, prod support, in-house developers, help desk etc), the DRP is a fundamental driver for defining the contracts and outlining areas of concern. The commitment to high quality through legally (and financially) bound commitment ensures efforts are made to honor them. (51)

The DRP documentation is context based, typically in one of three strategic views: (51)

1. Mitigation

What measures are in place to minimize the impact of identified disaster scenarios.

2. Anticipation

What are the measures to respond to and recovery from a disaster.

3. Preventative

What measures are in place to prevent disasters from happening? This includes problem and known error management.

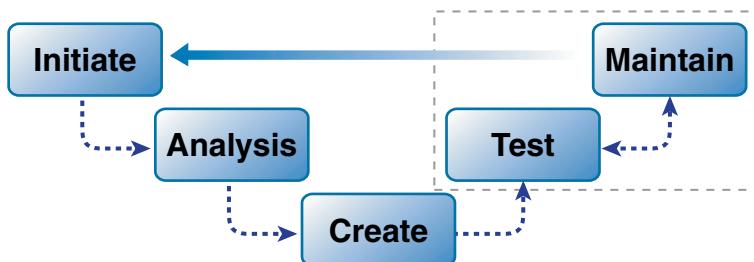
Baselining your existing environment to secure and manage business and IT dependencies in which to forge prevention strategies.

Our disaster plan “table of contents”, its ownership and iterative management is very much based on the document’s *strategic view*. Be aware that DRP documents need to be:

- Prescriptive;
- Simple to follow; and
- Fact orientated.

to be an effective *working* document. This may require professional (third party) assistance.

The process of DRP definition can be broken down as: (52)



Initiate – form team, identify scope, resources required, timeline, stakeholders, budget and management buy-in. Link the DRP’s statement of work to existing initiatives and most importantly to the business continuity requirements. The identification of overarching strategic view is required.



The DBAs may need to drive the process if nothing is currently in place or is not actively pursued by management. The process of drafting the first DRP document may be the process initiator.

Analysis – requirements gathering, scope item drill through, build activity list, pass through concerns and highlight issues, priorities and set deliverables.

Create – create plan, document all processes/procedures/steps required to meet stated scope and activities from the analysis and initiation phases.

Test (iterative) – evaluate and walkthrough DRP, of key importance is the relevancy or effectiveness for the stated objective. The planned should be appropriate for the reader, simple to implement and care taken with the workflow of steps, identifying contact points and highlighting weaknesses as early as possible. Test plans are retained for audit, and to assist with managed re-iteration.

Maintain (iterative) – ongoing plan maintenance, review and active ownership; measures should be applied at this stage to ensure the plan is persisted. Assign group responsibility and accountability.

Avoid merging DRP documents into one or two overarching papers for the entire IT shop, or for all databases. I highly recommend dividing them based on the context in which it is written. The plans should be key part of the Intranet website (secured appropriately of course), versioned and burnt to CD. There is little point in planning unless the results are documented and made available.

DRP for SQL Server Databases

Depending on the site the DBA may be more or less prescriptive with the steps taken to restore a service (i.e. list *all* steps for restoring the master db for example), this may be based on in-house expertise, team size and site attendance.

Based on our discussion of strategy and the iterative approach to DRP definition, the DBA needs to consider:

- Impact on system users if the database is not available
 - The DBA needs to understand the user and business impacts of the database not being available, both for read/write and read-only. You need to be somewhat pragmatic with the choices made to keep the system available, and be sure the business users are heavily involved; a 24x7 system with 1hr maximum downtime may consider a one day loss of data as an acceptable compromise for example; a online shopping system may not.
 - Communication plan – who is contacted and when? how? apart from your technical staff does the communication plan encompass business users?
- Storage of installation disks, associated physical media and license keys
 - Physical access (lockdown and security procedures) and storage of SQL Server installation media, especially license keys
 - Third party utilities must be considered
 - OS installation disks and license keys
 - How the change management process ensures media is updated (in the right areas and in a timely fashine)

- Restoration
 - Can we restore in time to meet SLA's? what can we do to achieve them? The cost of trying to do so (even at the risk of human error due to complexities involved) and do we need to revisit the SLA?
 - Recovery scenarios (server, full instance and binaries, databases, tables, replication, full text indexes, OLAP cubes)
 - User account/login details (consider essential OS and domain level logins, DBMS service accounts, dialup procedures and access rights and software required etc)
 - Access to backup tapes, time to restore and responsibilities, re-call of tape procedure (the cost and signatories to receive media)
 - Processes for dealing with corrupt backups, missing tapes (or overwritten ones) and/or database files
 - Process for system database restoration
 - Checkpoints before recovery will begin
- Staff capability and availability
 - Key staff contact and escalation list, phone numbers (and physical phones - never use private phones for business work)
 - Microsoft Support contact numbers and keys/credit information
 - New training requirements based on proposed HW/SW selection or base capability to date
 - External vendor support and underpinning contracts for DBA expertise
 - Reference books/manuals and how-to's
 - Dialup/remote access procedures (includes after hours system access, taxi/resource expense claims, minimum hardware required for remote access, what staff "cant" do whilst on call).
- Inter-system dependencies and distributed transactions or replicas
 - Dependencies to other business components, such as web servers, middle tier, distributed systems, especially where transactions span time and service.
 - Startup order, processing sequence of events (i.e. data resend or replication re-publishing steps etc). This is very important in sites using replication where the time taken to restore such services and re-push or pull replicas may be significant.
- Backups
 - Backup file access and file retention periods, both tape and on-disk
 - Speed of accessing backups, simplicity of? Require others to intervene?
 - Full backup cycle, including regular system database backups
 - Verify backups. Test restore procedures and measure to ensure this is occurring
 - Combinations of Full, Differential and Log backups
 - Log shipping – including the security and compression/encryption of log shipped files and the complexity this may bring
 - System dependencies that will affect the backup, namely the Windows registry, OS binaries (system state) etc.

- Database hooks, associated modules and links
 - Installed extended stored procedures
 - Database links and their usage/security/properties
 - Full text catalogs
- Hardware and Software Spares
- Change Management Procedures
- Audit of existing environment
- Fail-safe tasks – I recommend the following at a minimum
 - Full db script monthly
 - Check for DB corruption daily (when possible – dbcc checkdb)
 - SQL Diagnostic (sqldiag.exe) dump daily to assist with debugging major system crashes
 - Maintenance of global instance and database parameters (configuration and initialization settings, trace flags, startup stored procedures, database properties and file locations etc)

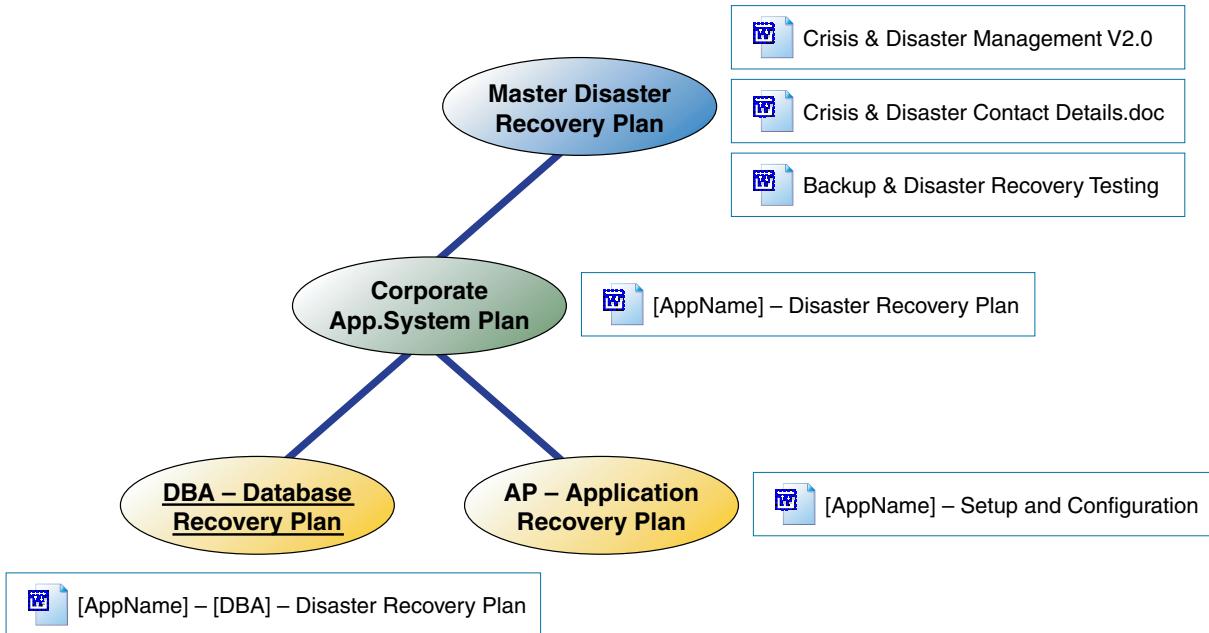
With these and others, the DBA is well on track to provide a solid framework in which to base DRP; that being *database and associated service recovery* in a *efficient* and *effective* manner.

Example – Disaster Recovery Planning

It will be rare that your existing company has no DR plans in some form that you can leverage and build upon—so my advise is *go and look for it*. It is important that you blend in with the initiatives of other team members and that of management to gain support in the time you will spend writing and maintaining them (which can be significant).

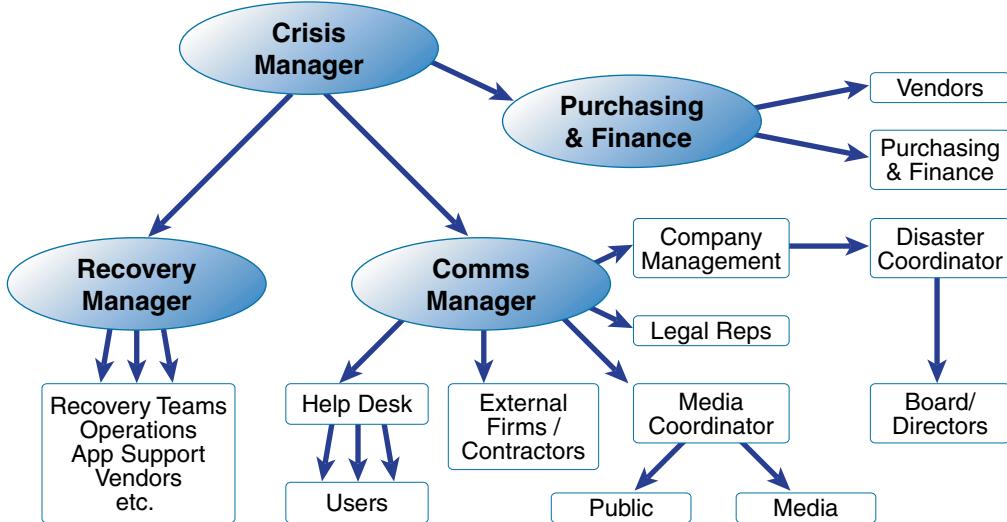
The DR documentation may be similar to the organizational chart, where all management and executive members are signatories to, and part of the communication of *service based recovery plans*. I say *service based* in that the databases you manage from day to day support core applications and delivery critical services to the business in which *you* are a part of.

The diagram below provides an example of the DR documentation produced for an organization, and its context in terms of the services it applies to:



1. “Crisis and Disaster Management” document contents

- a) Provides general guidance for management of a disaster.
- b) Defines the roles and responsibilities associated with the management of a crisis. It is not unusual to have three core roles—*crisis manager*, *communications manager*, *recovery manager* and the *finance/purchasing manager*.



- c) Communication initiation “sequence of events” and associated flow-charts, including:
 - Initial Notification
 - Ongoing Updates
 - Communication Method(s)
 - Logging of communications and time based events
 - Communication Milestones (dependencies)—this will include the *when*, *responsibility* and *action* information.
- d) Reference to the “Crisis and Disaster Contact Details” document and its use and maintenance.
- e) Crisis Management section that defines
 - When a crisis is declared (pre-conditions)
 - Crisis Management coordination process(es)—includes the use of a central communications center, crisis meetings, record keeping, involvement of the business, crisis closure
 - Disaster Management coordination process(es) —escalation to a disaster (triggers), disaster management center (where?, staffing?, point of contact?, notifications, record keeping, requests for resources, public/media questions)
 - Service Restoration Priority List—list of core services (i.e. IT applications/services)
 - Release of Funds
 - Templates for:
 - a. Crisis and Disaster Review Meetings
 - b. Actions Lists (notifications/escalations)
 - c. Contact Lists
 - d. Sample email and SMS messages
 - e. Broadcast messages
 - f. Activity Logs
 - g. Crisis and Disaster Declaration Memo's

2. “Crisis and Disaster Contact List” document contents

- a) Simple tabular form divided into section, representative of the contacts “role”, for example *management*, *external partners*, *help desk/others*
- b) Includes confidentiality notices and reference to the above plan on its use

3. “Backup and Disaster Recovery Testing” document contents

- a) This document is an overarching *statement of work* that spells out the process of backup and disaster recovery testing. The document lays out the ground rules and what is expected by all parties identified as responsible and accountable for the application. The document will include schedules for annual/monthly tests, including signoff forms and registers.
- b) Specifically, the document should include the following content items:
 - Definition of Application class levels and their frequency of disaster recovery testing
 - List of core applications and their *business priority* and the *class* of application
 - Description of the disaster recovery testing cycle (iterative cycle consisting of *walk-through*, *simulation testing*, *parallel testing* and *full interpretation testing* (running prod on DR machines).

TEST	COST	DISRUPTION	COMPLEXITY	EFFECTIVENESS
Walk Through	None	None	None	Minimal
Simulation	Cheap	None	Minor	Effective
Parallel	Moderate	None	Moderate	Very Effective
Full-interruption	Expensive	Likely	Major	Extremely Effective

- Detailed summary and flow charts of the implementation of a disaster recovery test. Including references to template documentation, staff contacts etc.
- Backup test procedures—basic summary of what a backup is, what should be considered when testing a backup.

4. “[AppName] – Disaster Recovery Plan” document contents

- a) This document is based upon a template from the master recovery plan and is used for each and every core service. Typical content items include:
 - Introduction, scope and audience
 - Recovery strategy – including a summary of the availability times (from SLA if you have one), invocation (who is authorized to), guidance as to *how* to initiate and control the situation, system dependencies (system, doc reference, contact), recovery team (name, title, contact), how to request additional resources, recovery team checklist (ie. confirmed invocation?, established recovery team?, arranged for backup/SW media? etc..)
 - Recovery procedure – Infrastructure (locations, media), Data Recovery (OS, file systems, databases), Application Recovery (interfaces, user interface), Assumptions/Risks, Referring low level documentation (see next).

5. “[AppName] – Setup and Configuration” document contents

- a) This document provides step by step instructions for complete reinstallation of all components making up the service. The document should NOT refer to lengthly installation sheets from vendors where possible, but if so, it needs to be well clarified with environment specific notes.
- b) In writing the document, consider color coding based on responsibility (sys admin, DBA, help desk) and function (web server, database, middle tier etc). The document should be executed serially where possible.
- c) Based on the Master Recovery Plan, this document will include the communication and escalation paths in a flow chart format. Contact details are also included which are specific to the application being delivered.

6. “[AppName] – [DBA] – Disaster Recovery Plan” document contents

- a) Your specific database backup plan may be part of a wider (enterprise) based backup and restore strategy for all corporate databases. A classic example is backups driven via centrally managed backup software, where the process for restore is the same no matter the underling DBMS (to some degree).
- b) The expertise of DBA's may dictate the prescriptive nature of this document; for example, will you describe at length the process for restoring the system databases? Consider this when developing the contents, for example:
 - Backup schedule
 - a. Types of backups taken and their retention periods, for example, you may do a nightly full, but also mention a full monthly that is retained for 6 months.
 - b. Backups start when and normally run for how long? (include the reasoning behind the times chosen and if any other job may impact the schedule)
 - c. Special conditions apply? i.e. is there any reason why the schedule will differ for a specific part of the month or year?
 - d. Standard backup file retention period
 - e. How the backups are performed (written to disk first then taken to tape? log shipped? Etc)
 - Backup Maintenance
 - a. Monitoring of backups
 - b. Archival of older backups
 - c. Backup testing
 - d. Assumptions and risks of testing

- Recovery process
 - a. Initiation and Communication Procedures (is a timely reminder of the overarching process that must be followed)
 - b. Media and Passwords
 - c. Requesting File Restoration
 - d. Database and Instance Configuration
 - i. Server level properties (applicable to the DBMS)
 - ii. Instance level properties
 - iii. Database level properties
 - iv. Replication Configuration
 - v. Full Text Indexing
 - vi. Logins and Users (including their database security properties)
 - vii. DTS Packages and their Job schedules
 - e. Order of Recovery
 - f. System Interface Recovery Procedures
 - g. Database Recovery
 - i. Pre-Conditions (recovery of media etc)
 - ii. Recovery Scenarios (may include a wide variety of scenarios, from system databases, suspect databases, lost DTS jobs, moving to the DR server etc).
 - iii. Post-Conditions (including steps to be taken if partial recovery was only possible)

Frameworks for IT Service Management

What I discuss throughout this book is very much technical; covering the how and why of backup and recovery at the DBMS. At a higher level, this should simply be part of a Corporate and IT Governance framework that actively translates business objectives to IT objectives through a common language, roles/responsibilities, accountabilities, and help drive the same strategic goals for the business.

This section will provide a very short summary of frameworks in relation to *IT service management* processes.

I cannot stress enough the importance of such frameworks within your organization. Later in this book I discuss a customized version of *change management* for a small applications development group, but many governance models take this much further in terms of a complete solution for service management and delivery.

CoBIT (Control Objectives for Best IT Practices)

CoBIT is an open standard outlining good practice for the management of IT processes, and most of which is free to download (www.isaca.org). The “ISACA and its affiliated IT Governance Institute lead the information technology control community and serve its practitioners by providing the elements needed by IT professionals in an ever-changing worldwide environment.” (4)

The key items of focus in terms of DR are found under the *Delivery and Support* control objective (aka Domain):

1. Manage Changes – outlines the process of change, requests for, SW release policies, maintenance and documentation etc, all essential components that can assist in further DR planning.
2. Ensure Continuous Service – the establishment of a continuity framework with business process owners.
3. Manage third party services – providers of third party services are controlled, documented and interfaces managed. This encompasses continuity of service risks.
4. Educate and train users
5. Manage problems and incidents
6. Ensure system security

The reader should download and read the Framework documentation related to *Delivery and Support* as we have only touched a small number of processes.



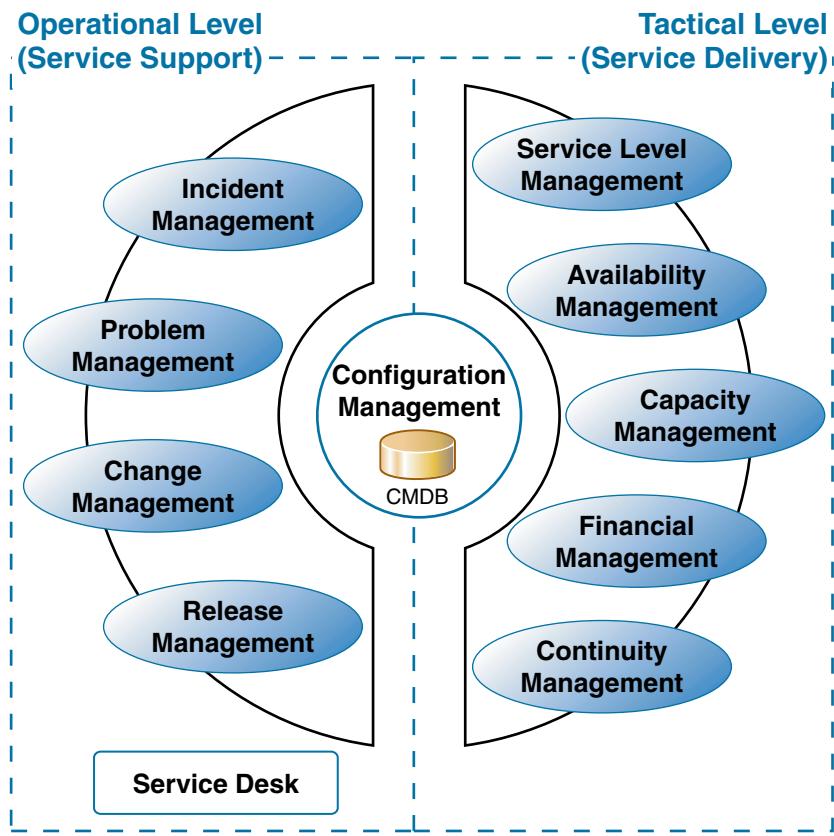
Many (if not all) of the COBIT processes map 1:1 to the ITIL framework discussed next.

ITIL (Information Infrastructure Library)

ITIL has developed into a defacto world standard for IT Service Management from its beginnings in the late 1980's by the British CCTA (Central Computer & Telecommunications Agency—now called the Office of Government Commerce). From its original library of some 32 books, the latest revision (2000/2001) sees a complete restructure down to two core condensed volumes, concerning itself with the delivery and support of IT services appropriate to the requirements of the business.

The framework itself provides proven methods for the planning of common processes, roles and activities and their inter-relationships (communication lines). More importantly, the framework (like others) is goal orientated around modules, so each process can be used on its own or part of a larger model.

In terms of DR within the ITIL framework (www.itil.co.uk), both the service delivery and service support sides provide complementary processes for the delivery of IT services to the business:



Some of these are:

1. Availability Management – deals with and guarantees the demands of systems availability. It is focused on the reliability and flexibility of operational systems, not only from internal staff with hard/soft problems, but includes contractual stipulations by suppliers.
2. IT Service Continuity – also known as contingency planning manages all information required to recover an IT service. It notes the importance of a business and IT focuses to continuity management and that both part of the same fabric. Through risk analysis, assessment and measurement, it will stipulate the how and when of the recovery cycle in terms of real business need.
3. Configuration Management – register of configurable items and their relationships (not just an IT asset register) within a database known as the CMDB. This provides the fundamental basis for other processes, with the registration of not only hardware and software, but SLA's, known errors and incidents etc. The key is relationship management to drive corporate repository of knowledge to assist all key ITIL processes in some form.

4. Release Management – manages the planned and applied changes to components in the IT infrastructure.
5. Problem Management – deals with the root causes of disruptions to IT services. The process attempts to distinguish, recognize, research and rectify issues with the aim to minimizing recurrence of such problems.
6. Incident Management – first line support processes/applications in place for customers where they experience problems with IT services.
7. Change Management – is accountable for all service changes within the IT environment. It is driven via a formal set of steps and management processes to manage change and coordinate the deployment of change with release management; updating the CMDB along the way. The change management processes are driven via RFC's (requests for change) and typically a CAB meeting to track, accept and evaluate proposed changes. Forward schedules of change clearly define the proposed release dates and keep all parties well informed.

All of the processes naturally encompass the daily working practices of any IT shop, no matter the size, and ITIL can be effectively adapted from the guiding principles it provides. Remember this is a service management system of core processes to assist in aligning IT to the business, and it is far from being a *technical or prescriptive how-to* for IT management.

Microsoft Enterprise Services Framework (ESF)

There are three components to the cyclic ESF framework:

1. Prepare – Microsoft Readiness Framework (MRF)
2. Plan and Build – Microsoft Solutions Framework (MSF)
3. Manage – Microsoft Operations Framework (MOF)

The MOF part of ESF can be regarded as an extension to ITIL in terms of distributed IT environments and the latest IT trends (web transactional based systems etc). In practice, it is a Microsoft implementation around Microsoft technologies.

The MOF has two core elements:

1. Team Model – describes in detail how to structure operations teams, the key activities, tasks and skills of each of the role functions, and what guiding principles top uphold in running a Microsoft platform.
2. Process Model – Is based around four basic concepts
 - a) IT service management has a life cycle
 - b) The cycle has logical phases that run concurrently
 - c) Operational reviews are release and time based
 - d) IT service management touches all aspects of the enterprise

With that in mind, the process model has four integrated phases: *changing*, *operating*, *supporting*, *optimizing*, and all forming a spiral life cycle. Each phase follows with a review milestone tailored to measure the effectiveness of proceeding phases.

The diagram below shows the high level roles and four process phases within MOF:

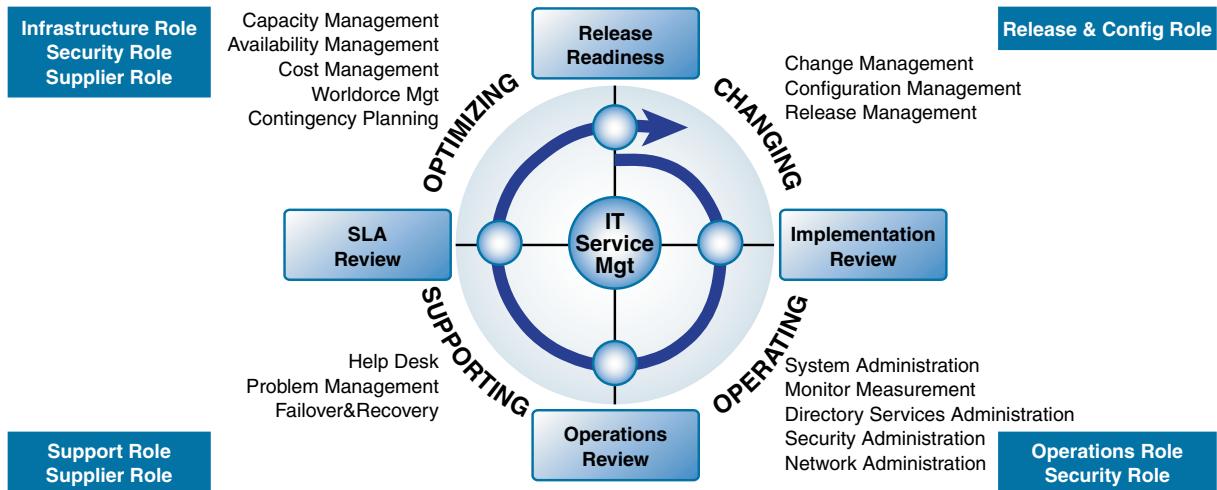


Illustration: MOF Process Model and Team Model, Roles and Process Phases,

<http://www.serview.de/content/english/itsm/4mof/7abbildung/view>

Balanced Scorecard

Although not a service delivery framework, the balanced scorecard (www.balancedscorecard.org) is a measurement-based management approach that works on the idea that “all business processes should be part of a measurement system with feedback loops”. (12) The system takes into consideration both strategic and technical plans that are deployed along with a measurement system; more importantly, this is a continuous cycle that is “aimed at continuous improvement and continuous adjustment of strategy to new business conditions” (12).

It should be noted that the balanced scorecard is a corporate “strategy into action” tool rather than an IT Governance framework (please be aware that I have used “strategy into action” very loosely, it is much more than simply this). Such a strategy is a key element in which governance models can reside and is well worth understanding at a high level.

Service Level Metrics

In order to improve you need to something to measure against. Simply we know, but rarely done in many fields of business, including IT. This section covers the *scale of nines* as one of the many forms of measurement, primarily used in service level agreements to define a level of *system availability* in its most simplistic form.

The Scale of Nines

Many service level availability measures talk in terms of the “nines”, as the table below shows: The five and six nines are a formidable and complex requirement for equipment vendors to deliver (and therefore they ask a premium price), and even more so within your organization. There are no hard and fast rules simply to say the higher the nines the more costly the solution, both in raw capital and operationally. Generally speaking, select the rating that represents the comfort level the organization (culturally, economically, politically)—be reminded that it is a *predictive measure* and not a hard and fast rule as its mathematical calculation in even the simplest environment is absurdly complicated. (38)

PERCENTAGE	DOWNTIME/YEAR
100	None.
99.9999	<= 30 secs
99.999	<= 5.2m
99.99	<= 52.22m
99.9	<= 8hrs 46m
99.0	<= 87hrs 36m
90.0	<= 36days 12hrs

For the SLA, the measurement is outside the scope of *scheduled downtime* or commonly known as a *change window*. The window must be relatively flexible in terms of its size to encompass large changes, but a fixed maximum period may be unavoidable. In order for this to occur, management must revisit current change practices, underpinning contracts with external suppliers and operational (internal) contracts between business entities.

Other Availability Metrics

Who you are establishing a service level with will ultimately determine the technical detail, or lack of, in terms of the availability measures we use.

For example, if we are establishing service levels between an IT shop and an external data carrier for your privately connected data centers, then availability metrics based upon packet loss, allocation errors, minimum transfer speeds, are effective measures. If the IT shop is dealing with the application owners using the data center services, then these metrics mean (and measure) nothing of any value and will not be able to be related to the overall experience of service.

Generally speaking there is no one definitive measure that can be applied to any one service, client or business. The choice of a metric depends upon the nature of the service and the technical sophistication of the users. (39).

All said, there are some common formulas and definitions, the value of which is tough to crack and more importantly, must be justified and documented if the final figures are challenged. They are:

$$\% \text{Availability} = (\text{MTBF} / (\text{MTBF} + \text{MTTR})) * 100$$

- OR -

$$\% \text{Availability} = (\text{Units_of_time} - \text{downtime}) / \text{total_units_of_time}$$

- WHERE -

MTBF = mean time between failures

MTTR = mean time to recover

Be aware that availability metrics are very different to *latency / response time* metrics, or *throughput* or *capacity* metrics, let alone *utilization metrics*. Be warned that these types of metrics require the utmost confidence in terms of measurement and how they pan out for ALL users of the service.

Remember that availability is purely “how much time a system is available for normal operation”, be that a fixed figure or one that varies throughout the SLA term. Where possible, utilize past statistics, trial availability measures and orientate the SLA to whom you are dealing with.



Without effective IT service measures in place, never attempt a charge back model.

What Is Achievable with SQL Server?

In order to achieve anything more than 99.99% uptime outside of a standard change window for the DBMS, the DBA may should consider the following:

PERCENTAGE	DOWNTIME/YEAR	CONSIDERATIONS
100	None.	Impossible. Possible in a perfect world with no change.
99.9999	<= 30 secs	Improbable. In a highly redundant environment using Microsoft clustering or a third party product the system will always fault for a short period of time (in the order of 30sec to one minute). The key here is the services around the DBMS and eliminating all single points of failure. Change management and access to the servers is critically important. Such a system cannot test "live failovers" unless it is part of the change widow, but is extremely risky.
99.999	<= 5.2m	Possible. With multiple redundant services in play. There is not time for inline hardware replacement during an emergency.
99.99	<= 52.22m	Possible and easily sustained (at a price\$). Hardware spares must be easily available if we choose not to cover all single points of failure. Reboots of hardware are not possible in most cases.

Responsibility vs. Accountability

When establishing any plan it is of utmost importance to not only define a persons "role(s)", but clearly identify it in terms of *responsibility and accountability*.

A responsibility is a basic requirement when performing an activity, i.e. when there is something we are required to do (13). For example, a DBA is responsible for making and validating backups of the database to facilitate complete recovery to a point in time.

An accountability stems from actions we (or others we are managing as a Senior DBA) do or don't take, regardless of the whether they are our direct responsibility or not. Simply put, we are answerable for actions and their results. (13) Clearly identifying them is difficult, but extremely important as a measure of service delivery and professionalism.

One interesting flow on topic is that of *authority*, "if you make someone responsible but do not give them authority, can they be held accountable"? (14). If you look at this pragmatically the point of writing down, and agreeing upon tasks that are in effect *promises* tends to require a level of authority for these tasks. Management need to ensure this remains in focus and does not fragment into the realms of *shared accountability* to a point where being accountable is no different from having responsibility.

It is critically important for business continuity, disaster planning and systems recovery that time is taken to identify responsibilities and those accountable for the actions taken.

The assignment of accountability and responsibility builds upon the fabric of process, procedure and control as without it, no one will be answerable to actions taken (or not as it seems in many cases) and can be a disastrous situation for the business.



A classic case from ITIL with the terminology applied is that of change and release management—*the change manager is accountable for all changes within the IT environment, the release manager is responsible for making and communicating the change. At all points in times the final outcome of the change in production rests with the change manager.*

Having a shared, global understanding of these terms is an important step forward in the allocation and *measurement* of work that forms the hygiene factors of any business. This is especially the case when dealing with disaster and change management.

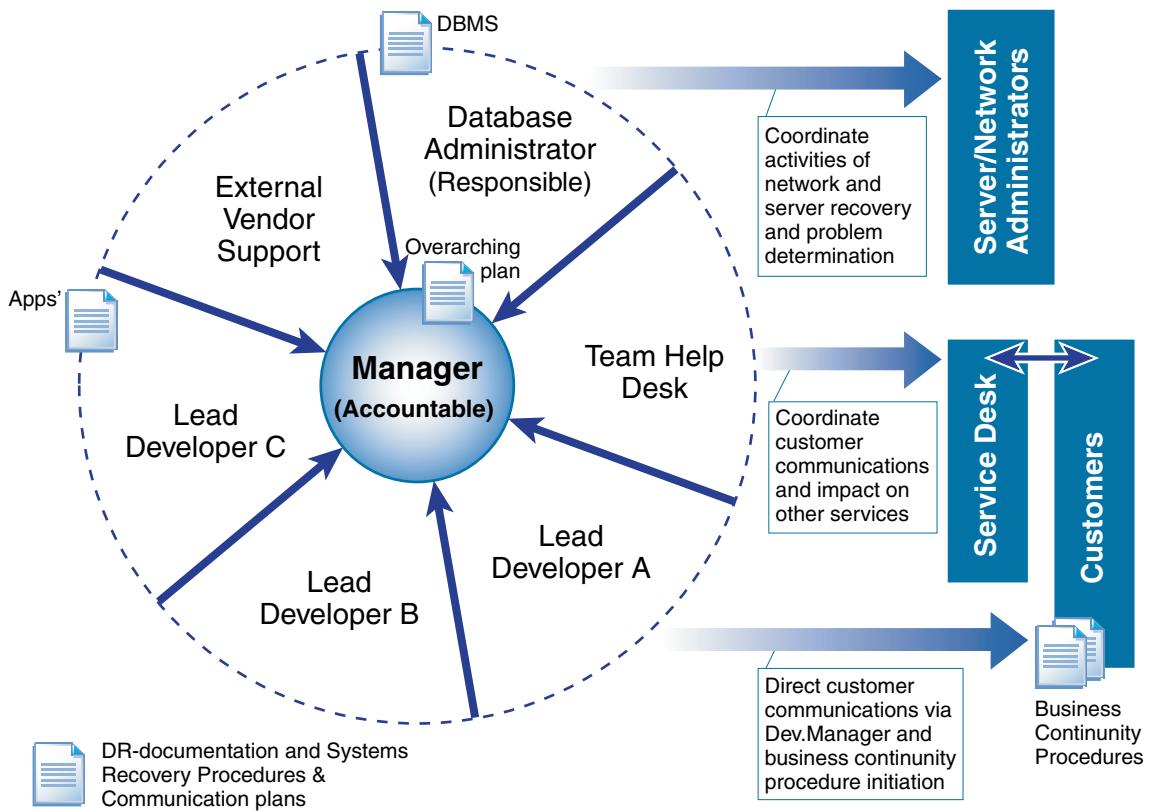
Building Staff Capability

Consider an Emergency Response Team (ERT)

An emergency response team (ER for IT or your dedicated production support team and signatory body and owners of all disaster recovery plans) will:

1. verify (testing) backups and associated recovery procedure checklists
2. assist customers and management in recognizing risks and steps (and costs) to mitigate them
3. respond to, manage, and coordinate emergency recovery procedures
4. ensure staff capability training policies are in place to better facilitate systems recovery, especially as services change over time, be they new technologies or upgrades ones.
5. attend change management meetings and be well advised of new system development initiatives or purchases/evaluations
6. be actively involved in business continuity planning; to ensure staff have a solid understanding of the procedures in place, and what *manual* procedures need to be followed if system downtime is in the order of hours or days.

As a very basic example, I worked in a small development team (25 staff in total) for a Government Department. The business unit was responsible for all business systems development and subsequently included some core business applications. The ER-team was formed based upon the following: This sort of model will differ based on the underlying IT governance model your business is pursuing, its culture and political boundaries, and of course budgetary constraints. That said, the driver here is the *coordination of team members* via a strict set of recovery processes, with the responsibilities defined and documentation in place to monitor, measure and coordinate actions during an emergency.



DBA Training and Skills (Building ERT Expertise)

In terms of database administration *roles*, I divide them into the following streams based upon different skill sets and IT focuses that of course mean a different style of training and self improvement (we do not cover mixed roles, namely Developer/DBA etc.):

DBA STREAM	SKILL REQUIREMENTS	TRAINING PLAN	DIFFERENTIATING YOURSELF
Production Support DBA	<ul style="list-style-type: none"> Highly skilled pure DBA with expert knowledge of all facets of the DBMS. Has little involvement with developers apart from tuning on request. Provide expert advice and skills for system upgrades, replication, advanced SQL and DBMS features. Expert in maintaining multiple, large and small scale databases systems in a production environment. Expert in database and systems recovery scenarios. Highly skilled at “thinking” 	<p>The production support DBA is a specialist role. The DBA is typically found in large data centers, ISP's or business with numerous mission critical databases that rely on a fulltime production presence.</p> <p>The DBA must focus on certification and the key driver for ongoing skills development, along with seminars and short term training courses relevant to the technologies planned or being implemented.</p> <p>The DBA must have a solid</p>	<p>This role requires the DBA to cross skill in Oracle or DB2 at an absolute minimum with certification.</p> <p>Consider future roles such as Chief DBA, Operations Manager and beyond.</p>

DBA STREAM	SKILL REQUIREMENTS	TRAINING PLAN	DIFFERENTIATING YOURSELF
	<p>outside the box”</p> <ul style="list-style-type: none"> • Applying systems recovery calmly and methodically. • 24x7 support is often required. 	<p>grasp of recovery and high availability scenarios; as such, time must be given to sustain these skills.</p> <p>Consider jobs that encompass a wide range of databases vendors and numerous production instances or where uptime is critical and achieved through enterprise-class hardware and software features.</p>	
Application Development DBA	<ul style="list-style-type: none"> • Highly skilled in writing, tuning and passing on expert SQL and DBMS design knowledge to developers. • Highly skilled in logical and physical database modeling, including classic patterns, (de)normalization, modeling tools • Advanced DTS • Good understanding of the implications and development requirements or issues of using replication, clustering, XML in the DB, blob data storage, large database design etc. • OLAP management and good MDX skills • Average system administration skills, including the OS, IIS, FTP, encryption, SSL • Database performance tuning. • Skilled at Backup/Recovery (typically due to the fact that developers require numerous copies of databases) • Server performance management, marrying up figures to the underlying DBMS • Expert in profiler, user and DB tracing, especially blocking, locking and deadlocks • Has a good understanding of 	<p>To remain effective in this role the DBA needs to thoroughly embrace and cross skill in the languages and development technologies in play against the DBMS – but at the same time not becoming part of the development team and locking you in as a “Developer/DBA” which is a very different role. That said, back-end development in T-SQL etc, is a core requisite, include complex DTS builds etc. These skills should be continually developed; don’t alienate yourself from the development team.</p> <p>The DBA is asked frequently about indexing, fragmentation management, and moving/exporting/importing/restoring databases throughout the day.</p> <p>More importantly, the DBA must keep on top of any new DBMS feature and determining the costs/benefits and issues with its use and implementation for the sake of the team and its agility. Major decisions will be made off the DBA’s</p>	<p>The DBA should really take a team leadership role where possible; bringing the team together and driving the importance of standards, procedure, change management, DBMS design and SQL improvements etc.</p> <p>Consider running Special Interest Group sessions during lunch times to drive this goal.</p> <p>Keep highly informed, it is one thing understanding the technology, but apply critique, cost/benefit analysis backed by the wisdom of</p>

DBA STREAM	SKILL REQUIREMENTS	TRAINING PLAN	DIFFERENTIATING YOURSELF
	<p>the SQL Server engine, but is a little rough in some areas due to the wide range of skills required</p> <ul style="list-style-type: none"> • Can tend to be a little on the “cowboy” DBA side and care must be taken to pull them into line to follow process and procedures. • Regarded as a “Jack of all trades”. 	recommendations, research skills and ability to sell and pursue ideas are important.	<p>yourself, others and research is the magic value add. Strive for this in the role.</p> <p>Consider IT consulting courses over certification.</p>
Applications DBA	<ul style="list-style-type: none"> • Good DBA skills, solid daily administration, SQL tuning, profiling and tracing • Good understanding of OLAP, DTS, MSDTC, XML, TSQL etc • Good to Expert knowledge of two or more core vendor applications (Microsoft, SAP, Oracle etc), especially for application customization and extension • Advanced TSQL, DTS skills. May include 3rd party language and excellent report writing skills. 	The applications DBA is a skilled DBA role with specialist knowledge of a enterprise class application, such as SAP over SQL Server for example. Apart from a thorough understand of the underling database structures, the DBA has expert knowledge of deployment, setup, and administration of the application over the chosen DBMS platform. Training should be clearly orientated around these skills and maintaining them. At the same time, looking at vendor hooks for adhoc reporting and application extension should be considered and make part of the persons work plan.	<p>Consider enterprise class application only, namely SAP, CRM, Portal, BizTalk and other Integration technologies, Oracle Applications etc. Specialist skills in such products are highly sought after, but watch the market carefully; especially your local one and adapt accordingly.</p>

CHANGE CONTROL

The need for a managed change control procedure is a fundamental requirement for any IT department. In terms of disaster recovery, it allows the business to analyze the risk a change will have on business as usual, and clearly spells out the pre and post tasks to be performed when applying a change. The idea here is managed change to reduce human error and impact on the business. This chapter will the policies and procedures of an example change management system and detail the use of Microsoft Visual Source safe for source code management.

Managing Change Control by Example

This section will discuss the processes I used from day to day for change management in a relatively large development. We will cover:

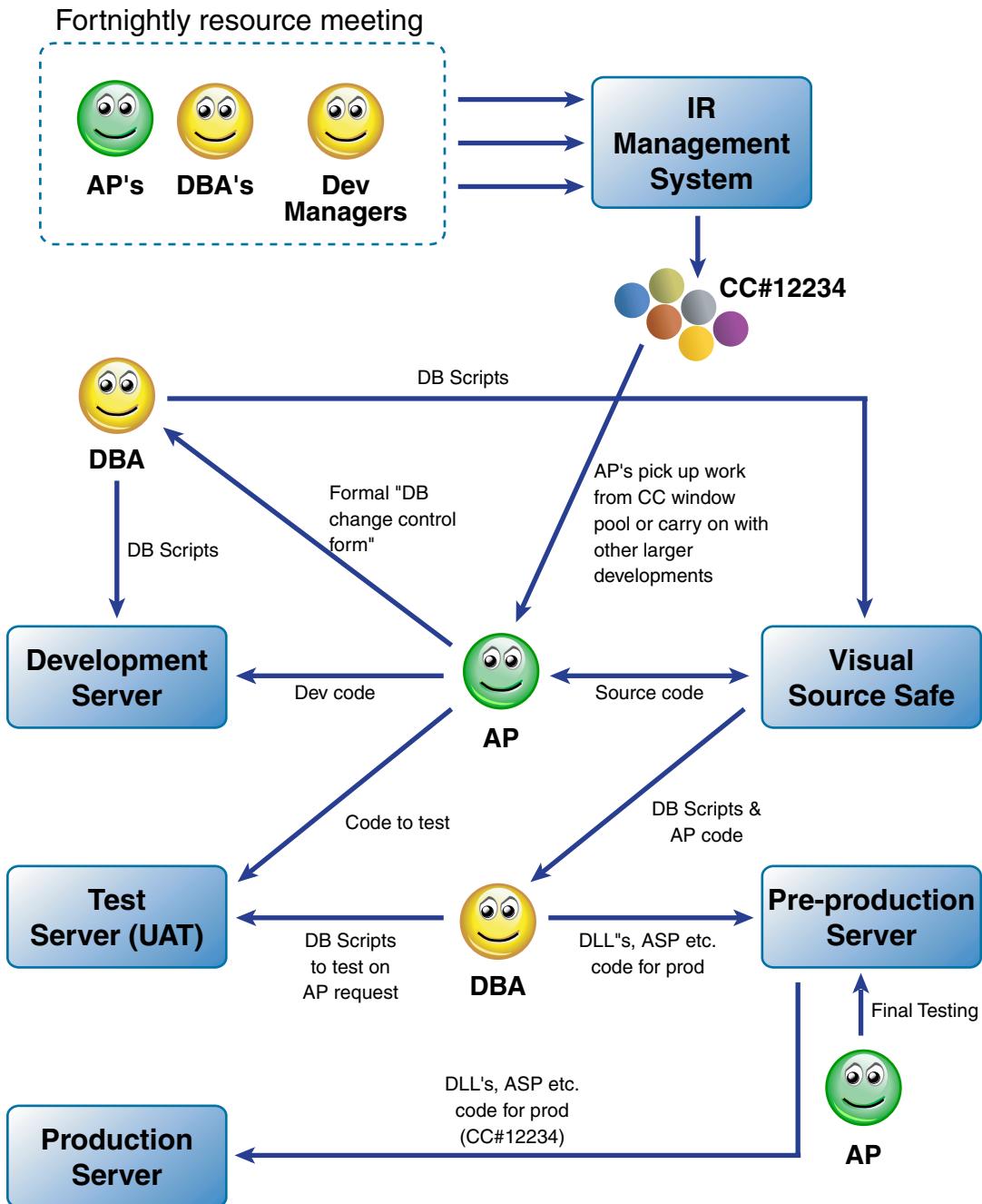
1. formalizing the process
 - a) document
 - b) agree upon and prepare the environment
 - c) build and maintain list of definitive software and server install audit log
 - d) management support
2. database script management
3. developer security privileges in extreme programming environments
4. going live with change
5. managing ad-hoc (hot fix) changes

Environment Overview

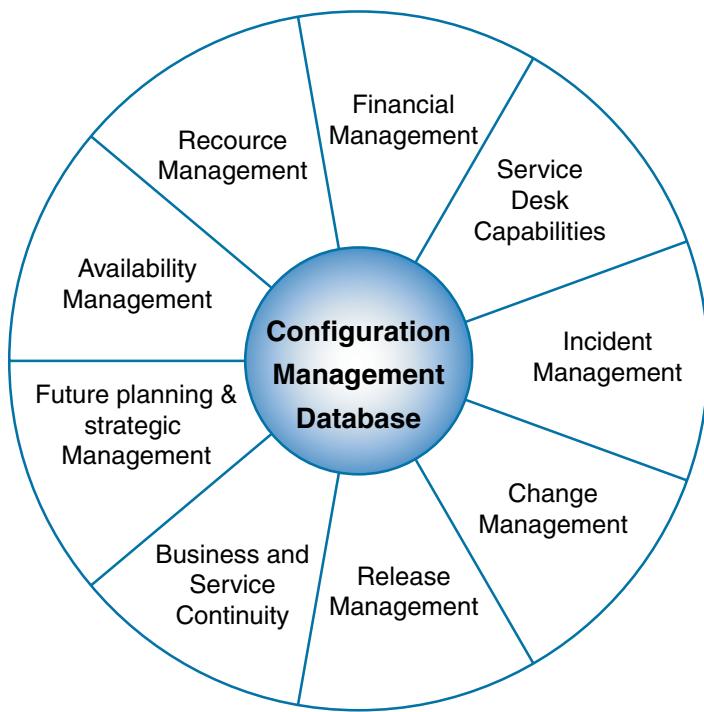
With any serious, mission critical applications development, we should always have three to five core environments in which the team is operating. They include:

1. Development
 - a) Rarely the environment (database) is rebuilt from production. Its servers are generally busy and reflects any number of pending change controls, some of which never get to test and others go all the way through to production.
2. Test
 - a) refreshed from production of a regular basis and in sync with a “batch” of change controls that are going to production within a defined change control window.
 - b) ongoing user acceptance testing
 - c) database security privileges reflect what will (or is) in production
3. Production Support (optional, some regard as Test) / Maintenance
 - a) mirror of production at a point in time for user testing and the testing of fixes or debugging of critical problems rather than working in production.
4. Pre-production or Compile Server (optional)
 - a) a copy of production as of “now”
 - b) used when “compiling code” into production and the final pre-testing of production changes
 - c) locked down image to ensure maximum compatibility on go live

5. Production The cycle of change is shown in the diagram below through some of these servers:



The whole change management system, be it in-house built or a third party product has seen a distinct shift to the whole CRM (relationship management) experience, tying in a variety of processes to form (where possible) this:



This ties in a variety of policy and procedures to provide end-to-end service delivery for the customer. The “IR (incident record) database” shown does not quite get meet all requirements, but covers resource planning, IR and task management, and subsequent change window planning. With good policy and practice, paper based documentation of server configuration and application components will assist in other areas of the service delivery and maintenance.

See ITIL framework for complete coverage of the underling concepts presented in the diagram above.

Pre-Change Window Resource Meeting

Every fortnight the team leaders, DBAs and the development manager discuss planned and continuing work over the next two weeks. The existing team of 20 contract programmers works on a variety of tasks, from new development projects extending current application functionality (long term projects and mini projects) to standard bug (incident request) fixing and system enhancements. All of this is tracked in a small SQL Server database with an Access front end, known as the “IR (incident reporting)” system.

The system tracks all new developments (3 month max cycle), mini projects (5–10 days), long term projects (measured and managed in 3 month blocks) and other enhancements and system bugs. This forms the heart and soul of the team in terms of task management and task tracking. As such, it also drives the change control windows and what of the tasks will be rolled into production each week (we have a scheduled downtime of 2 hours each Wednesday for change controls).

The resource meeting identifies and deals with issues within the environments, tasks to be completed or nearing completion, and the work schedule over the next two weeks. The Manager will not dictate the content of the change window but guide resource and task allocation issues. The team leaders and the development staff will allocate their tasks to a change control window with a simple incrementing number representing the next change window. This number and associated change information in the IR database is linked to a single report that the DBA will use to on Tuesday afternoon to "lock" the change control away and use it to prepare for a production rollout.

Visual Source Safe (VSS)

The key item underpinning any development project is source control software. There is a variety on the market but most sites I have visited to date use Microsoft VSS. Personally, I dislike the product; with its outdated interface, lack of functionality and unintuitive design, it's something most tend to put up with (its better than nothing!). Even so, a well managed and secured VSS database is critical to ongoing source management.

Consider These Items When Using VSS:

- Spend time looking around for better change manage front-ends that lever off the VSS API / automation object model, if possible. Web-based applications that allow remote development would be handy.
- Consider separate root project folders for each environment \$/
 - development
 - test (unit test)
 - production
- Understand what labeling and pinning mean in detail, along with the process of sharing files and repining. These fundamentals are often ignored and people simply make complete separate copies for each working folder or worse still, have a single working folder for dev, test and production source code (i.e. 1 copy of the source).
- All developers should check in files before leaving for the day to ensure backups cover all project files.
- Take time to review the VSS security features and allocation permissions accordingly.
- If pinning, labeling, branching etc is too complex, get back to basics with either three separate VSS databases covering development, test and production source code, or even three project folders. Either way the development staff needs to be disciplined in their approach to source control management.
- Apply latest service packs.

We will discuss VSS in depth later in the chapter.

Managing Servers

There are few development teams that I have come across that have their own server administrators. It is also rare that the servers fall under any SOE or contractual agreement in terms of their ongoing administration on the LAN and responsibility of the IT department. As such, the DBA should take the lead and be responsible for all server activities where possible, covering:

- server backups – including a basic 20 tape cycle (daily full backups) and associated audit log. Insist that tapes are taken off site and keep security in mind.
- software installed – the DBA should log all installations and de-installations of software on the server. The process should be documented and proactively tracked. This is essential for the future rollout of application components in production and for server rebuilds.
- licensing and terminal server administration
- any changes to active-directory (where applicable)
- user management and password expiration
- administrator account access

On the Development and Test servers I allow Administrator access to all developers to simplify the whole process. Before going live, security is locked down on the application and its OS access to mimic production as best we can. If need be, we will contact the companies systems administrators to review work done and recommend changes.

In terms of server specifications, aim for these at a minimum:

- RAID-1, 0+1 or RAID-5 for all disks – I had 4 disks fail on my development and test servers over a one year period. These servers really take a beating at times and contractor downtime is expensive. I recommend:
 - 2+ Gb RAM minimum with expansion to 4+ Gb
 - Dual PIII 900Mhz CPU box

Allowing administrative access to any server usually raises concerns, but in a managed environment with strict adherence of responsibilities and procedure, this sort of flexibility with staff is appreciated and works well with the team.

Development Server

The DBA maintains a “database change control form”, separate from the IR management system and any other change management documentation. The form includes the three core server environments (dev, test and prod) and associated areas for developers to sign in order for generated scripts from dev to make their way between server environments. This form is shown below: In terms of security and database source management, the developers are fully aware of:

Database Change Request Form	
Application Name : _____	Server IP(s) : _____
Database Name : _____	Request Date&Time : _____
Change Requestor : _____	
Change Approval / Acceptance	
Approved (Dev) : <small>(Business Analyst / Date)</small>	DBA Only [____] [____] <small>backup done</small> _____ signature / date
Approved (Test) : <small>(Business Analyst / Date)</small>	[____] [____] <small>backup done</small> _____ signature / date
Approved (Prod) : <small>(Business Analyst / Date)</small>	[____] [____] <small>backup done</small> _____ signature / date
Dependencies (list all dependencies that must occur prior to the changes being run): 	
Change Details :	

In terms of security and database source management, the developers are fully aware of:

- naming conventions for all stored procedures and views
- the DBA is the only person to make any database change
- database roles to be used by the application database components
- DBO owns all objects
- Database roles will be verified and re-checked before code is promoted to test
- Developers are responsible for utilizing visual source safe for stored procedure and view management
- the DBA manages and is responsible for all aspects of database auditing via triggers and their associated audit tables

- production server administrators must be contacted when concerned with file security (either the DBA if they have the responsibility or system administrators) and associated proxy user accounts setup to run COM+ components, ftp access, and security shares and to remove virtual directory connections via IIS used by the application.
- strict NTFS security privileges

With this in mind, I am quite lenient with the server and database environment, giving the following privileges. Be aware that I am a change control nut and refuse to move any code into production unless the above is adhered to and standard practices are met throughout the server change cycle. There are no exceptions.

1. Server

- a) Administrator access is given to all developers via terminal services to manage any portion of the application
- b) DBA is responsible for server backups to tape (including OS, file system objects applicable to the application and the databases)

2. Database

- a) ddl_admin access – to add, delete or alter stored procedures, views, and user defined functions.
- b) db_securityadmin access – to deny/revoke security as need be to their stored procedures and views.

No user has db_owner or sysadmin access; DBA's should be aware that developers may logon to the server as administrator and use the built-in/administrator account to attain sysadmin access. I do not lock it down, but make people fully aware of the consequences (i.e. changes don't go to production and may be promptly removed).

Database changes are scripted and the scripts stored in visual source safe. The form is updated with the script and its run order or associated pre-post manual tasks to be performed. To generate the scripts, I am relatively lazy. I alter all structures via the diagrammer, generate the script, and alter those that can be better scripted. This method (with good naming conventions) is simple and relatively fail-safe, and may I say, very quickly. All scripts are stored in VSS.

The database is refreshed during “quite” times from production. This may only be a data refresh but when possible (based on the status of changes between servers), a full database replacement from a production database backup is done. The timeline varies, but on average a data refresh occurs every 3–5 months and a complete replacement every 8–12 months.

Test Server

The test server database configuration in relation to security, user accounts, OS privileges, database settings are close to production as we can get them. Even so, it's difficult to mimic the environment in its entirety as many production systems include web farms, clusters, disk arrays etc that are too expensive to replicate in test.

Here the DBA will apply scripts generated from completed change control forms that alter database structure, namely tables, triggers, schema bound views, full-text indexing, user defined data types and changes in security (and which have already run successfully in development). The developers will ask the DBA to move up stored procedures and views from development into test as need be to complete UAT (user acceptance testing).

The DBA will "refresh" the test server database on a regular basis from production. This tends to coincide with a production change control window rollout. On completion of the refresh, the DBA might need to re-apply database change control forms still "in test".

All scripts are sourced from VSS.

Refreshing TEST from PRODUCTION

We will only dot point one of many possibilities from a database perspective, the process can be very complex with large numbers of application components and runtime libraries.

- Notify developers of your intention – check that important user or change testing is not underway. The DBA doesn't really want to synchronise the table data as it can be an overly complex task in large database schemas (100+ tables for example – my last DB had 550!)
- Check free space on the test server to restore the database backup from production (backup file) and accommodate the expanded storage after the restore.
 - If the production databases are huge and the only free space is in production, then consider restoring a recent copy of production as "test_<prodname>" within the production instance, deleting/purging appropriate records and shrinking before taking a backup of this database over to test.
- Restore the database into the test database instance as "new_<db-name>" (for example), allowing the developers to continue using the existing test database.
- Fix the database users to the logins (sp_change_user_login) as required – retain existing default database
- Notify developers that no further changes to DBMS structure will be accepted.

- Use a scripting tool (SQL Compare from RedGate software is very good) to compare the existing database structures. You may not take over all existing changes. Go back over your change control documentation and changes raise and marked as “in test” as these will be the key changes the developers will expect to see. This process can take a full day to complete and you may have to restore again if you get it wrong (very easy to do).
- Email development staff notifying them of the cutover.
- Switch the databases by renaming them.
- Check user and logins carefully, change the default database as required.
- Fix full text indexes as required.
- Fix any internal system parameter or control tables that may be referring to production. Copy the data from the old test database as required.
- Notify developers that the database is available.

Production Support

The production support server box is similar to that of test, but is controlled by the person who is packaging up the next production release of scripts and other source code ready for production. This server is used for:

- production support – restoring the production database to it at a point in time and debugging critical application errors, or pre-running end of month/quarter jobs.
- pre-production testing – final test before going live with code, especially handy when we have many DLL’s with interdependencies and binary compatibilities issues.

All database privileges are locked down, as is the server itself.

Production

The big question here is, “who has access to the production servers and databases?” Depending on your SLAs, this can be wide and varied, from all access to the development team via internally managed processes all the way to having no idea where the servers are let alone getting access to it. I will take the latter approach with some mention of stricter access management.

If the development team has access, it’s typically under the guise of a network/server administration team that oversee all servers, their SOE configuration and network connectivity, OS/server security and more importantly, OS backups and virus scanning. From here, the environment is “handed over” to the apps team for application configuration, set-up, final testing and “go live”.

In this scenario, a single person within the development team should manage change control in this environment. This tends to be the application architect or the DBA.

When rolling out changes into production:

1. application messages are shown (users notified)
2. web server is shutdown
3. MSDTC is stopped
4. Crystal reports and other batch routines scheduled to run are closed and/or disabled during the upgrade
5. prepare staging area "c:\appbuild" to store incoming CC window files
6. backup all components being replaced, "c:\appatches\<system>\YYYYMMDD"
 - a) I tend to include entire virtual directories (even if only 2 files are being altered)
 - b) COM+ DLL's are exported and the DLL itself is also copied just in case the export is corrupt
7. full backup of the database is done if any scripts are being run
8. consider a system state backup and registry backup, emergency disks are a must and should always be kept up to date.

Take care with service packs of any software (never bundle a application change with a service pack). The change (upgrade or downgrade) of MDAC, and the slight changes in system stored procedures and system catalogs with each SQL Server update, can grind parts (or all) of your application to a halt.

Here is an example of an *appatches* directory on the server we are about to apply a application change to. The directory is created, and all files to be replaced are copied:

The screenshot shows a Windows file explorer window. On the left is a tree view of the 'appatches' directory structure, which contains several subfolders named 20031009, 20031016, 20031023, 20031106, 20031107, 20031120, 20031121, 20031204, 20031215, and 20031218. A red arrow points from the tree view to the file list on the right. The file list is a table with columns: Name, Size, Type, and Modified. It lists several files and folders:

Name	Size	Type	Modified
trs.MSI.cab	224 KB	WinZip File	18/12/2003 5:30 PM
trs.MSI	263 KB	Windows Installer P...	18/12/2003 5:30 PM
jidjobprocessor.M...	93 KB	WinZip File	18/12/2003 5:29 PM
jidjobprocessor.MSI	131 KB	Windows Installer P...	18/12/2003 5:29 PM
organisation.MSI....	524 KB	WinZip File	18/12/2003 5:29 PM
organisation.MSI	569 KB	Windows Installer P...	18/12/2003 5:29 PM
trs		File Folder	18/12/2003 5:31 PM
organisation		File Folder	18/12/2003 5:31 PM
crystalreports		File Folder	18/12/2003 5:30 PM
cpl		File Folder	18/12/2003 5:30 PM

The DBA may choose to copy these files to tape or copy to another server before running the upgrade. If the change was a virtual directory, I would copy the entire directory rather than selective files, it simplifies backup process and avoids human error.

Hot Fixes

Unless you are running a mission critical system, there will always be minor system bugs that result in hot fixes in production. The procedure is relatively simple but far from ideal in critical systems.

1. Warn all core users of the downtime. Pre-empt with a summary of the errors being caused and how to differentiate the error from other system messages.
2. If possible, re-test the hot fix on the support server.
3. Bring down the application in an orderly fashion (e.g. web-server, component services, sql-agent, database etc).
4. Backup all core components being replaced/altered.

Database hot fixes, namely statements rolling back the last change window, are tricky. Do not plan to disconnect users if possible. But careful testing is critical to prevent having to do point in time recovery if this gets worse.

Finally, any hot fix should end with a $\frac{1}{2}$ page summary of the reasons why the change was made in the monthly production system report. Accountability is of key importance in any environment.

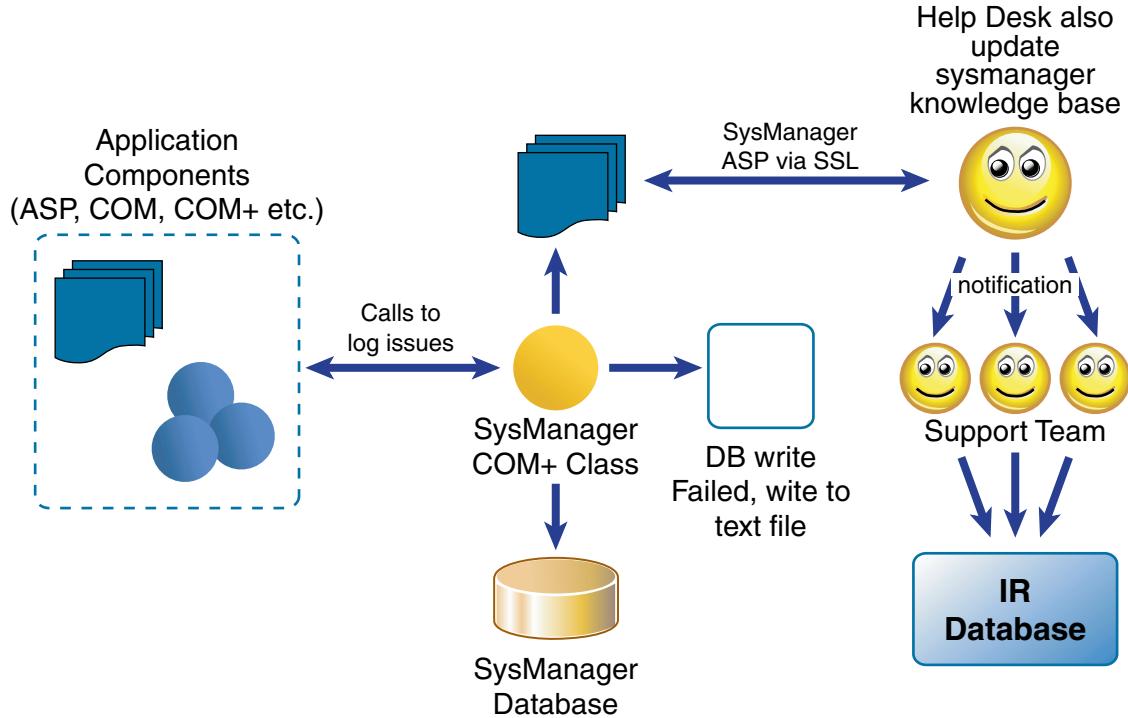
Smarten Up Your Applications (Autonomic Computing)

Autonomic computing “is an approach to self-managed computing systems with a minimum of human interference” (IBM). In other words, self repairing, reporting, managing systems that looks after the whole of the computing environment. So what has this got to do with change management? everything actually.

The whole change management process is about customers and the service we provide them as IT professionals. To assist in problem detection and ideally, resolution, system architects of any application should consider either:

1. API for monitoring software to plug in error trapping/correcting capability
2. Application consists of single entry point for all system messages (errors, warning, information) related to daily activity
3. The logging system is relatively fault tolerant itself, i.e. if it can not write messages to a database it will try a file system or event log.
4. Where possible, pre-allocate range of codes with a knowledge base description, resolution and rollback scenario if appropriate. Take care that the numbers allocated don't impose on sysmessages (and its ADO errors) and other OS related error codes as you don't want to skew the actual errors being returned.

A simplistic approach we have taken is shown below; it's far from self healing but meets some of the basic criteria so we can expand in the future:



It is not unusual for commercial applications to use web-services to securely connect to the vendors support site for automatic patch management, error logging and self correction. This adds a further complexity for the system administrators in terms of firewall holes and their ability to control the application.

MRAC of IR/Task Completion

This is going off track a little in terms of change control but I felt its worth sharing with you. The MRAC (Manage, Resource, Approve, Complete) principle is a micro guide to task management for mini projects and incident requests spanning other teams/people over a short period of time. The idea here is to get developers who own the task to engage in basic project management procedures. This not only assists in documenting their desired outcome, but communicating this to others involved and engaging the resources required to see the entire task through to its completion.

The process is simple enough as shown in the table below. The development manager may request this task breakdown at any time based on the IR's complexity. The developer is expected to meet with the appropriate resources and drive the task and its processes accordingly. This is not used in larger projects in which a skilled project manager will take control and responsibility of the process.

TASK OR DELIVERABLE	PLANNED COMPLETION DATE	MANAGED BY	RESOURCED TO	APPROVED BY	COMPLETED BY
Requirements					
Design					
Build					
Test					
Implement					
Audit (optional)					

The tasks of course will vary, but rarely sway from the standard *requirements, design, build, test, implement* life-cycle. Some of the key definitions related to the process are as follows:

Managed	Each task or deliverable is managed by the person who is given the responsibility of ensuring that it is completed.
Resourced	The person or persons who are to undertake a task or prepare a deliverable.
Accepted	The recorded decision that a product or part of a product has satisfied the requirements and may be delivered to the Client or used in the next part of the process.
Approved	The recorded decision that the product or part of the product has satisfied the quality standards.
Authorized	The recorded decision that the record or product has been cleared for use or action.
Variation	A formal process for identifying changes to the Support Release or its deliverables and ensuring appropriate control over variations to the Support Release scope, budget and schedule. It may be associated with one or more Service Requests.

This simple but effective process allows developers and associated management to better track change and its interdependencies throughout its lifecycle.

Summary

No matter the procedures and policies in place, you still need the commitment from management. Accountability and strict adherence to the defined processes is critical to avoid the nightmare of any project, that being source code versions that we can never re-created, or a production environment in which we do not have the source code.

Failure to lay down the law with development staff (including the DBA) is a task easily put in the too hard basket. It is not easy, but you need to start somewhere.

This section has presented a variety of ideas on the topic that may prompt you to take further action.

Using VSS by Example – Part 1

This section is divided in multiple parts to introduce you to a VSS (Microsoft Visual Source Safe) framework that I have successfully used for a large applications development team. It provides a working framework for source code management in Microsoft Visual Source Safe. The framework presented support a system currently under support and maintenance, as well as other development initiatives (mini projects) that may be adding new functionality.

For those working with multi-client software in which “production source code” is at different patch levels (and may include custom code), you will need to rework VSS to adopt the strategy—namely due to multiple side-by-side releases with different custom changes to the same product.

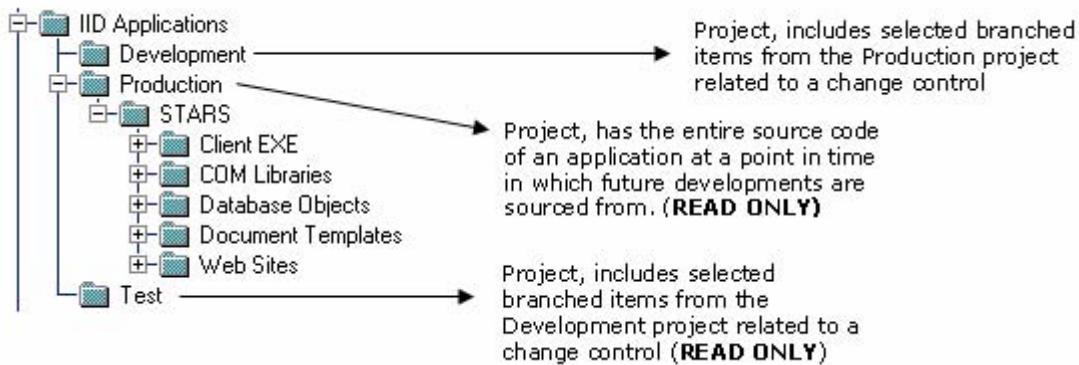
The article is scenario driven, starting with a very basic scenario. We then move to more complex changes.

Terminology

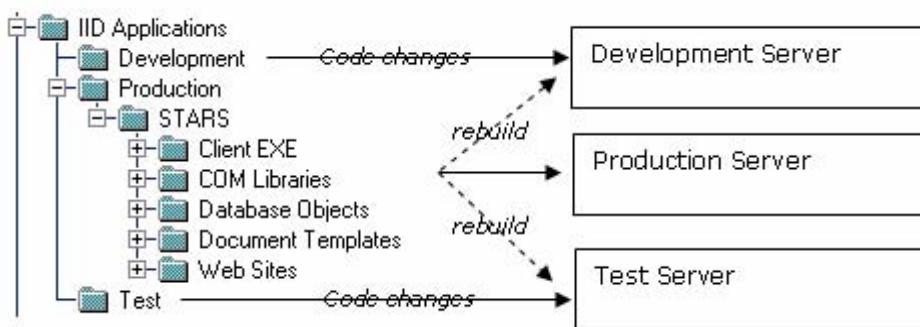
CCM	Change Control Manager.
PROJECT	Is similar to a folder in explorer. It contains configuration items (files), VSS will automatically version projects and files with an internal value.
BRANCH	Is synonymous to a <i>copy</i> in explorer, but VSS maintain links between <i>versions</i> of branches to facilitate <i>merging</i> at some later stage. VSS via its internal version numbering will maintain the “logical branch history” in order to facilitate merging with any other branch.
SHARE	A shared file is a “link” to its master, or in other terms, a shortcut. The shared file will be automatically pinned (cannot change this action by VSS); to alter it you must unpin it. If the shared file is altered its master will also be changed, along with any other “shortcut” (share).
MERGE	Process of merging the changes from two branched files. The VSS GUI will show the hierarchies of branches and associated links/paths, broken branches that can not be merged too, or branches from incompatible parents (can merge the branches from two different trees!).

Build Phase – Moving to the New Change Control Structure

The project folder layout of VSS is as follows:



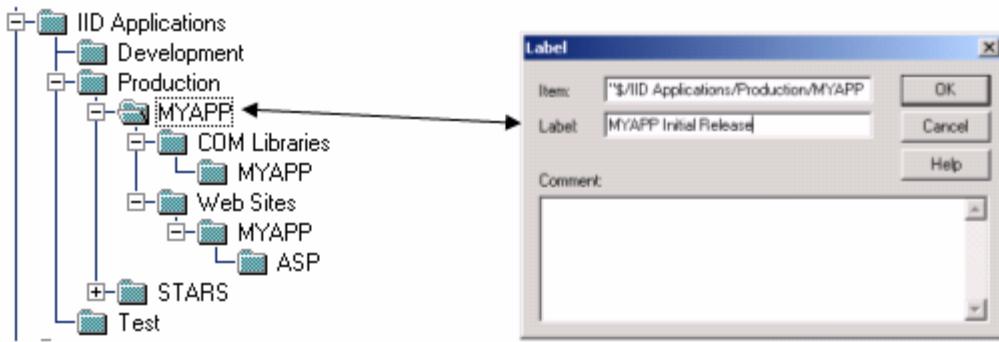
This structure maps to your equivalent development, test and server environments. Source code for development and test environments is built from VSS production code project (projects are shown as Windows explorer folders in VSS) and the subsequent development/test project code applied over top (roll forward) to form the environment. The developers need to manage themselves when there are multiple “versions” of a DLL or other file source moving into test over time. Either way, the VSS environment allows the full “build” of development or test servers from a known image of production source code (which is critical):



The /production project folder includes the names of other applications currently under the change control process. To start the process of change management using this structure, we will be introducing a new application call “MYAPP”. The CCM will:

1. create a new project in /production call “myapp”
2. create sub-projects to hold myapps’s source code
3. label the MYAPP project as “MYAPP Initial Release”
4. arrange a time with the developer(s) and check-in source code into the projects created

This will give the structure: The production folder is “read only” for all but the change control manager (CCM).



First Change Control

All application project changes in the /production project “folder”, follow a change control window path, as such, in the development project folder the CCM will:

1. create a new project in the development folder representing the change control window (if none exist)
2. notify developers of VSS development project status

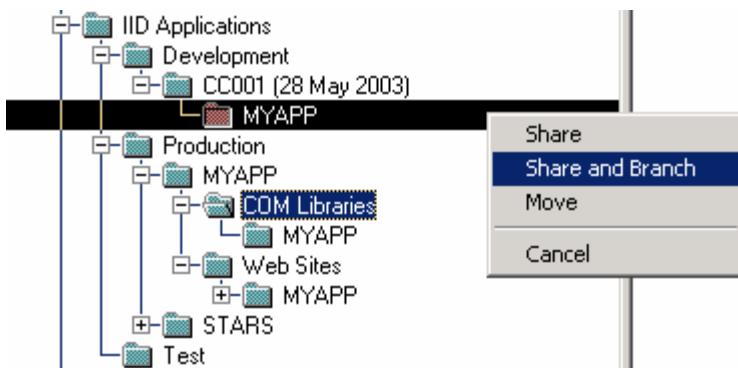
For example:



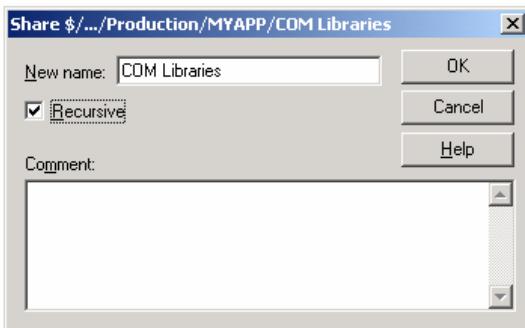
The MYAPP developer needs to alter his single ASP and COM files for this change control window. The developer will need to:

1. Create a “MYAPP” project under the CC001 project in development

Expand the production source code project, navigate to MYAPP and the COM Libraries project, then with a right click and drag the project to /development/cc001/myapp and from the menu displayed, select share and branch



Check the “recursive” check box if any sub-project folders are also required.



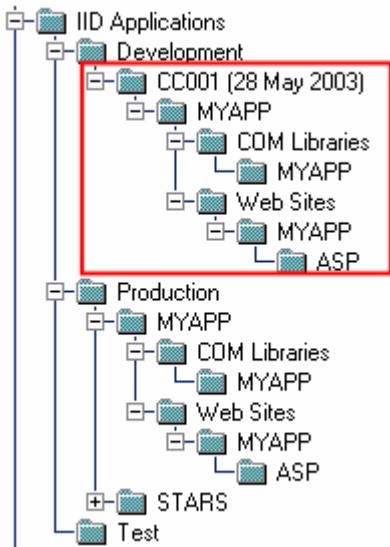
2. Do the same for Web Sites.



If there are multiple websites and you only want selected projects, then manually create the same structure in dev then use VSS to share the projects from there.

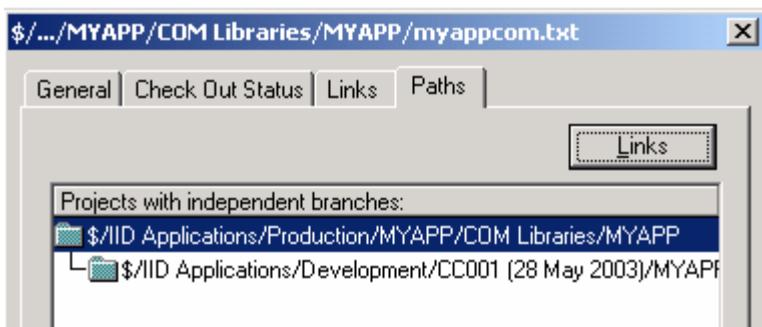
3. Revise what you have done—only share/branch the files required for the change control. If you forget files or require others, then repeat this process from the **same** production source project. Always keep the same project folder structure as in the production project.
4. If you make a mistake, delete your project and start again.

The above steps will leave us with the following project structure:



To review where the original source came from in the production folder:

1. select a file
2. right click *properties*
3. select the *paths* tab
4. click *links* if the GUI is hard to read



We can now check in/out code as need be in the development project folder. The DEV (development) server environment is then built (i.e. refreshed) from this source code in VSS.

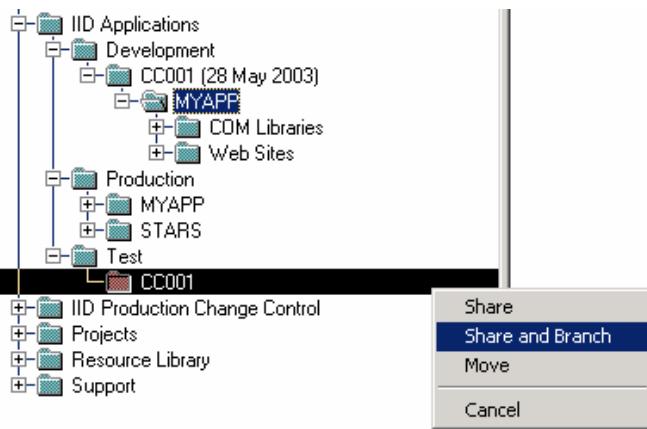
Moving Code to TEST

The CCM will:

1. create a CC001 project in the /test project “folder”

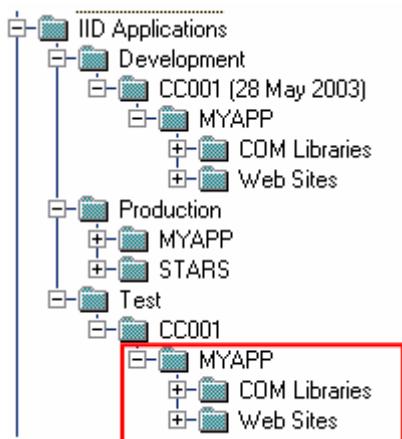
The developer(s) will:

2. remove any reference to “myapp” in the /test/cc001 project (if applicable)
3. expand the cc001/myapp project, select myapp then right click and drag to the /test/cc001 project, selecting the share and branch option



4. Select “recursive” check box (as all code for MYAPP is going up as a single unit of testing).

This will give us:



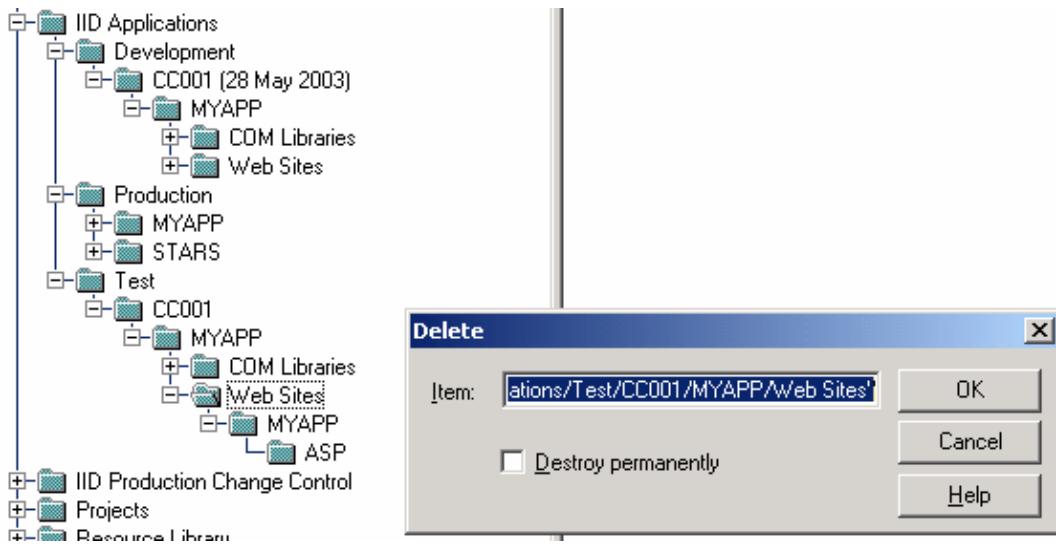
This scenario will assume all code will transition from DEV and TEST and into PROD. The TEST server environment is then built (refreshed) from this source in VSS.

Overwriting existing code in TEST from DEV

During testing we found a range of problems with our ASP page. We fixed the problem in /development and now want to overwrite/replace the VSS /test copy for the CC001 change control.

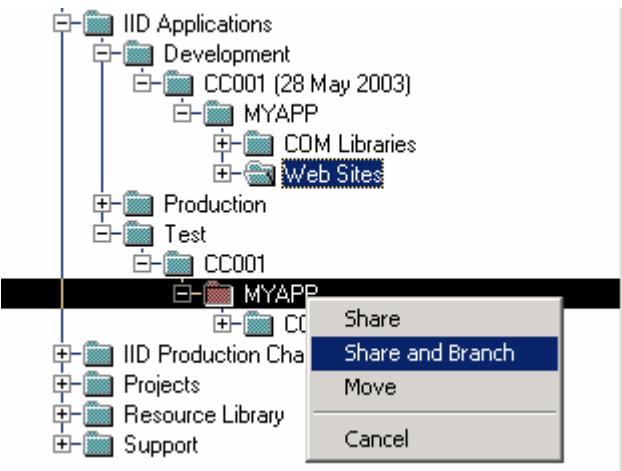
The developer will:

1. Check-in code in the development folder for the change control
2. Navigate to the /test/cc001/myapp/Web Sites/ project Delete from this point



You can do individual files if you need to, rather than a whole project.

3. Goto the /development/cc001/myapp project folder, right click and drag the websites project



Test now has the fixes made in dev. Use the files to build/update the test server environment.

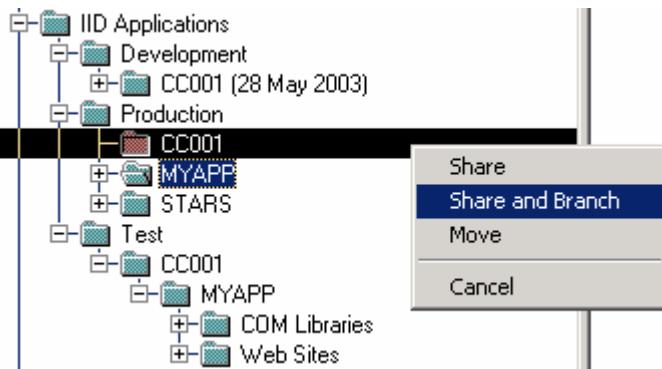
Taking a Change Control into Production

The developers will:

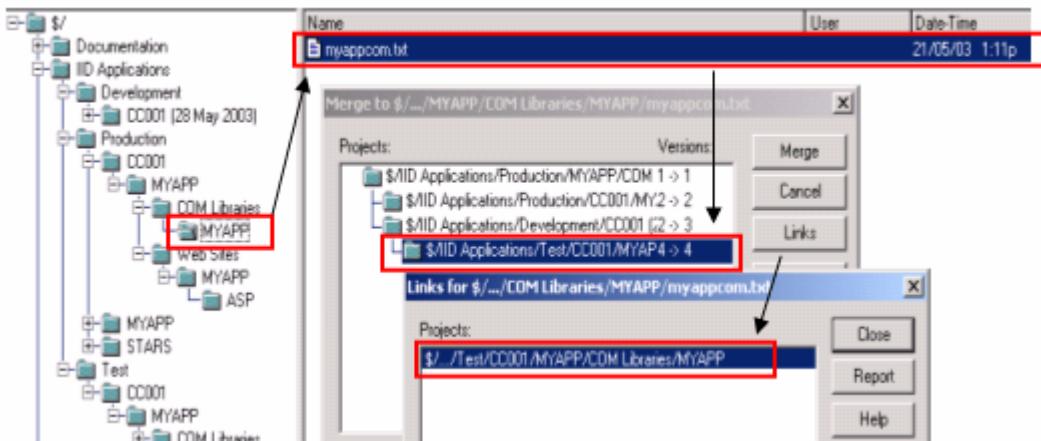
1. ensure all code in the /test/cc001 folder is OK to go live
2. remove any files that are not ready

The CCM will:

3. create a new project in the production project folder to represent the change control going “live”
4. share and branch the myapp application (entire structure) into the new project folder in production:



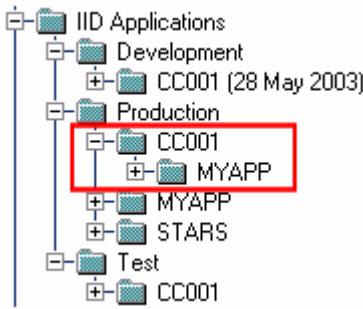
5. Navigate to the /test/cc001/myapp project, get a list of files to “go live”
6. Navigate back to /production/cc001/myapp project and locate these files, for each file:
 - a) Select the file, SourceSafe Merge Branches
 - b) Pick the appropriate branch to merge to



Press the *merge* button.

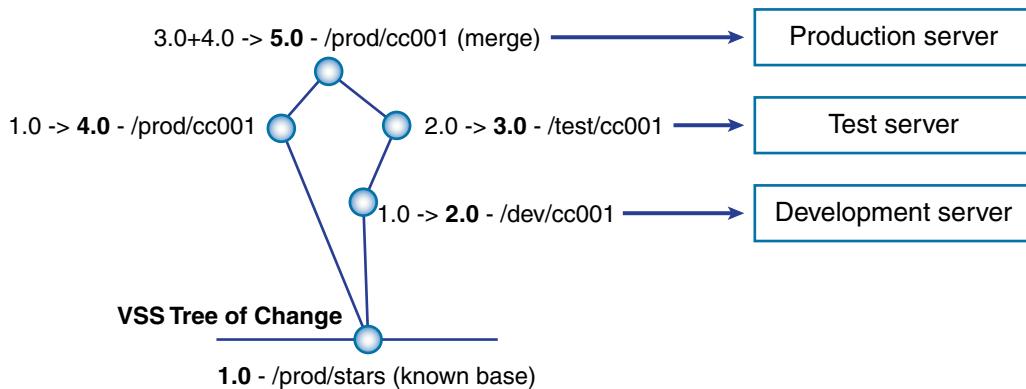
- c) Check conflicts if need be.

The /production project structure is:



Once all files are merged into the new production project folder, the production server environment can be built from these altered files.

The branching in VSS provides an effective solution to managing versions and the inter-relationships between components (files) and physical projects. The above scenario for example gave us this tree like structure to work from (versions in diagram are VSS internal versioning numbers—use labels to assist with source identification if you are having trouble with it):



Using VSS by Example – Part 2

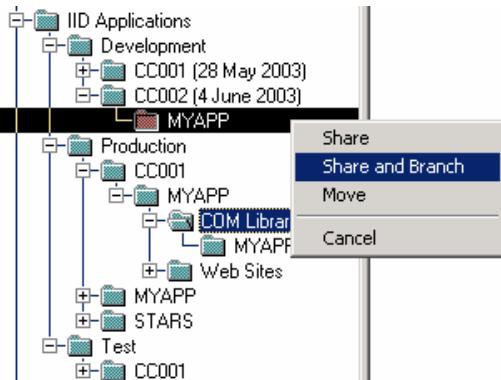
In this section we expand on other source control scenarios and how our defined project structure and VSS branching functionality accommodates it.

How Do I Move Files to Next Weeks Change Control?

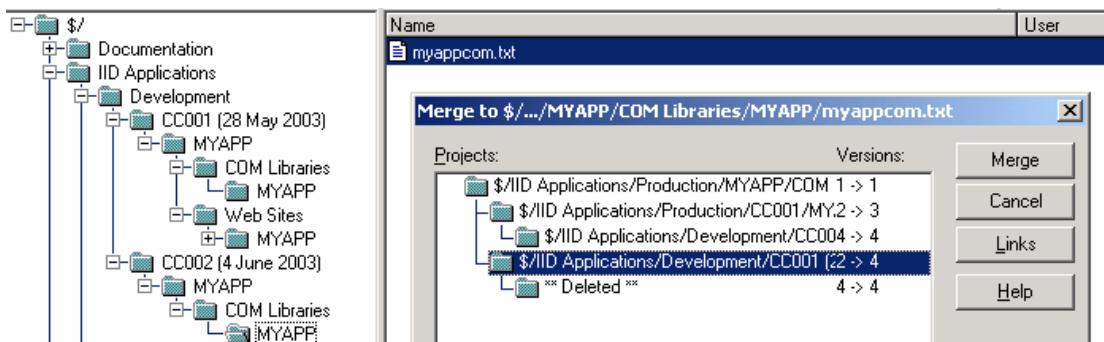
In the previous example, we only processed the single ASP file, for some reason, the developers wanted to delay the DLL for the next round of change controls. Therefore we need to get the COM file in /development/cc001 into the new project /development/cc002 for it to be included in that change window.

The CCM will:

1. Create /cc002 in /development folder
2. The developer will branch the DLL code from /production/cc001 into /development/cc002



3. Select the file(s) to be merged in /development/cc002.



4. Resolve the differences (we know what they are and are fine with all of them)

The screenshot shows a merge tool window titled "myappcom.txt: Merge \$/IID Applications/Development/CC001 (28 May 2003) and \$/IID Applications/Development/CC002 (4 June 2003)". It displays two panes of code. The left pane has a green border and contains the following lines:

```
1 myapp com v1.0
2 cc001 change in com
3 cc001 didnt go live change.
```

The right pane has a blue border and contains the same three lines. Below the panes is a "Merged version" section with a red border, containing only the first line:

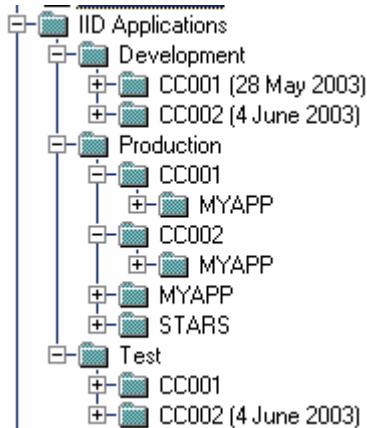
```
1 myapp com v1.0
```

At the bottom of the interface are buttons for "Deleted lines", "Changed lines", "Inserted lines", and "Ln 1, Col 1".

5. After the merge, double check your code before moving on with development.

What Does VSS Look Like After 2 Iterations of Change?

After two complete iterations of changes, we end up with this structure in VSS:



We have a snapshot of production for all iterations:

1. initial snapshot
2. change control 1
3. change control 2

We can also rebuild the test server from a production build and roll forward through change controls. In the development project, we have a well managed source environment for each change and its progression through the environments.

I Forgot to Take a File into Production for a Schedule Change

The CCM needs to:

1. check files are in the equivalent /test project folder
2. if not, the developer should update this project folder by branching the missing files from the equivalent development folder.
3. As the /production/ccxxx folder already exists (had to for other code to go into prod), the CCM simply merges this missing file into the project and takes the files from here to the production server environment.

I Have a Special Project That Will Span Many Weeks, Now What?

The scenario is this:

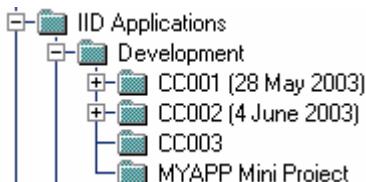
1. we do weekly change controls
2. a new special project will take 2 standard change control iterations to move into production
3. the standard change controls will be altering files the special project will be using
4. we need to ensure that the standard change controls are not affected and visa versa for the special project until the special project is ready to do live.

To manage this we need to ensure these rules apply:

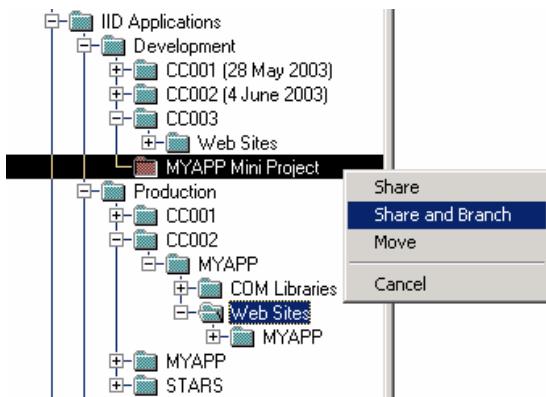
1. strict branching from a single point in time from the /production project folder
2. the special project files will merge with some future change control and go live with this change control window to ensure VSS consistency.

At CC003, we start a new special MYAPP project that affects the single ASP file we have. The CCM will:

1. create /development/CC003 project
2. create /development/MYAPP Mini Project



- The developer branches in the files to be altered into both /development/cc003 and /myapp mini project

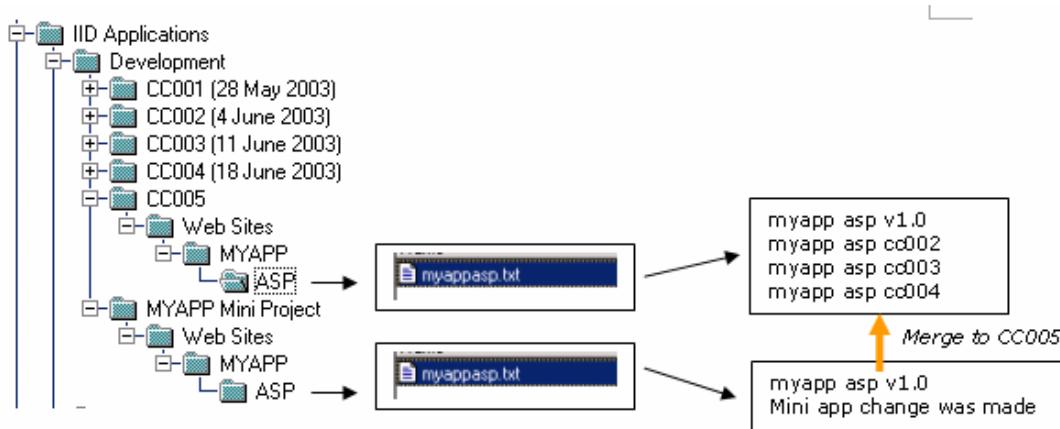


- Now both project folders in dev are branched from the single “tree root” (for one of a better word). Developers continue to work in DEV and branch to test as need be to facilitate standard app testing.

Merging “MYAPP Mini Project” back into the standard change control.

Here we assume CC003 and CC004 changes over the last 2 weeks are in production, we need to get MYAPP Mini Project live, this will be done in CC005. The VSS structure is:

- CCM creates /development/CC005
- CCM branches code used in “MYAPP Mini Project” from /production/cc004 (current production source)
- CCM then, working with the developers, merges into /development/MYAPP Mini Project with the /development/CC005 project files.



- Merging is not straight forward in VSS. Take care, and where possible attempt do this “offline” from the product (using 3rd party source compare tools for example).

The screenshot shows a 'Merge' dialog window with two panes. The left pane displays changes from the '\$/IID Applications/Development/MYAPP Mini Project' branch, showing lines 1 through 3. The right pane displays changes from the '\$/IID Applications/Development/CC005/Web Site' branch, showing lines 1 through 5. A central 'Merged version' pane at the bottom contains the line '1 myapp.asp v1.0'. Below the merged version pane are tabs for 'Deleted lines', 'Changed lines' (which is selected), and 'Inserted lines'. An arrow points from the 'Changed lines' tab down to the 'Merged version' pane.

Left Branch (\$/IID Applications/Development/MYAPP Mini Project)	Right Branch (\$/IID Applications/Development/CC005/Web Site)	Merged Version
1 myapp.asp v1.0	1 myapp.asp v1.0	1 myapp.asp v1.0
2 Mini app change was made	2 myapp.asp cc002	
3	3 myapp.asp cc003	
	4 myapp.asp cc004	
	5	

Merged version

1 myapp.asp v1.0

Deleted lines Changed lines Inserted lines ↴

Merged version

1 myapp.asp v1.0
2 Mini app change was made
3 myapp.asp cc002
4 myapp.asp cc003
5 myapp.asp cc004

- On completion of the merge, remove MYAPP Mini Project or renamed to “use CC005 – MYAPP Mini Project”
- Go through standard dev → test → prod VSS practices as outlined through this document.

Using VSS by Example – Part 3

Re-Iterating our Chosen VSS Structure

The structure we have chosen, i.e. `/development`, `/test`, `/production` VSS project folders, is perhaps seen by many as *just a file system with undelete and some labeling capability*, and that's fine. We have divided off the "environments" from one another and know that source for production at some known point in time is at location XYZ in VSS and not part of a single copy that is being used for ongoing development. This goes with the `/test` project and was the main reason why we went down this path.

It takes some time to get developers used to the fact that we do have three environments in VSS between which code is shared/branched (copied) between; for all intended purposes, developers should see no real difference in the way they currently develop and interact with VSS via their development software IDE.

The development staff, in liaison with a senior developer (the ultimate owner of code branched from `/dev` to `/test` then `/prod`), can setup the local PC development environment, and hook in the `/development` folder to their IDE with little effort, and continue working in this environment.

The developers need to use the VSS GUI to add code as required into `/development` folders. The only real complaint here is the need to branch code into `/test` which some complain as time consuming (but just darn lazy in my opinion).

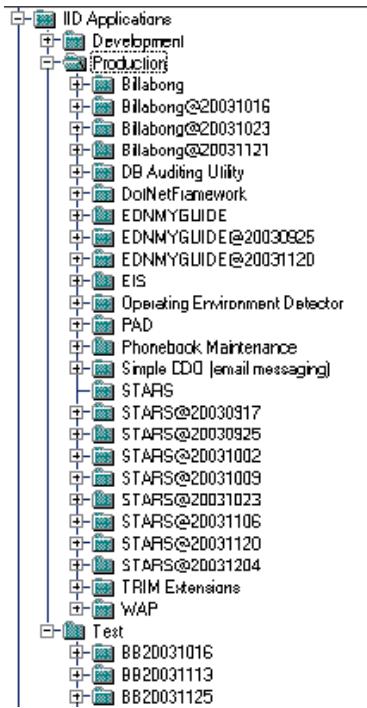
The key in `/test` is the naming of our folders. The folders represent the change control numbers, which means that your *testing of component XYZ is planned to be released into production as change control number 20041201*. If it isn't, then we branch/move it to another change control. All said and done, we are not overly strict in `/test` within VSS, we do allow other folders to be created to suit the needs of the team, but nothing rolls into `/production` that isn't inside of a `/test/cc` folder.

Finally the `/production` folder. The fundamental concept here is the full duplication of all source that makes up the production server before the change control was rolled into production. We copy this source to a new project folder and the change manager then merges code from our `/test/cc` folder into this copy, thus forming the new production source code image.

This method is simplistic and easy to use. As the change manager of VSS becomes more adept with the product, one may use labeling as the core identifier of source at a single point in time; be warned that it is not simple and a mistake can be very costly.

What Does My VSS Look Like to Date?

Here is the screen shot of my VSS project after seven iterations of scheduled change controls:



I will discuss /development next. As we see, /production has a “as of now” project folder for each application, and then the copies of the same folder “as at YYYYMMDD”. So for the STARS application we see:

```
/production/stars      (developers branch into /development from here for any changes to prod code)  
/production/stars20030917    (the STARS system and all its source code as at 17/09/2003)
```

I will archive off the projects in production as soon as the size of VSS gets too large. It grows quickly with such a scheme. Even so, it doesn’t slow VSS and disk space is plentiful (200 GB free so I am not stressed over it).

As we can see in the /test project folder, we have projects representative of the change control. It is important to remember that this folder is used to build the test server before formal testing of your changes to go up for that change control. It takes some discipline for developers to do this and not copy files from the development server to the test server without going via VSS. The developers are responsible at the end of the day, and I have solid backing from the development manager.

On a change control go live, I email the team notifying them of the lock down of the /test change control folder. I then check for checked-pout files in this project then use VSS security to lock the project. I then do a “get latest version” to my pre-production server, compile all DLL’s, developers test once I have emailed a “please test pre-prod” request, and wait for final approvals from the developers before 5.30pm that night.

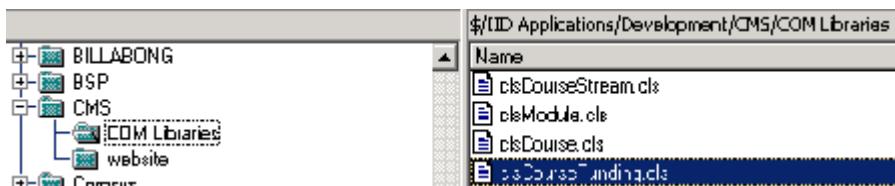
Generally, all working well to date.

What Do You Do with /Development After Each Change Control?

The general thinking here is the removal of all /development code that was taken into production for a change control and is not being worked on further. I let the senior developers take responsibility here and they manage this well. I have yet to see any issues caused by poor management of the /development project folder and its contents. The developers use the compare and merge facilities of VSS extensively between /test, /dev and at times /production.

What Do You Branch into /Development in Terms of VB6 COM Code?

You basically have two options here will branch all your COM project code into development, or branch only selected class files for the COM project. The development team opts for selected classes only, and does a get latest version on the production source to their local computer for compilation. This allows team member to more effectively track what classes are being altered in development without relying on the versioning of VSS, file dates or leaving files checked out. This tends to be senior developer specific on what approach is taken in the /development projects for each application.

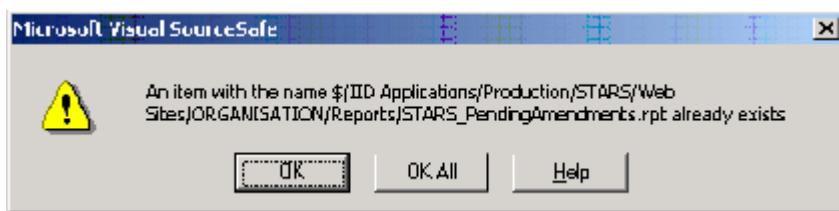


As shown above, the /development/cms application folder holding our COM (cms.dll), we find only 4 of the 16 class files for this component.

VSS Issues

Share/Branching Files from /Test into /Production

You cannot overwrite files with share/branch. If I attempt to share/branch the file myfile.txt from one project to another and the file exists at the destination, you are told this but the branch will give you no option to overwrite the file.



To get around this, delete the files at the destination project first, then share/branch from your source file. An alternatively method is via merging, but can be very time consuming and error prone.

Building the New /Production Project

When a change control is complete and all code is now in production from our /test/CCYYYYMMDD project, we need to re-establish our production source environment again. For example, for the STARS application we would see this:

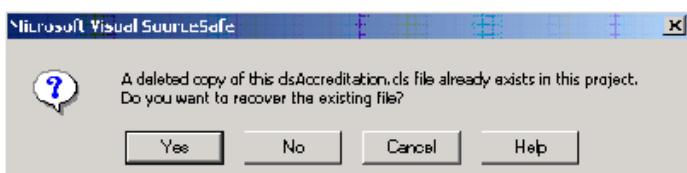
```
/test/CC20031204/STARS/<other projects and altered source taken into prod>
/production
  /STARS (copy all source from /STARS20031204 and merge /test/CC20031204/STARS/ here)
    /STARS20031204 (before the change control)
```

So the /STARS project under production is an image of all source currently on the production server.

To do the copy of /STARS20031204 and paste as /STARS, one would think they could simply share/branch this folder the /production and be prompted to rename? Well, it doesn't work. You need to manually create the /STARS project, then for each subproject folder under /STARS20041204 share/branch it to /STARS. This is extremely inconvenient and does not follow standard GUI interactivity as per other MS products.

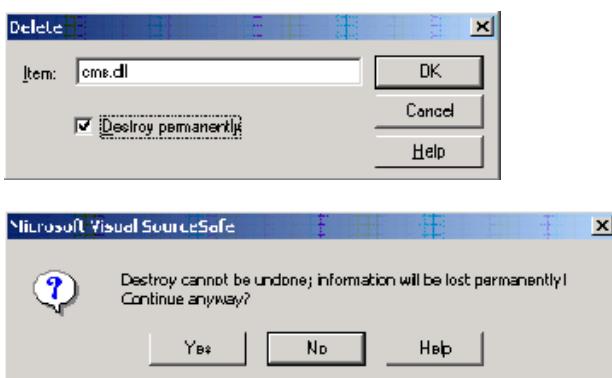
Adding/Removing Project Source Files

If you remove files from a project folder in VSS, you will find that VSS tracks this removal for subsequent recovery, all fine and good. When you add files back into this project folder, you will get something like this:



In this case I am adding 45 files that are part of a single large DLL. There is no "yes all" option, so I need to click on NO forty-five times.

When deleting files, you are prompted with the list, and a small dialog, check the "destroy permanently" option:



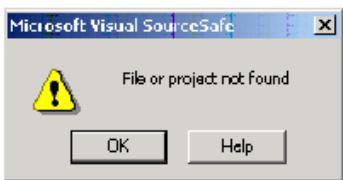
This will prevent the previous message from popping up forty five times but there is no rollback.



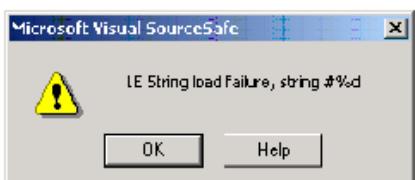
If you delete a range of source files from a project, then check new files in over the top, you may find all history is lost of previous actions against these files. Consequently, always branch from /test back into /production to retain this history. Developers rely on this history and may get very upset if it's lost.

Error on Renaming Projects

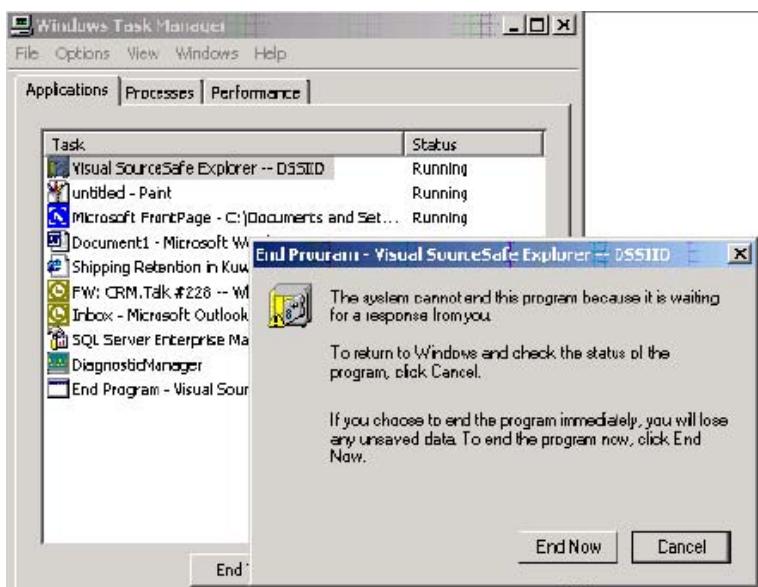
Removing large projects from VSS is an issue. When I rename, say, /production/stars to /product/stars20031204, it sits there for 20+sec then I repeatedly get this message:



then this:



This will carry on for all sub-project folders. If I kill the process via task manager and return to VSS, the project has been successfully renamed and I have no further issues with the project.



Use Labels Where Appropriate

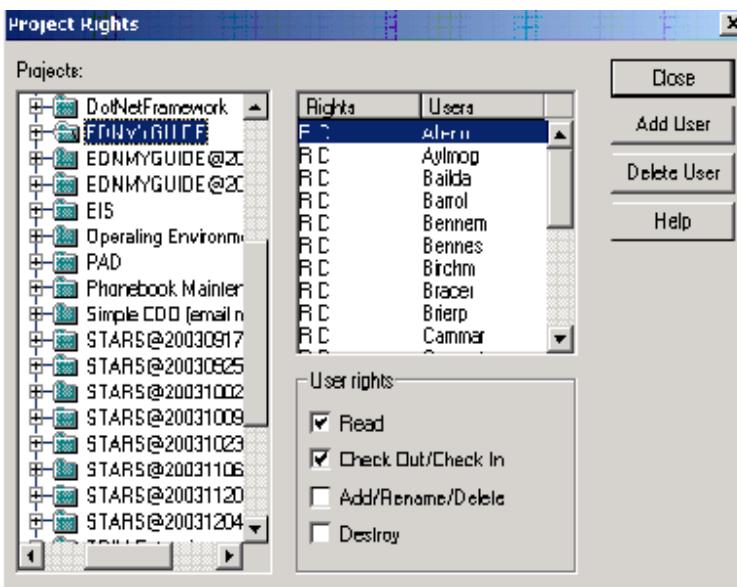
This is not a gripe on VSS, but a feature I encourage. Anytime code is checked-in, VSS will assign it an internal version number. The developers can complement this with labels that can act as textual version numbers to provide better clarity on what a specific version encompasses. It is not unusual to label a single project and all its files, once done, we can get latest version based on this label to retrieve all marked source files with this label. This is done by right clicking on the VSS project—show history—check the labels only check box in the project history dialog shown. A list of labels is shown, click on the label to be retrieved and click get; this will do a get latest version to the projects marked working directory.

The “Guest” User

A very quick one—the guest account cannot be granted check-in/out, add/rename/delete or destroy privileges on VSS project folders. As such, when bulk changing user access unselect the guest account to allow you to bulk change the remainder of the users.

Security Issues in /Production for Branching Files

In the /production project folder, a developer cannot branch files into the /development folder unless they have the check-in/out security option enabled for the /production project. This is a real problem for me. Why? Developers can not check in/out over production source and may also branch incorrectly (i.e. share rather than branch). As shown below, the developer requires read and check in/out for the branch option to be available.



Here are some possible options to get around the issues:

1. Ask developers to email me what files are required for branching into development (far from ideal)
2. Pre-branch from /production back into /development and the developers can sort out what to remove or keep for their next phase of development
3. Give developers the check-in/out access in production and trust that all will be fine and source will be branched correctly and never checked in/out of the production folder directly.
4. Change the way to name /production folders so we always have a “copy” of current production (locked down) and another for developers to branch from.
5. Microsoft upgrades VSS (not likely—come on guys, all other software has seen major change except VSS). That said, the change has been in the VS.Net GUI space as we will see later.

Me? I have opted for 3) for the time being to remain flexible.



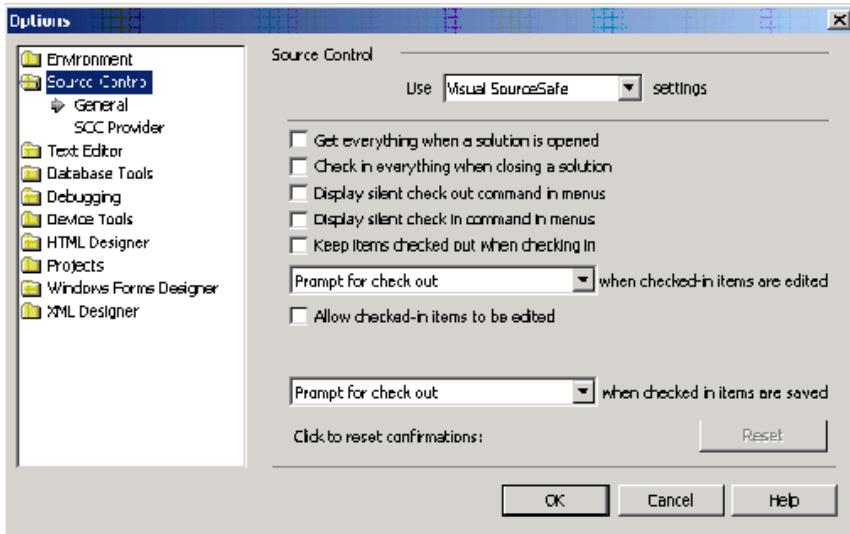
Do not forget that, if using option 3), developers incorrectly share rather than branch code, or check in/out of the /production folder rather than /development you have all the VSS features to resolve the issue, namely the inbuilt versioning of source (i.e. we can rollback), the viewing of this version history, and the fact that in /production we have denied the "destroy" (delete) option.

Welcome to .Net

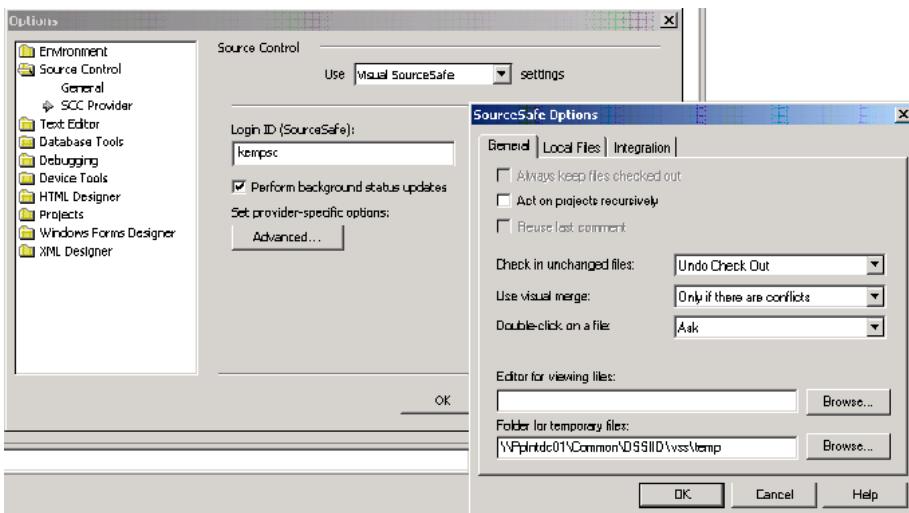
Initial Configuration of Visual Studio .Net

You require Visual Source Safe SP6d or greater for Visual Studio .Net 2003 to complete the following steps. I have found that, under some circumstances VSS 6c will not work, the source control options are shown but are all grayed out (disabled), very strange indeed.

Start VS.Net and select Tools then Options to open the IDE options dialog below. From here we are interested in two key areas in relation to VSS, the source control options:

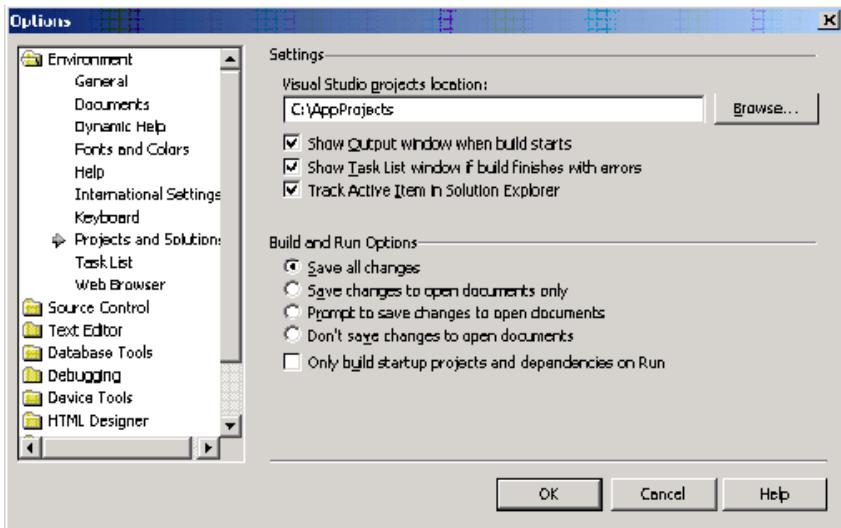


Click on SCC Provider to connect to your source safe database, for example:



From here we configure the VSS database connection string, some core check in/out options and login details. These are automatically populated based on my existing VSS client setup.

Equally important is the projects and solution settings under environment. All developers should map to the same physical location on their development machines, in this example it is c:\AppProjects, I do not like to prefix the company's name or division/team name as, like all things, they tend to change often. The directory will be automatically created.



If you repair your VS.Net installation you may need to re-install VSS 6d as the source safe option may "disappear" from your File menu.

VS.Net Solutions and Projects

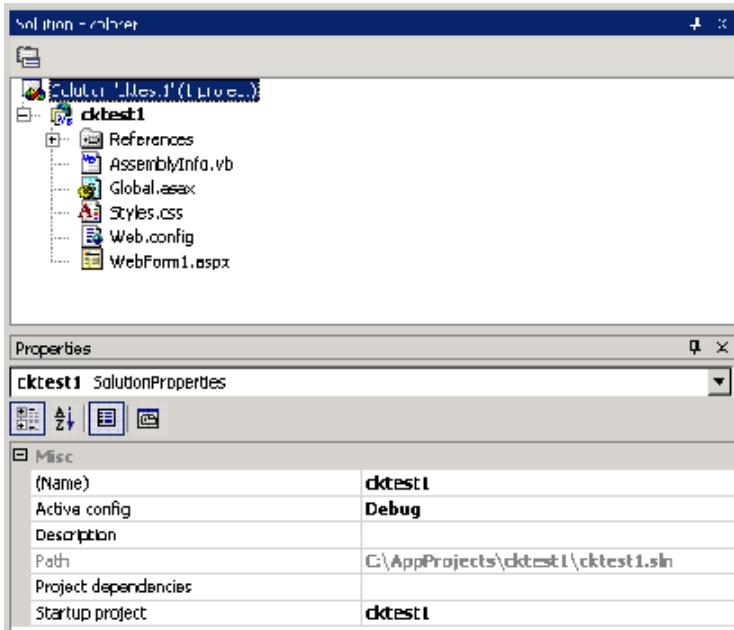
Important Notes Before We Continue

Here are some general rules before we continue on with examples:

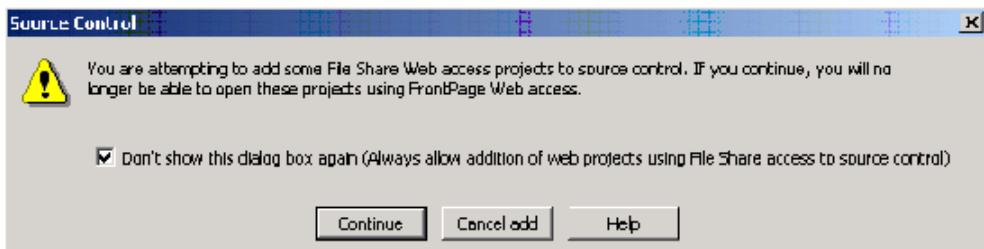
1. Use VS.Net source control options where possible—don't run the VSS GUI, checkout code, then go about opening the source. Use the VS.Net options and the open from source control menu option.
2. Developers need to standardize a local PC directory structure for the checkout of code from source safe, it is not difficult, but means everyone gets together than sorts out the base working directory structure for checked out code. You may find VSS and VS.Net hard coding specific directories that will make life very difficult for you later in the development phase.
3. Keep your project and solutions simple. Avoid multiple solutions for multiple sub-components of an application, keep the structure flat and simple. Only "play" when you (and your team) are sufficiently experienced comfortable with it.

Adding a New (Simple) Solution to Source Control – Example

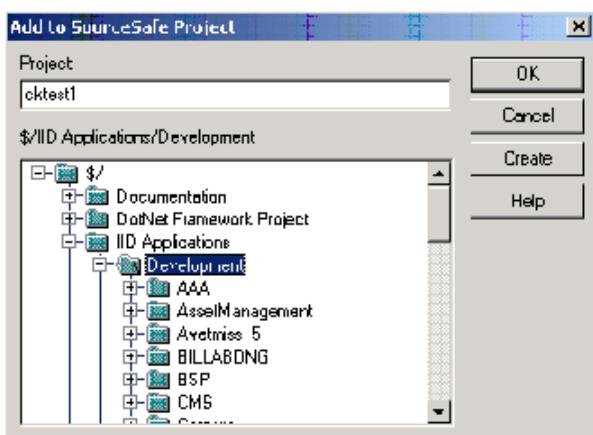
Here is a very simple (default) solution called cktest1. No rocket science here by any means.



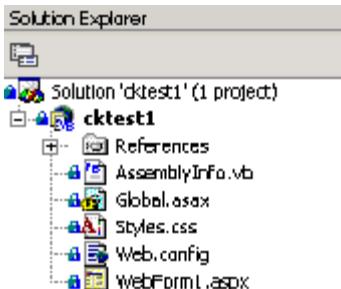
From the file menu, select Source Control and Add to source, you are presented with this warning in relation to Front-Page, if all fine, press continue.



You are prompted for the root location of this solution within VSS. You can name your VSS project folder different to that of the solution, but I do not recommend it. Here we select our /development folder:



The /cktest1 VSS project is then created and the lock icons within VS.Net represent the source control activation against this solution and its project(s).



The VSS structure is a little messy though—we get this by default:

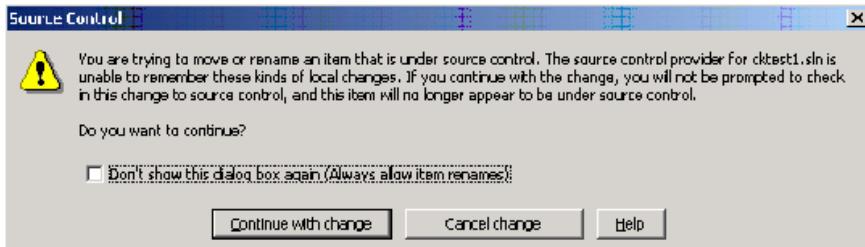


/cktest1 is the *solution*, and /cktest1_1 is the *project*, so lesson learnt - name your solution and projects before hand. Either a prefix or postfix standard is recommend here, for example:

cktest1**Solution** and cktest1**Project**

but does get a little more complicated with many projects.

If you want to change your solution and project names you will get this message:



Cancel this change. If you don't like the VSS standard structure, and prefer a more logical hierarchy like this:

/development/cktest1 (aka. the system name!)

/cktest1**Solution**

/cktest1**Project**

It gets a little tricky.

To do this, check-in all code into VSS and close your solution. Open the VSS GUI, and navigate to your newly created project.

Now share & branch your code from:

/cktest1/cktest1 == to ==> /cktest1/cktest1Solution

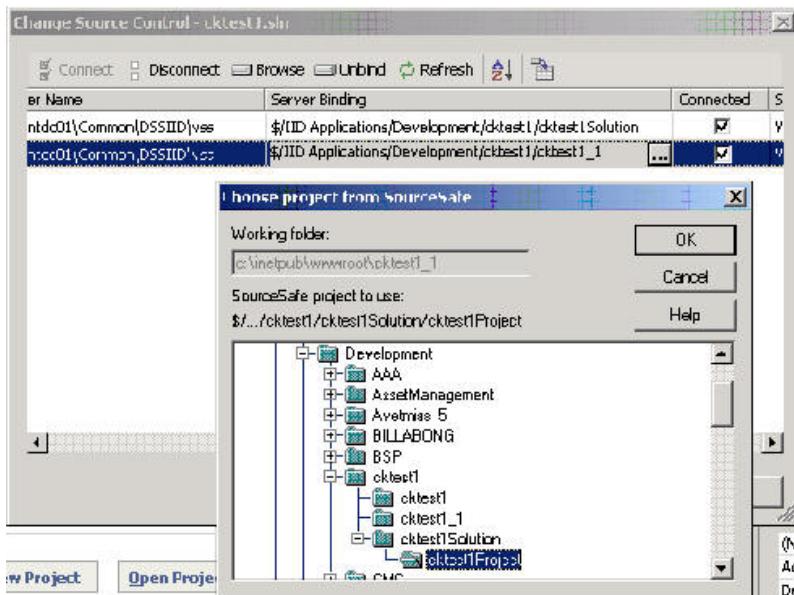
and

/cktest1/cktest1_1 == to ==> /cktest1/cktest1Solution/cktest1Project

Do not remove the old project folders.

Go back to VS.Net and open the solution once again (do not checkout code!).

Select the solution, then select File, Source Control, and pick the Change Source Control option.



Under the server bindings area, we select the new project folders as create previously. The solution will be automatically checked-out to complete the new VSS bindings.

Open the VSS GUI and remove the old /cktest1 and /cktest1_1 projects. I would recommend a get latest version on the root project just in case.

Whether you remain with a flat structure, or re-bind as we have done above, is an issue for the change manager more than anything in terms of best practice.

VSS for the DBA

The examples have presented a complete solution for change control management using VSS.

For the DBA may choose to store scripts in VSS created from development, test and those applied to production. The examples apply in all cases, as scripts are simply files they need to be managed between environments; and VSS is the tool for the job.

The DBA should also consider VSS as a backup utility. The DBA should script all production databases on a monthly basis, and store the scripts within VSS. Using VSS and its merge functionality can be handy in this respect to locate differences between months for example, or retrieve the definitions of lost database objects, constraints, views etc rather than recovering full databases from tape.

The DBA should not forget objects outside of the user database, namely:

- Full text index definitions
- Logins
- Important passwords
- Scripted DTS jobs
- DTS Packages saved as VB files
- Linked Server Definitions
- Publication and Subscription scripts

THEORY AND ESSENTIAL SCRIPTS

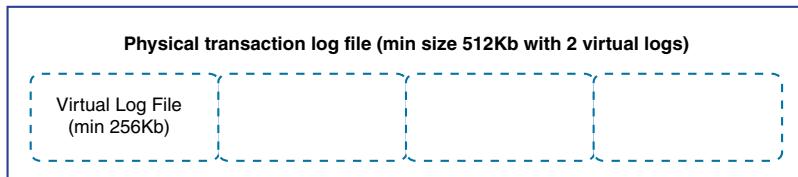
The DBA must have a solid understanding of the instance and the databases. The DBA can then better plan system backup and recovery procedures. This chapter provides the knowledge to better understand your SQL environment. As a reminder, this ebook is not a beginners guide to SQL databases. This chapter provides value added knowledge to those already familiar with installation, setup and configuration of SQL Server 2000.

Undo & Redo Management Architecture

The key component for rollback and redo in SQL Server is the transaction log that is present in each database within the instance. The transaction log is a serial record of all transactions (DML and DDL) executed against the database. It is used to store:

- start of each transaction
- before and after changes made by the transaction
- allocation and de-allocation of pages and extents
- commit and rollback of transactions
- all DDL and DML

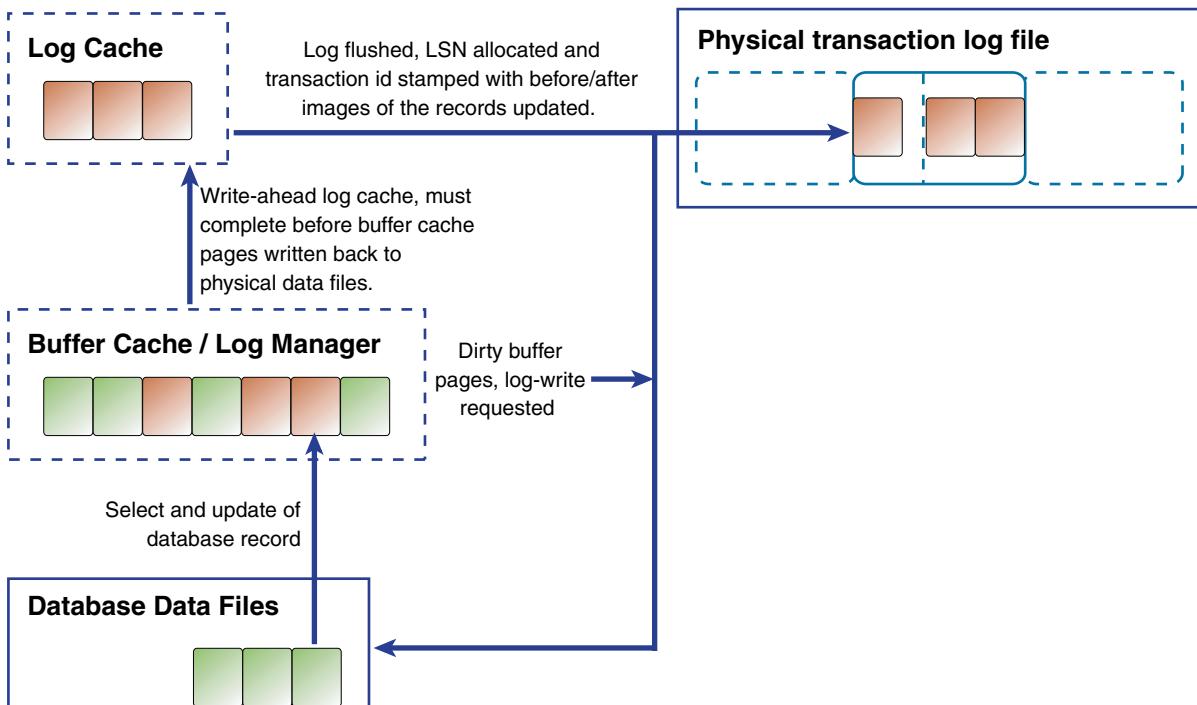
The transaction log itself consists of one or more physical database files. The size of the first must be greater than or equal to 512Kb in size. SQL Server breaks down the physical file into two or more virtual transaction logs. The size of the file and its auto-growth settings will influence the number of virtual logs and their size. The DBA cannot control the number of, or sizing of, virtual logs.



The log-writer thread manages the writing of records to the transaction log and the underlying data/index pages. As pages are requested and read into the buffer cache, changes are sent to the log-cache as a write-ahead operation which log-writer must complete with a write to the transaction log before the committed change is written to the data files as part of a check-point. In terms of an update, the before and after images are written to the log; a delete the before image is written; an insert tracks only the new records (not including many other record entries as mentioned previously as part of the transaction). The log-writer (or log manager) allocates a unique LSN (log sequence number—a 32bit#) to each DML/DDL statement including the transaction ID that links like log entries together.



The log manager (writer) alone does not do all of the writing; the logwriter thread will write pages that the worker threads don't handle.

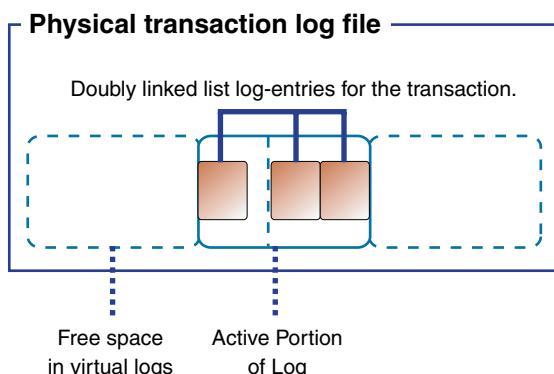


The transaction entries (committed and uncommitted) in the transaction log are doubly linked lists and each linked entry may contain a range of different (sequential) LSN's. The buffer manager guarantees the log entries are written before database changes are written, this is known as write-ahead logging; this is facilitated via the LSN and its mapping to a virtual log. The buffer manager also guarantees the *order* of log page writes.



Every database page has a LSN in its header; this is compared to the log entry LSN to determine if the log entry is a redo entry.

The transaction itself remains in the active portion of the transaction log until it is either committed or rolled back and the checkpoint process successfully completes.

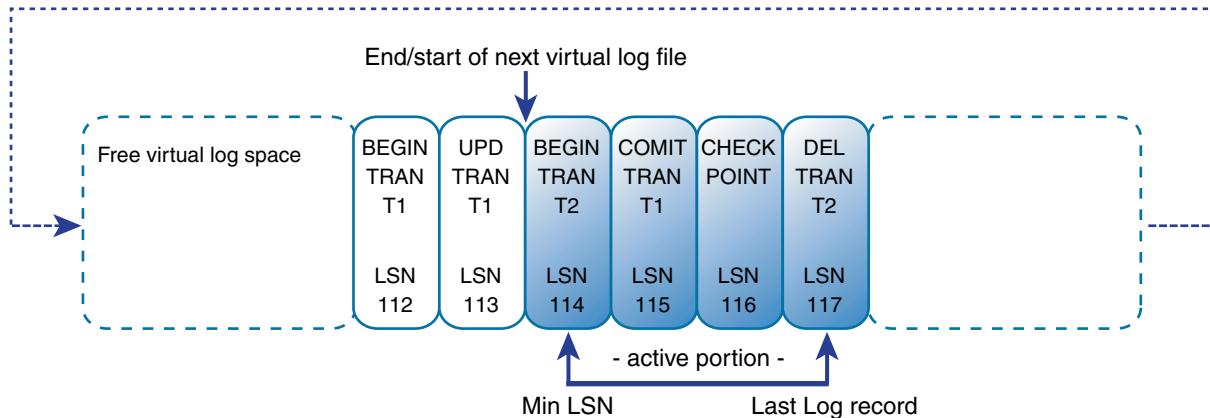


The checkpoint process keeps around 100 pending I/O operations outstanding before it will yield to the User Mode Scheduler (UMS). The performance monitor counter **checkpoint pages/sec** is an effective measure of the duration of checkpoints.

Although not shown above, space is also allocated for each log entry record for rollback purposes; therefore actual space utilization can be significantly more than expected. Microsoft defines the *active* part of the log to be the portion of the log file from the MinLSN to the last written log record (end of logical log).

The MinLSN is the first of possibly many yet uncommitted (or roll backed) transactions.

Cyclic and serial transaction log



The log records for transaction are written to disk before a commit acknowledgement is sent to the client. Even so, the physical write to the data files may have not occurred. Writes to the log are *synchronous* but the writes to data pages are *asynchronous*. The log contains all the necessary information for redo in the event of failure and we don't have to wait for every I/O request to complete. (33)

When the database is using a *full* or *bulk-logged* recovery model, the non-active portion of the log will only become "free" (can be overwritten) when a full backup or transaction log backup is executed. This ensures that recovery is possible if need be via the backup and the DBMS can happily continue and overwrite the now free log space. If the database is using the simple recovery model at a database checkpoint, any committed (and check pointed) or rollback (and checkpointed) transaction's log space will become immediately free for other transactions to use. Therefore, point in time recovery is impossible.

The checkpoint process is key to completing the committed transactions and writing the dirty buffers back to disk. In relation to the transaction log, a checkpoint will:

1. write a log entry for the start of the checkpoint
2. write the start LSN for the checkpoint chain to the database for subsequent recovery on instance failure
3. write a list of active (outstanding) transactions to the log
4. write all dirty log and data pages to disk for all transactions
5. writes a log file record marking the end of the checkpoint

The checkpoint will occur:

1. On issue of the CHECKPOINT or ALTER DATABASE statement
2. On instance shutdown (SHUTDOWN statement)
3. On SQL service shutdown
4. Automatic checkpointing
 - a) DBMS calculate timing based on the recovery interval setting
 - b) “Fullness” of the transaction log and number of transactions
 - c) Based on timing set with recovery internal parameter
 - d) Database using simple recovery mode?
 - If becomes 70% full
 - Based on recovery interval parameter



Dirty page flushing is performed by the lazywriter thread. A commit does not trigger an immediate checkpoint.

You can see the checkpoint operation in action via the ::fn_virtualfilestats (SQL 2k) system function. The function has two parameters, the database ID and the file ID. The statistics returned are cumulative. The test is simple enough, here we run insert and update DML, we call :fn_virtualfilestats between each operation and at the end force a checkpoint, here is what we see:

DbId	FileId	TimeStamp	NumberReads	NumberWrites	BytesRead	BytesWritten
7	2	1996132578	317	154	17096704	2127360
7	3	1996132578	30	42	819200	540672
7	2	1996132578	317	156	17096704	2128384 [1024 diff]
7	3	1996132578	30	42	819200	540672 [0 diff]
7	2	1996132578	317	158	17096704	2129408 [1024 diff]
7	3	1996132578	30	42	819200	540672 [0 diff]
7	2	1996132593	317	160	17096704	2130432 [1024 diff]
7	3	1996132593	30	42	819200	540672 [0 diff]
7	2	1996132593	317	162	17096704	2131456 [1024 diff]
7	3	1996132593	30	42	819200	540672 [0 diff]
7	2	1996132625	317	165	17096704	2140672 [9216 checkpt]
7	3	1996132625	30	43	819200	548864 [8192 written]

File ID 2 = Log File

File ID 3 = Data File

Each color band represents new DML operations being performed. The last operation we perform is the CHECKPOINT. Here we see the BytesWritten increase as the logs are forced to flush to disk (file ID 3).

Running the LOG command you will see the checkpoint operations:

```
DBCC LOG(3, TYPE=-1)
```

where 3 is the database ID (*select name, dbid from master..sysdatabases order by 1*)

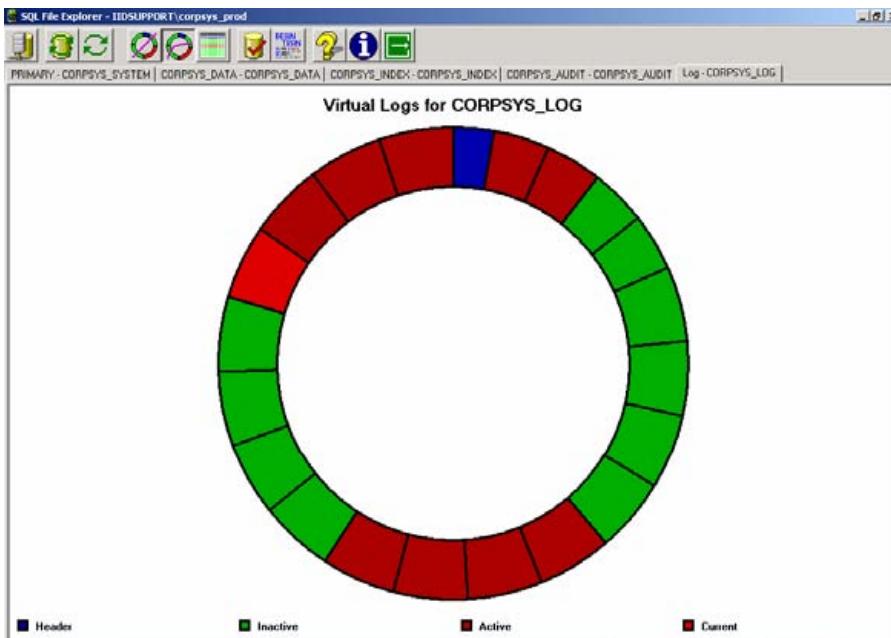
	Current LSN	Operation	Context	Transaction ID
1	00000005:0000001a:0001	LOP_BEGIN_CKPT	LCX_NULL	0000:00000000
1	00000005:0000001b:0001	LOP_END_CKPT	LCX_NULL	0000:00000000

Apart from the recovery interval and recovery mode parameters, the DBA has little control of checkpointing. It can be forced via the CHECKPOINT statement if need be and I recommend this before each *backup log* statement is issued.

To drill into the virtual logs and the position, status, size, MinLSN within the transaction log files, use the commands:

```
dbcc loginfo  
dbcc log(<db#>, TYPE=-1)
```

The 3rd party tool “SQL File Explorer” includes a simple but effective GUI view of your transaction log file and its virtual logs. This can be very handy when resolving log file shrink issues.



..and a textual view..

The screenshot shows a window titled "SQL File Explorer - IIDSUPPORT.corpsys_prod". The main pane displays a table with the following columns: File ID, Offset, Size, Sequence, and Status. The table lists multiple entries for file ID 2, representing log files. The status column indicates various states such as Active, Inactive, and Curwart.

File ID	Offset	Size	Sequence	Status
2	0	8192		N/A Header
2	8192	253952	210616	Active
2	262144	253952	210615	Active
2	516296	253952	210614	Inactive
2	770340	270528	210613	Inactive
2	1048976	13107200	0	Inactive
2	14195776	13107200	0	Inactive
2	27365296	13107200	0	Inactive
2	40370176	13107200	0	Inactive
2	53477376	13107200	210620	Active
2	66584576	13107200	210619	Active
2	79691776	13107200	210618	Active
2	92798976	13107200	210617	Active
2	105906176	13107200	0	Inactive
2	119013376	13107200	0	Inactive
2	132120576	13107200	0	Inactive
2	145227776	13107200	0	Inactive
2	158334976	13107200	0	Inactive
2	171442176	13107200	0	Inactive
2	184549376	13107200	0	Inactive
2	197656576	13107200	210621	Curwart



To get even more details about log entries, consider the command:

```
select * from ::fn_dblog(null, null)
```

The process of recovery involves both:

1. Redo (rolling forward or repeating history to time of the crash); and
2. Undo (rolling back or undoing un-committed transactions) operations.



In SQL Server 2k, the database is only available after undo complete (not including when the STANDBY restore clause is used). In SQL Server Yukon, the database is available when undo begins! Providing faster restore times.

The redo operation is a check, we are asking “for each redo log entry, check the physical files and see if they change has already been applied”, if not, the change is applied via this log entry. The undo operation requires the removal of changes. The running of these tasks is based upon the last checkpoint record in the transaction log. (33)

The management of transactions is a complex tasks, and is dealt with not but the buffer manager, but via the *transaction manager* within the SQL engine. Its task includes:

1. isolation level lock coordination with the lock manager, namely when locks can be released to protect the isolation level requested; and
2. management of nested and distributed transactions and their boundaries in which they span; this includes the coordination with the MSDTC service using RPC calls. The manager also persists user defined savepoints within transactions.



The transaction log is a write-ahead log. Log writes are synchronous and single threaded. Actual DB writes are asynchronous, multi-threaded and as shown in the diagrams, optimistic. Also note that compressed drives and their associated algorithms disable the write-ahead logging (WAL) protocol, and can effectively stall checkpoints and the timing of calls.

Audit the SQL Server Instance

Walking into a new server environment is never easy, and quickly understanding the database running on your instances at a high level is a basic task that needs to be completed quickly and accurately. The DBA should the following—all are important in some way to systems recovery (DR) and system troubleshooting in general:

1. Instance version and clustered node information (if applicable)
2. Base instance properties covering switches, memory, security settings etc.
3. Service startup accounts, including the SQL instance service, SQL Agent and the SQL Agent proxy account (if used)
4. Select snapshot of configuration settings
5. The existing of performance counters and statistics from a select few
6. List of sysadmin and dbowner users/logins
7. List of databases, options set, total size and current transaction log properties, if database has no users other than DBO:
 - a) Files used and their drive mappings
 - b) Count of non-system tables with no indexes
 - c) Count of non-system tables with no statistics
 - d) List of Duplicate indexes
 - e) Count of non-system tables with no clustered index
 - f) Tables with instead-of triggers
 - g) List of schema bound views
 - h) Count of procedures, views, functions that are encrypted
 - i) List of any user defined data types
 - j) List of PINNED tables
 - k) For each user, a count of objects owned by them rather than DBO
8. Last database full backup times—see section on backups.
9. Publications currently in existence
10. Subscriptions currently in existence
11. Startup stored procedures in effect
12. Currently running traces and flags set on startup
13. List of linked servers and their properties
14. If the database is currently experiencing blocking

15. Login accounts with no password, and those with sysadmin access
16. Non standard users in the master and msdb databases (consider model as well to determine if any special “generic” user exists when other databases are created), those users with xp_cmdshell access
17. Current free space on local disks
18. Full text index statistics/summary

This e-book provides many of these answers based on the problem at hand. The internet is a huge source of scripts, I have no single script but use many crafted by fellow DBA's. Rather than deal with the issues of copyright, I charge you with the task of accumulating and adapting suitable scripts. Feel free to email me if you require assistance.

As a general guide for systems recovery, collect the following:

```
DBCC MEMORYSTATUS
SELECT @@VERSION
exec master..xp_msver
exec sp_configure
select * from master..syslogins
select * from master..sysaltfiles
select * from master..sysdatabases
select * from master..sysdevices
select * from master..sysconfigures
select * from master..sysservers
select * from master..sysremotelogins
select * from master..sysfilegroups
select * from master..sysfiles
select * from master..sysfiles1
sp_MSforeachtable and sp_MSforeachdb stored procedures
execute master..sp_helpsort
execute master..sp_helpdb -- for every database within the instance.
```

To determine the actual “edition” of the SQL instance, search the internet for a site listing the major and minor releases. A very good one can be found at:

<http://www.krell-software.com/mssql-builds.htm>

The SERVERPROPERTY command is a handy one, giving you the ability to ping small but important information from the instance. For example:

```
SELECT SERVERPROPERTY('ISCLUSTERED')
SELECT SERVERPROPERTY('COLLATION')
SELECT SERVERPROPERTY('EDITION')
SELECT SERVERPROPERTY('ISFULLTEXTINSTALLED')
SELECT SERVERPROPERTY('ISINTEGRATEDSECURITYONLY')
SELECT SERVERPROPERTY('ISSINGLEUSER')
SELECT SERVERPROPERTY('NUMLICENSES')
```

The database level equivalent is DATABASEPROPERTY and is covered well in the BOL.

To locate the install path (undocumented command):

```
DECLARE @sql SYSNAME  
DECLARE @data SYSNAME  
  
EXEC master..sp_msget_setup_paths @sql OUTPUT, @data OUTPUT  
  
SELECT @sql, @data
```

Do not forget the SQLDIAG.EXE command to dump to into a single ASCII file core system and SQL configuration information. I highly recommend that you run this daily to assist with systems recovery:

```
cd "C:\Microsoft SQL Server\MSSQL$CORPSYS\binn"  
sqldiag -icorpsys -oc:\sqldiag_corpsys.txt --corpsys is the named instance
```

Goto the binn dir for the instance and run sqldiag. Remove the `-i<myinstancename>` as necessary. I always run named instances of sql server (a personal preference more than anything—especially if you want to run an older DBMS version of SQL on the box), so the default instance will be:

```
C:\Program Files\Microsoft SQL Server\MSSQL\Binn>sqldiag -oc:\sqldiag.txt
```

Meta Data Functions

It is easy to forget about the SQL Server meta data functions, all of which are essential to making life a damn site easier with more complex scripts. I recommend you spend some time exploring the following at a minimum:

FUNCTION NAME	PURPOSE
DB_NAME	Given the database ID from the master..sysdatabases table, returns the name of the database.
DB_ID	As above but you pass the name and the ID is returned.
FILE_NAME	Given the fileid number from sysfiles for the current database, returns the logical name of the file. The value is equivalent to the <i>name</i> column in sysfiles. Be aware that databases in the instance can share the same ID number, so file_name(1) can work equally well in any database, this ID value is not unique for the entire instance.
FILEGROUP_NAME	Pass in the <i>groupid</i> value as represented in sysfilegroups for each database and returns the <i>groupname</i> column. Remember that we can have <i>primary</i> and <i>user defined</i> filegroups, transaction logs do not have a group with a maximum of 256 filegroups per database.
FILEGROUOPROPERTY	We can use the previous command as the first input into this routine, the next parameter includes three basic options, IsReadOnly, IsUserDefinedFG, IsDefault, this undoubtedly change with future releases on SQL Server.

FUNCTION NAME	PURPOSE
FILEPROPERTY	Will return file specific property information, we can use the FILE_NAME function as the first input, the next parameter can be one of the following: IsReadOnly, IsPrimaryFile, IsLogFile, SpaceUsed. The space is in pages.
FULLTEXTSERVICEPROPERTY	Returns the MSSEARCH (Microsoft Search Service) service properties specific to SQL Server full text engine integration.
OBJECT_NAME	Based on the current database of the connection

Listing SQL Server Instances

To get a list of SQL Instances on the network, consider this command:

- isql -L
- OR -
- osql -L

There are a range of third party alternatives available on the internet, such as **sqlping.exe** from www.sqlsecurity.com. Also check my website under hints/tips for a bunch of SQL tools from a variety of authors.

Information Schema Views

It is not uncommon for system tables to change between releases; so significant can be the change that impacts on existing scripts can result in complete re-writes. To ease the pain a little, a range of what I call Metadata++ views are available that tend to be more consistent and additive, rather than taken away.

The metadata views are referred to as the *information_schema*.

INFORMATION SCHEMA	COLUMN SUMMARY
information_schema.tables	Catalog (database), Owner, Object name, Object type (includes views)
information_schema.views	Catalog (database), Owner, Object name, View Source (if not encrypted), Check Option?, Is Updatable?
information_schema.columns	Catalog (database), Owner, Table name, Col Name, Col Position, Col Default, Is Nullable, Data Type, Char Min Len, Char Octal Len, Numeric Prec <etc>

INFORMATION SCHEMA	COLUMN SUMMARY
information_schema.schema	Database Name, Schema Name, Schema Owner, Def Char Set Catalog, Def Char Set Schema, Def Char Set Name
information_schema.referential_constraints	Catalog (database), Owner, Constraint Name, Unique Constraint Database, Unique Constraint Owner, Unique Constraint Name, Update Rule, Delete Rule

Note that the views return data for the currently active (set) database. To query another database use the db-name(dot) prefix:

```
select * from mydb.information_schema.tables
```

There are no special security restrictions for the views and they are accessible through the public role. Be aware though that the schema views will only return data to which the user has access to. This is determined within the views themselves via the *permissions()* function.

To view the code of any of the views:

```
use master
exec sp_helptext 'information_schema.Referential_Constraints'
```

As you have probably noticed, the views are very much DDL orientated and lack items such as *databases* or *linked servers* for example; these would prove very handy rather than having to query system tables. (34)

Database, File and File Group Information

Extracting Basic Database Information

Use the system stored procedure found in the master database:

```
exec sp_helpdb mydatabase
```

Determining Database Status Programmatically

The `sysdatabases` table in the master database includes the `status` column of type int(eger). This is an essential column for many scripts to use to determine the state of instance databases before continuing on with a specific recovery scenario etc. Here is an example:

```
alter database pubs set read_only
go
select name, 'DB is Read Only'
from master..sysdatabases
where status & 0x400 = 1024
go
```

where 0x400 is equivalent to 1000000000 binary, (BOL tells us that bit position 11 (or 1024 decimal) tells us the database is in read-only mode). Including more than one option is a simple matter of addition; the DBA should recognize that 0x (zero x) is the prefix for a hexadecimal value.

STATUS	BINARY & HEX CODES	MEANING
32	0x20 100000	Loading
64	0x40 1000000	Pre-Recovery
128	0x80 10000000	Recovering
256	0x100 100000000	Not Recovered
512	0x200 1000000000	Offline
1024	0x400 10000000000	Read Only
2048	0x800 100000000000	DBO use only
4096	0x1000 1000000000000	Single User
32768	0x8000 1000000000000000	Emergency Mode

Common Status Code Bits.

An alternate method is a call to the DATABASEPROPERTY function and is much more understandable:

```
select name, 'DB is Read Only'  
from master..sysdatabases  
where DATABASEPROPERTY(name, N'IsReadOnly') = 1  
go
```

The DATABASEPROPERTYEX function is a drill-through function, providing more specific property information about the database itself. These functions are called *meta-data functions*.



Multiple bits can be set at any one time. The list above is not the definitive set of values, but are the most important to recognize in terms of Backup and Recovery.

Using O/ISQL

The osql or isql command line routines are very handy and are well worth exploring. For a majority of DBA work though, the command can go unused, but occasionally they prove essential, especially for complex scripts where dumping data to flat files is required. For example:

```
osql -E -h-1 -w 158 -o test.txt -Q "SET NOCOUNT ON SELECT name FROM  
master..sysdatabases WHERE name <> 'model'"
```

or run an external SQL file:

```
exec master..xp_cmdshell 'osql -S MySQLServer -U sa -P -  
ic:c:\dbscripts\myscript.sql'
```

Rather than embedding the username/password, especially within stored procedures via xp_cmdshell, consider -E for integrated login via the service user account. The SQL script may contain multiple statements, be they in a global transaction or not.



to represent a TAB delimiter for output files, use -s “ “, the space represents a TAB character.

Taking this further, a Mike Labosh at www.devdex.com shows how to process many .sql scripts in a directory using a single batch file:

RunScripts.bat

```
@ECHO OFF IF "%1"=="" GOTO Syntax FOR %f IN (%1\*.sql) DO osql -i %f <--- also add  
other switches for osql.exe GOTO End :Syntax ECHO Please specify a folder like this:  
ECHO RunScripts c:\scripts ECHO to run all the SQL scripts in that folder :End ECHO.
```

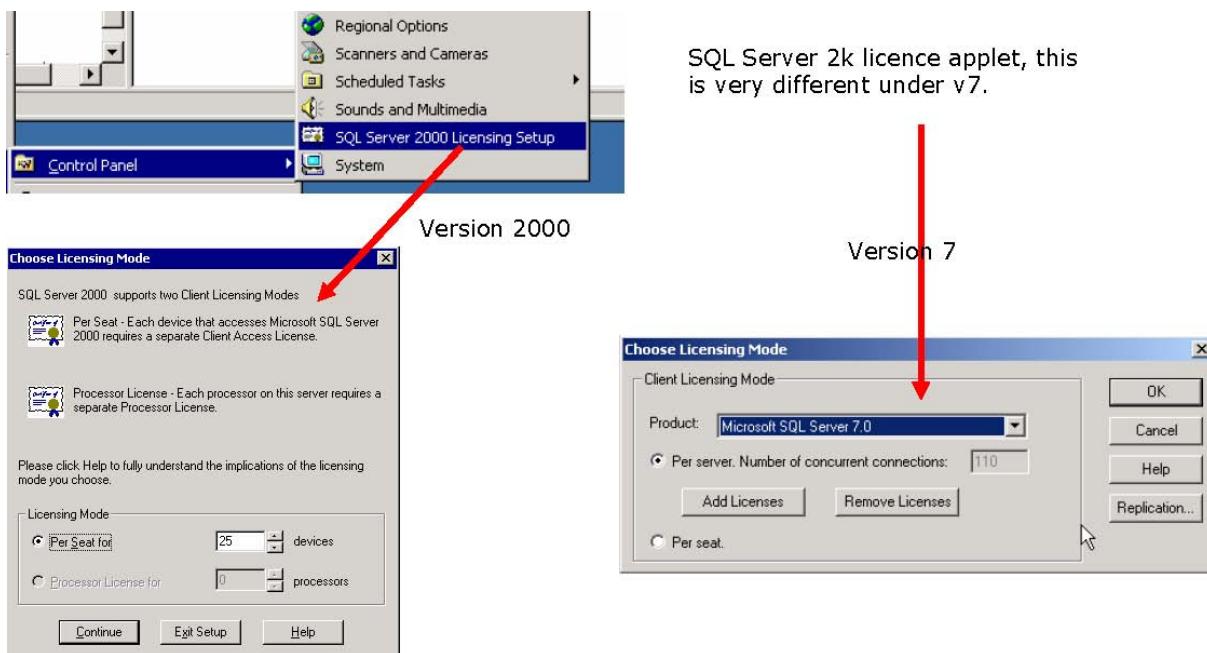
Retrieving Licensing Information

To locate information about the existing licensing scheme, run the following within Query Analyzer:

```
select SERVERPROPERTY ('LicenseType')
select SERVERPROPERTY('NumLicenses')
```

where 'NumLicenses' is not applicable for the per CPU scheme. If you are running the Developer Edition, 'LicenseType' returns "disabled" and 'NumLicenses' is NULL.

The License Manager program (control-panel -> administrative tools) cannot display information related to per-processor licensing for SQL Server (Q306247). To get around this issue the installation will create a new administrative tool program to track such information as shown in the screen shots below.



The Settings → Control panel → Licensing → shows only server licensing for SQL Server 2000 installations (unlike v7). The applet above is used instead. Notice that you cannot switch to a *per-processor* license; you must re-install the instance to do this.



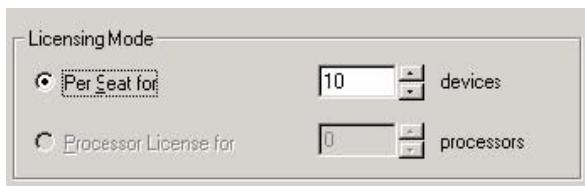
Unless specified in your license contract or SQL Server version on installation, the installed instance will not automatically expire or end.

Alter Licensing Mode After Install?

Once the licensing mode is set on instance installation, that's it. You cannot change it back. In control panel we have the icon:



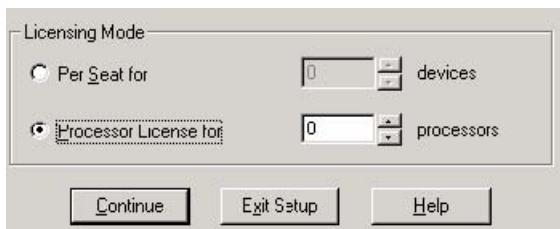
This will allow you administer your licenses, but you will see one of the options is grayed out, such as:



To get around this go to the following registry key entry:

Name	Type	Data
(Default)	REG_SZ	(value not set)
ConcurrencyLimit	REG_DWORD	0x0000000a (10)
DisplayName	REG_SZ	SQL Server 2000
FamilyDisplayName	REG_SZ	Microsoft SQL Server
FlipAllow	REG_DWORD	0x00000001 (1)
Mode	REG_DWORD	0x00000001 (1)

Change the *mode* to 1 (default is zero) and you get both options:



Microsoft may not support you if this method is used.

Allowing Writes to System Tables

Use the following code whilst logged in as SA or with sysadmin privileges:

```
exec sp_configure "allow updates", 1 -- 0 (zero) to disable
go
reconfigure with override -- force immediate change in config item
```

Updating system tables directly is not supported by Microsoft, but there are some occasions where it is required. In terms of DR, it is important to understand the process if you need to apply changes.

Count Rows & Object Space Usage

The DBA has a number of methods here to return a table rowcount and space utilization details. Ensure your scripts return table objects and not schema bound views or views themselves:

```
OBJECTPROPERTY (<object-id>, 'IsUserTable') = 1
```

If you want to run a SQL command for each table in the database, consider the command *master..sp_MSforeachtable*, for example:

```
exec sp_MSforeachtable @command1="select count(*) from ?"
```

The only other MS options are *master..sp_MSforeachdb* or *master..sp_MSforeach_worker*. It is a pity we have no others for views, functions, stored procedures etc. Search Google and you will find many examples:

www.sqlservercentral.com/columnists/bknight/sp_msforeachworker.asp

Do note that the MS_ @command1 parameter itself can accept multiple physical commands. For example this is valid:

```
@command1="declare @eg int print '?' select count(*) from ?"
```

- OR -

```
@command1="declare @name varchar(50) set @name=PARSENAME('?', 1) DBCC UPDATEUSAGE(0,
@name) WITH NO_INFOMSGS select count(*) from ?"
```

Here are some methods of row counting and returning object space usage:

METHOD	GENERAL NOTES
<pre>select count(*) from mytable</pre>	Will not return total number of bytes used by the table. IO intensive operation in terms of table and/or index scanning.
<pre>SELECT object_name(id) ,rowcnt ,dpages * 8 FROM mydb..sysindexes WHERE indid IN (1,0) AND OBJECTPROPERTY(id, 'IsUserTable') = 1 - OR - SELECT o.name, i.[rows], i.dpages * 8 FROM sysobjects o INNER JOIN sysindexes i ON o.id = i.id WHERE (o.type = 'u') AND (i.indid = 1) and o.name not like 'dt%' ORDER BY o.name</pre>	<p>Patch or version changes in DBMS can break script. Returns row count and bytes/space used. Does not factor in multiple indexes.</p> <p>May not be 100% accurate (based on last statistics update).</p> <p>Consider DBCC UPDATEUSAGE beforehand. The indid column value may be zero (no clustered indexes) or one (clustered); note that 0 and 1 will not exist together.</p>
<pre>exec sp_spaceused 'mytable'</pre>	Returns data and index space usage. Will auto-update the sysindexes table. Will scan each data block, therefore can be slow on larger tables—use the @updateusage parameter set to 'false' to skip this scan. Prefix with master..., and the routine will search this DB for the object name passed in.
<pre>dbcc checktable('mytable')</pre>	Performs a physical data integrity check on tables or indexed views; CHECKALLOC is more thorough in terms of all allocation structure validation. Slow and DBMS engine intensive. Will return structural error information with the table (if any), along with a row and pages used count. Can repair errors (see BOL for options). Will validate all indexes unless specified.
<pre>exec sp_msforeachtable @command1=@command1="declare @name varchar(50) set @name=parename('?', 1) dbcc updateusage(0, @name) select count(*) from ?"</pre>	Another example using the msforeachtable routine (undocumented system routine).

Space and Memory Usage

To get the current database size along with data/index/reserved space:

```
use mydb
exec sp_spaceused
```

- or just the basic file information -

```
select fileid, size from sysfiles
```

- or via FILEPROPERTY -

```
USE master go SELECT FILEPROPERTY('master', 'SpaceUsed')
```

To retrieve fixed disk space usage:

```
exec xp_fixeddrives
```

To get the space used on a table:

```
exec sp_spaceused largerow, @updateusage = true
```

To retrieve transaction log space usage:

```
dbcc sqlperf (logspace)
```

To determine amount of free space for a database, consider the following SQL:

```
select name, sum(size), sum(maxsize) from mydb..sysfiles where status & 64 = 0 group by name -- size is in 8Kb pages  
select sum(reserved) from mydb..sysindexes where indid in (0,1,255)
```



The DBCC UPDATEUSAGE command should be run before sp_spaceused command it runs to avoid inconsistencies with the reported figures.

Black Box Tracing

Microsoft Support recommends the black box trace when dealing with instance lockups and other strange DBMS errors that are difficult to trace. The trace itself will create the files *blackbox.trc*, *blackbox_0n.trc* (switches every 5Mb). These return critical system error messages. The trace is activated via:

```
declare @traceID int  
exec sp_trace_create @traceID OUTPUT, 8 -- Create the trace  
exec sp_trace_setstatus @traceID, 1 -- Start the trace
```

and produces this file:

blackbox 0 KB SQL Profiler - trace ... 23/07/2002 10:37 PM

If you want to start the trace every time SQL Server starts, then place the code into a stored procedure within the *master* database and set the option:

```
exec sp_procoption 'mystoredprocnamehere', 'startup', true
```

As a general rule, do not unless you have a specific need. To verify the trace:

```
USE master  
GO  
SELECT * FROM ::fn_trace_getinfo(1)  
GO
```

traceid	property	value
1	1	8
1	2	C:\Program Files\Microsoft SQL Server\MSSQL\$MY2NDINSTANCE\data\blackbox
1	3	5
1	4	NULL

Do not run the trace as a regular enabled trace as it may degrade system performance.

Here is a more complete script to startup the blackbox trace. It includes logging messages to the error log.

```
use master
go
CREATE PROC dbo.sp_blackbox
AS

declare @traceID int
declare @errid int
declare @logmessage varchar(255)

exec @errid = sp_trace_create @traceID OUTPUT, 8      -- Create the
TRACE_PRODUCE_BLACKBOX
exec sp_trace_setstatus @traceID, 1                  -- Start the trace

if @errid <> 0 begin
    set @logmessage = 'Startup of Black box trace failed - sp_blackbox -
error code - '
    + cast(@errid as varchar)
    exec xp_logevent 60000, @logmessage, ERROR
end
else begin
    set @logmessage = 'Startup of Black box trace success - trace id# - ' +
cast(@traceID as varchar)
    exec xp_logevent 60000, @logmessage, INFORMATIONAL
end
GO
exec sp_procoption N'sp_blackbox', N'startup', N'true'
GO
```

Date	Source	Message
2003-05-12 14:25:40.09	spid51	Using 'xpstar.dll' version '2000.80.194' to execute extended stored procedure '
2003-05-12 14:25:31.34	spid51	Startup of Black box trace success - trace id# - 1.
2003-05-12 14:25:31.34	spid51	Error: 60000, Severity: 10, State: 1
2003-05-12 14:25:31.32	spid51	Using 'xplog70.dll' version '2000.80.194' to execute extended stored procedure
2003-05-12 14:25:30.92	spid3	Launched startup procedure 'sp_blackbox'
2003-05-12 14:25:30.81	spid3	Recovery complete.
2003-05-12 14:25:30.76	spid5	Starting up database 'tempdb'.

To stop the trace:

```
declare @traceID int
set @traceID = 1
exec sp_trace_setstatus @traceID, 0
```

If you are really keen, run Process Explorer from sysinternals.com and look through the handles for the sqlserver process. It will list the black box trace file.

To read a trace file, the easiest approach is to run the Profiler GUI. Here is our example file showing commands running at the time of the crash:

EventClass	TextData	DatabaseID	NTUserName	NTDomainName	Host
SQL:BatchStarting	sp_enumererrorlogs	1	kempsc	TRAINING	PC-J
SQL:BatchStarting	select * from master..sysobjects	1			PC-J
SQL:BatchStarting	select c.name, user_name(o.uid), o.crddate, o.id, ISNU...	1	kempsc	TRAINING	PC-J
SQL:BatchStarting	sp_enumererrorlogs	1	kempsc	TRAINING	PC-J
SQL:BatchStarting	sp_readerrorlog	1	kempsc	TRAINING	PC-J
Exception	Error: 623, Severity: 24, State: 2	1	kempsc	TRAINING	PC-J
Exception	Error: 623, Severity: 24, State: 2	1	kempsc	TRAINING	PC-J
SQL:BatchStarting	sp_readerrorlog 1	1	kempsc	TRAINING	PC-J
Exception	Error: 623, Severity: 24, State: 2	1	kempsc	TRAINING	PC-J
Exception	Error: 623, Severity: 24, State: 2	1	kempsc	TRAINING	PC-J
SQL:BatchStarting	select @@trancount	1			PC-J
Exception	Error: 5001, Severity: 21, State: 4	1			
Exception	Error: 623, Severity: 24, State: 2	1			
Exception	Error: 5001, Severity: 21, State: 4	2			
Exception	Error: 623, Severity: 24, State: 2	2			

Be very careful with re-starting your instance after a crash. The instance will overwrite the file if you leave the default filename. In your stored procedure, write some smarter code to better control the filename, for example:

```
set @v_filename = 'BlackTrace_' + convert(varchar(20),getdate(), 112) + '_' +
convert(varchar(20), getdate(), 108)
```

giving you files like:

BlackTrace_20030512_150000.trc

If you want to create a table from the trace file, rather than opening it via profiler, the use the command ::fn_trace_gettable():

```
SELECT *
INTO myTraceResults
FROM ::fn_trace_gettable('c:\sqltrace\mytrace_20040101.trc', default)
```

If the single trace created many files (split by size), and this if the first then the *default* parameter will ensure ALL files are loaded into the table (very handy feature).

Scan Error Log for Messages?

Our friends at Microsoft released a great support document 115519, “INF: How to Scan SQL Errorlog or DBCC Output for Errors”, namely with the findstr DOS command.

Database Last Restored and From Where?

When you select properties of a database within an instance, it pops up a window with a range of tabs. The *General* tab provides a high level summary of the database, including the last fullback and transaction log backup. But for development/test servers, the question is often posed, “where and when was this database restored from?”

From a backup perspective, the table *msdb..backupset* stores the dates used within EM, namely the column *backup_finished_date* where *type* =’D’ for full backups and ‘L’ for transaction logs.

To retrieve restoration history, query the table *msdb..restorehistory*:

```
select
    destination_database_name as DBName,
    user_name as ByWhom,
    max(restore_date) as DateRestored
from
    msdb..restorehistory
where
    destination_database_name = 'mydb'
and
    restore_type = 'D' -- full
group by
    destination_database_name, user_name
```

Join this query to *msdb..restorefile* over the *restore_history_id* column to retrieve the physical file name for each database file restored.

If you want to get the backup files dump location and name from where the restore occurred, then look at the *msdb..backup** tables. The *restorehistory* table column *backup_set_id* joins to *msdb..backupset* and is the key for locating the necessary information. This table is an important one as it stores all the essential information related to the database restored, namely its initial create date, last LSN’s etc. This restored history is the main reason why many prefer EM for restoration to graphically view this meta data.

Also consider the command:

```
restore headeronly from disk='c:\mydb.bak'
```

to view backup file header information.

What Stored Procedures Will Fire When My Instance Starts?

The DBA can specify custom stored procedures kept in the *master* database to run on service startup via the command:

```
exec sp_procoption 'mystoredproc', 'startup', 'on' It can also be set within Enterprise Manager when editing your stored procedures (see checkbox at the bottom of the edit stored proc window).
```

The DBA can “skip” the procedures on instance startup via the trace flag:

-T4022

You can check if the option is enabled for a stored procedure using `OBJECTPROPERTY` statement:

```
SELECT OBJECTPROPERTY(OBJECT_ID('mystoredproc'), 'ExecIsStartup')
```

Alternatively run this command to get a listing. Care is required when using the system tables between SQL Server version changes:

```
select name
from sysobjects
where category & 16 = 16
order by name
```

When the Database Was Last Accessed?

The best command I have seen to date is this: `EXEC master..xp_getfiledetails 'C:\work\ss2kdata\MSSQL$CKTEST1\Data\pubs_log.ldf'`

Alternate Name	Size	Creation Date	Creation Time	Last Written Date	Last Written Time	Last Accessed Date	Last Accessed Time	Accessed
NULL	786432	20040119	212332	20040626	90412	20040626	90412	32

Essential Trace Flags for Recovery & Debugging

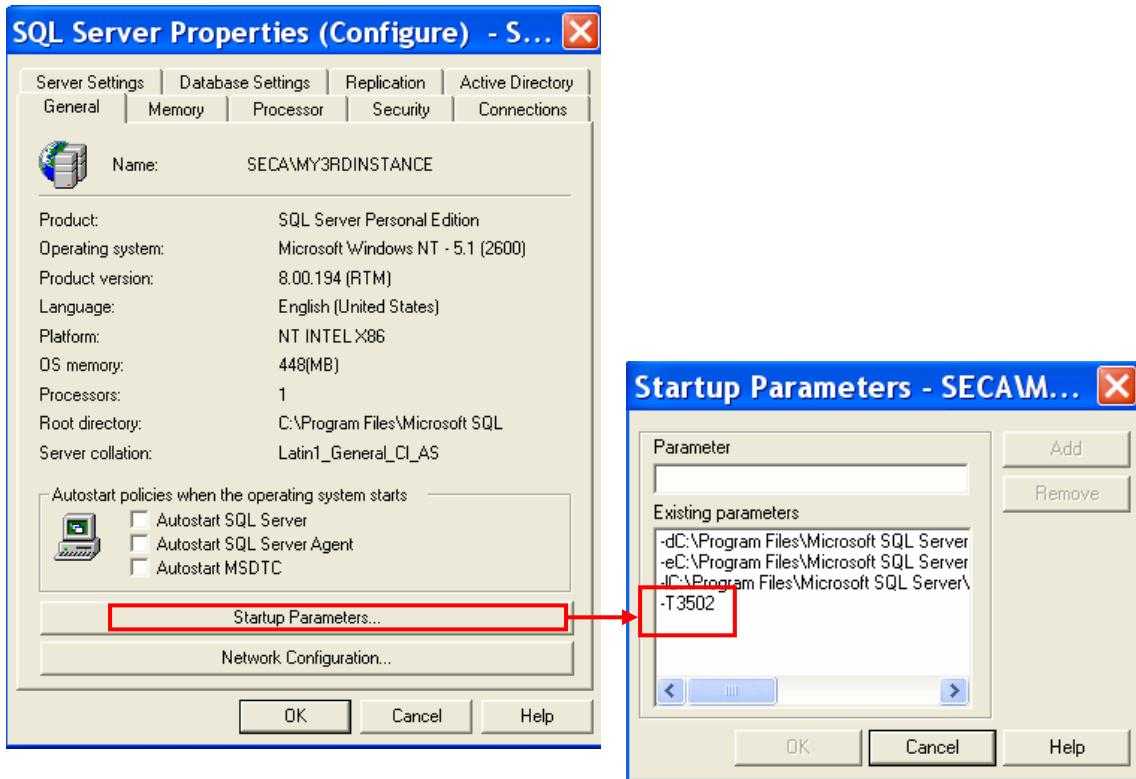
The following trace flags are essential for a variety of recovery scenarios. These flags are referred to throughout this handbook. The use of trace flags allow the DBA to gain a finer granularity of control on the DBMS not normally given.

260	Show version information on extended stored procedures
1200	Prints lock information (process ID and lock requested)
1204	Lock types participating in deadlocking
1205	Detailed information on commands being run at time of deadlock
1206	Complements 1204.
1704	Show information about the creation/deletion of temporary tables
3502	Prints information about start/end of a checkpoint
3607	Skip auto-recovery for all instance databases
3608	As above, except master database
3609	Skip creation of the tempdb database
4022	Bypass master database stored procedure that run on instance startup
7300	Get extended error information on running a distributed query
7502	Disable cursor plan caching for extended stored procedures

Remember that each flag has its own –T<trace#>. Do not try and use spaces or commas for multiple flags and a single –T command. Always review trace output in the SQL Server error log to ensure startup flags have taken effect.

Example of Setting and Verifying the Trace Flags

The trace flag can be enabled via EM as shown below:



This particular trace will force the logging of checkpoints:

Checkpoint -- force the checkpoint

```
2002-07-24 09:35:58.66    spid52  Ckpt dbid 1 phase 1 ended (100000)
2002-07-24 09:35:58.66    spid52  Ckpt dbid 1 started (100000)
2002-07-24 09:35:58.73    spid52  Ckpt dbid 1 complete
```



Trace flags set with the -T startup option affect all connections.

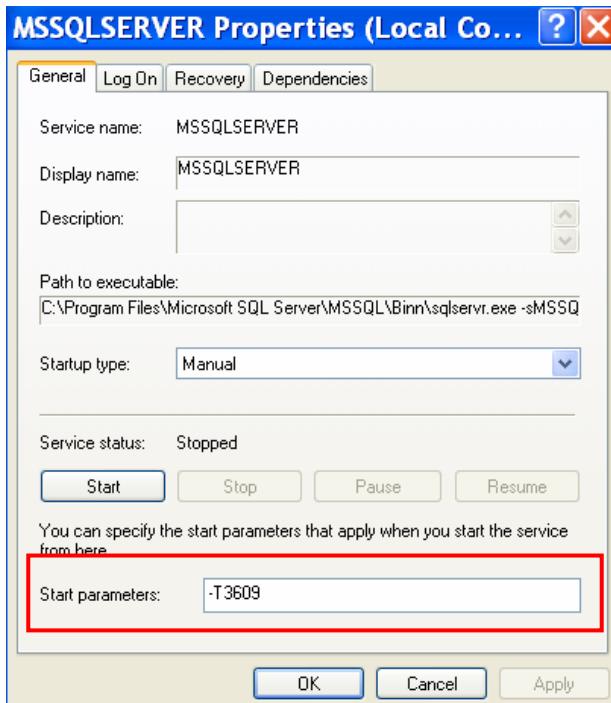
The DBA can also enable trace flags on the command line via the `sqlservr.exe` binary:

```
C:\Program Files\Microsoft SQL Server\MSSQL\Binn>sqlservr -T3609
2004-01-01 23:45:28.75 server      Microsoft SQL Server 2000 - 8.00.194 (Intel X86)
Aug 6 2000 00:57:48
Copyright (C) 1988-2000 Microsoft Corporation
Desktop Engine on Windows NT 5.1 <Build 2600: Service Pack 1>

2004-01-01 23:45:28.78 server      Copyright (C) 1988-2000 Microsoft Corporation.
2004-01-01 23:45:28.79 server      All rights reserved.
2004-01-01 23:45:28.80 server      Server Process ID is 1952.
2004-01-01 23:45:28.81 server      Logging SQL Server messages in file 'C:\Program Files\Microsoft SQ
L Server\MSSQL\LOG\ERRORLOG'.
2004-01-01 23:45:28.83 server      SQL Server is starting at priority class 'normal'(1 CPU detected).

2004-01-01 23:45:28.97 server      SQL Server configured for thread mode processing.
2004-01-01 23:45:28.99 server      Using dynamic lock allocation. [500] Lock Blocks, [1000] Lock Own
er Blocks.
2004-01-01 23:45:29.14 spid3     Recovering all databases but not clearing tempdb.
2004-01-01 23:45:29.15 spid3     Starting up database 'master'.
2004-01-01 23:45:29.42 spid3     0 transactions rolled back in database 'master' (1).
2004-01-01 23:45:29.43 spid3     Recovery is checkpointing database 'master' (1)
2004-01-01 23:45:29.53 server      Using 'SSNETLIB.DLL' version '8.0.194'.
2004-01-01 23:45:29.70 spid5     Starting up database 'model'.
2004-01-01 23:45:29.86 spid3     Server name is 'OLDCOMP'.
2004-01-01 23:45:29.87 spid3     Skipping startup of clean database id 4
2004-01-01 23:45:29.96 spid3     Recovery complete.
2004-01-01 23:45:30.00 spid3     Warning: override, autoexec procedures skipped.
2004-01-01 23:45:34.33 server      SQL server listening on TCP, Shared Memory, Named Pipes.
2004-01-01 23:45:34.34 server      SQL server listening on 122.168.0.1:1433, 203.0.9.185.162:1433, 127
.0.1:1433.
2004-01-01 23:45:34.36 server      SQL Server is ready for client connections
```

Rather than at the command line, we can also enable traces via the services applet:



To view the current enabled session and database wide traces, use these commands:

```
DBCC TRACEON (3604)
DBCC TRACESTATUS (-1)
```

Or rather than using -1, enter the specific trace event#.

For the current connection we can use the DBCC TRACEON and DBCC TRACEOFF commands. You can set multiple flags on and off via DBCC TRACEON(8722, 8602[,etc]). Use -1 as the last parameter to affect all connections:

```
DBCC TRACEON (xxxxxx, -1)
```

Use *profiler* to trace existing connections. Remember that these trace flags are not the same as *trace events* as used by profiler (sp_trace_setevent).



To check startup parameters without connecting to the instance, run regedit, navigate to HKLM\Software\Microsoft\Microsoft SQL Server<instance>\MSSQLServer\Parameters, and review the SQLArgX list of string values.

“Trace Option(S) Not Enabled for this Connection”?

You may get this message with attempting to view trace flag status via the command:

```
DBCC TRACESTATUS (-1)
```

You may know that XYZ flags are enabled, but this routine returns the above message, leaving you in the dark. You could go back over your SQL Server logs and hope to catch the DBCC TRACEON output for the flags used. Otherwise run this first (before tracestatus):

```
DBCC TRACEON (3604)
```

Bulk Copy out All Table Data from Database

Basically I am not going to reinvent the wheel here. There is a MS support document 176818 titled “INF: How to bulk copy out all the tables in a database”. Do note that it simply does a *select * from* tables and uses BCP via xp_cmdshell to dump the files to disk. This may be problematic with TEXT fields. If so, consider the *textcopy.exe* utility that came with SQL Server 6.5 and 7.0. An excellent coverage of this routine can be found on Alexander Chigrik’s website at—“Copy text or image into or out of SQL Server”, www.mssqlcity.com/Articles/KnowHow/Textcopy.htm

SQLSERVR Binary Command Line Options

Starting the SQL Server instance (default or named) manually via the command line is an essential skill for the DBA to master. When I say master, I generally mean that you should be familiar with the options and have experienced first hand the outcome of the option.

The options as of SQL Server 2000 are:

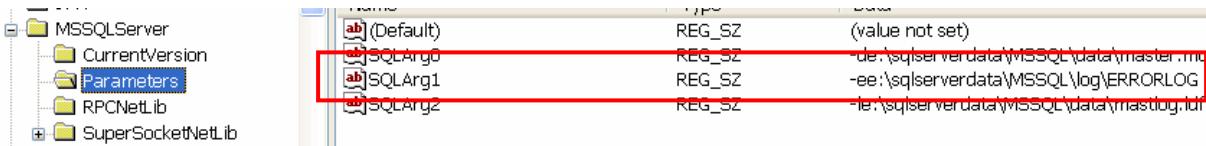
OPTION	SUMMARY / USE
-I<IO affinity mask>	Introduced in SP1 of SS2k.
-c	Do not run as a service
-d<path\filename>	Fully qualified path to the master database primary data file
-l<path\filename>	Fully qualified path to the master database log file
-e<path\filename>	Fully qualified path to the error log files
-m	Start SQL Server in single user (admin) mode; fundamental for master database recovery from backup files.
-f	Minimum configuration mode. Tempdb will be of size 1Mb for its data file and the log will be 0.5Mb; this will only occur if you have not altered tempdb as it only works over the tempdev and templog logical files.
-Ttrace-number	Startup the instance with specified trace flag. Use multiple -T commands for each trace number used.
-yerror-number	If <i>error-number</i> is encountered, SQL will write a stack trace out to the SQL Server error log file.



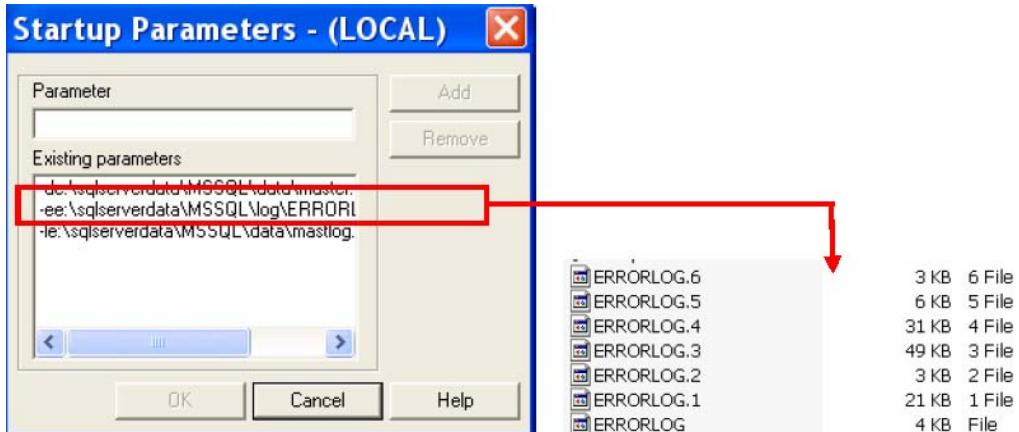
Although not mandatory, the -c and -f parameters are typically used together when starting an instance in minimum configuration mode.

SQL Server Log Files

In SQL Server the *error log* and its destination is defined by the -e startup parameter. This is also stored within the registry during instance start-up:



From within EM, we can right click the instance for global properties and view the startup parameters rather then searching the registry:



The management folder in EM allows the DBA to view the contents of the log files. I say files because SQL Server will cycle through six different log files (default setting). The cycling of the files will occur on each re-start of the SQL Server instance, or via exec `sp_cycle_errorlog`.

The DBA can control the number of cycled logs via a registry change. This change can be done within Query analyser if you like or via EM by right click for properties in the SQL Server Logs item under the Management folder.

```
Exec xp_instance_regwrite
    N'HKEY_LOCAL_MACHINE',
    N'SOFTWARE\Microsoft\MSSQLServer\MSSQLServer',
    N'NumErrorlogs', REG_DWORD, 6
```

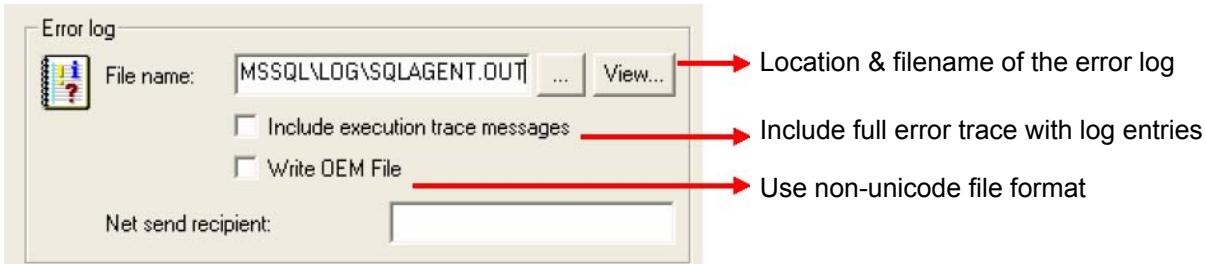
The DBA can view error logs within query analyser using:

```
exec master.dbo.xp_readerrorlog [number of log file, values 0 to 6]
```

The valid list can be retrieved via:

```
exec sp_enumerrorlogs
Archive# Date Log File Size (Byte)
0 06/15/2002 23:59 3646
1 06/14/2002 21:25 21214
2 06/03/2002 21:36 3063
3 06/03/2002 10:29 49852
4 05/26/2002 14:25 31441
5 05/12/2002 16:27 5414
6 05/11/2002 15:54 2600
```

The error log for SQL*Agent is best managed with EM. With SQL*Agent shutdown, select its properties and you can control a range of log settings:



Read Agent Log Example

The website “SQLDev.Net” (<http://sqldev.net/sqlagent.htm>) has a fantastic stored procedure that really simplifies the reading of the SQL Agent logs. Well worth a look.

How and When Do I Switch SQL Server Logs?

I switch them manually at the end of each day (around midnight). The files can get huge and makes life overly difficult when looking for errors, especially with deadlock tracing enabled. To switch them I created a small SQL Agent Job which can be invoked with the following SQL command:

```
exec sp_cycle_errorlog
```

You may feel the need to retain 10 or more logs, rather than the standard 6 (see previous section item). I find the default is more than enough when cycled daily.

Detecting and Dealing with Deadlocks

The NT performance monitor is a good place to start to determine the extent of a problem. We use the counter:

SQLServer:Locks \ Number of Deadlocks\sec

Ideally its value is zero and/or a rare event. There are situations where this is difficult—*especially third party applications or your OLTP database that is also being used for reporting and other batch type events out of your control*. The DBA should follow up with SQL Profiler to better trace the deadlocks occurring.

Profiler is a powerful tracing tool, but it does have some problems when tracing deadlocks as we will see later. On starting a new trace, the DBA should include the events:

```
Errors and Warnings  
Exception  
Locks  
Lock: Deadlock  
Lock: Deadlock Chain
```

If you stayed with this, and waited for your expectant deadlock to occur, you will get very little information of the objects affected or statements executed unless you select the data column *Object Id*. From here you need to manually use OBJECT_NAME to determine the object affected.

Why is this a problem? To get more information you typically include the event T-SQL: SQL: *Batch Completed*. If you run the trace with this option then you will be tracing ALL completed batches and in a busy system, this can mean thousands of entries within minutes; this makes tracing a difficult and time consuming task. Even so, if you can deal with this, you will get a thorough list of statements related to the deadlock; stop the trace after the deadlock occurred and use the find dialog to search the columns for deadlock event, then search backwards from the SPIDS involved in the trace to get a summary of the commands before the deadlock.



Running profiler whilst locking is already underway and a problem, will do you no good. You may only get a small amount of relevant information about the issue (i.e. profiler doesn't magically trace already running processes before continuing on its way with current events).

The client application involved in a deadlock will receive the error# 1205, as shown below:

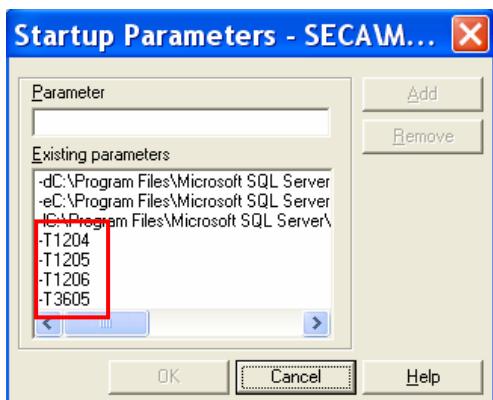
Server: Msg **1205**, Level 13, State 50, Line 1
Transaction (Process ID 54) was deadlocked on {lock} resources with another process and has been chosen as the deadlock victim. Rerun the transaction.

To assist in this circumstance, utilize EM or run the following commands:

```
exec sp_who2                                -- view all sessions
dbcc inputbuffer (52)                         -- get SQL buffer for 52
exec sp_MSget_current_activity 56,4,@spid=52   -- get extended locking information
```

Finally, the DBA can utilize trace flags. This is an effective method for debugging deadlocks and provides some excellent error log data. The flags are:

- 1204 Get lock type and current command affected by deadlock
- 1205 Get extended information about the command being executed (e.g. graph)
- 1206 Complements 1204, get other locks also participating in the deadlock
- 3605 Send trace output to the error log (optional, will go there anyhow).



The screen shots below illustrate the output from a deadlock with the traces enabled. I have no statistics on the adverse effect to DBMS performance, but this is a very effective method for debugging problem systems that are deadlocking frequently but you can never get a comprehensive set of data to debug it.

```
ResType:LockOwner SType:'OR' Mode: S SPID:52 ECID:0 Ec:(0x194e1558) Value...
Victim Resource Owner:
ResType:LockOwner SType:'OR' Mode: S SPID:53 ECID:0 Ec:(0x196b3558) Value...
Requested By:
Input Buf: Language Event: SELECT * FROM Customers
SPID: 52 ECID: 0 Statement Type: SELECT Line #: 1
Owner:0x194cfca0 Mode: X Flg:0x0 Ref:0 Life:02000000 SPID:52 ECID:0
Grant List:?
KEY: 6:1977058079:1 (010086470766) CleanCnt:1 Mode: X Flags: 0x0
Node:2

ResType:LockOwner SType:'OR' Mode: S SPID:52 ECID:0 Ec:(0x194e1558) Value...
Requested By:
Input Buf: Language Event: SELECT * FROM Employees
SPID: 53 ECID: 0 Statement Type: SELECT Line #: 1
Owner:0x194cfac0 Mode: X Flg:0x0 Ref:0 Life:02000000 SPID:53 ECID:0
Grant List:?
KEY: 6:2073058421:1 (d000b4ee90a7) CleanCnt:1 Mode: X Flags: 0x0
Node:1
```

The actual deadlock is around the *customer* and *employer* tables. Two processes have updated one of the two separately and have yet to commit the transaction; they attempted to select each others locked resources resulting in the deadlock. This is not reflected in the log dump.

The ECID is the execution context ID of a thread for the SPID. The value of zero represents the parent thread and other ECID values are sub-threads.

Check with <http://support.microsoft.com> for some excellent scripts to monitor blocking in SQL Server.

Example Deadlock Trace

We have a large COM+ based application that was experiencing deadlocking issues. The key issue here is that COM+ transactions use an isolation level of serialisable. As such, locks of any sort can be a real problem in terms of concurrency. To start resolving the problem we:

1. Worked with the developers in determining out how to replicate the error
 - a) this allowed us to identify the code segments possible causing the error and assisted of course with testing.
2. Set instance startup parameters -T1204 -T1205 -T1206, re-started the instance

3. Ran Profiler

- a) Filter the database we are concerned with
- b) Include event classes
 - Lock:Deadlock
 - Lock:Deadlock Chain
 - SQL:StmtCompleted
 - RPC:Completed
- c) Included standard columns, namely TextData and SPID

4. Ran the code to cause the deadlock. Search the profiler trace:

Lock:Deadlock Chain	Deadlock Chain SPID = 65	Lock:Deadlock Chain	Deadlock Chain SPID = 67	Microsoft (R...)	4	4	67
Lock:Deadlock				Microsoft (R...)	4244		

Lock:Deadlock identifies that SPID 67 was killed. Go back through the trace to locate the commands executed in sequence for the two SPIDS in the chain. Take some time with this, you need to go back through the chain of transaction begins (in this case they are COM+ transactions) to clearly determine what has happened for each SPID's transaction block.

To assist with further debugging, goto your instance error log and locate the deadlock dump chain:

```
2003-01-07 13:46:23.28 spid4 -----
2003-01-07 13:46:23.28 spid4 End deadlock search 17 ... a deadlock was found.
2003-01-07 13:46:23.28 spid4
2003-01-07 13:46:23.28 spid4 RestType:LockOwner Stpye:'OR' Mode: IX SPID:67 ECID:0 Ec:(0x25bf3558) Value:0x26
2003-01-07 13:46:23.28 spid4 Victim Resource Owner:
2003-01-07 13:46:23.28 spid4 RestType:LockOwner Stpye:'OR' Mode: IX SPID:67 ECID:0 Ec:(0x25bf3558) Value:0x26
2003-01-07 13:46:23.28 spid4 Requested By:
2003-01-07 13:46:23.28 spid4 Input Buf: RPC Event: [REDACTED]
2003-01-07 13:46:23.28 spid4 SPID: 65 ECID: 0 Statement Type: SELECT Line #: 153
2003-01-07 13:46:23.28 spid4 Owner:0x1937d520 Mode: S Flg:0x0 Ref:2 Life:02000000 SPID:65 ECID:0
2003-01-07 13:46:23.28 spid4 Grant List:-
2003-01-07 13:46:23.28 spid4 TAB: 8:918450496 [REDACTED] CleanCnt:1 Mode: S Flags: 0x0
2003-01-07 13:46:23.28 spid4 Node:2
2003-01-07 13:46:23.28 spid4
2003-01-07 13:46:23.28 spid4 RestType:LockOwner Stpye:'OR' Mode: Range-S-S SPID:65 ECID:0 Ec:(0x25673558) Val
2003-01-07 13:46:23.28 spid4 Requested By:
2003-01-07 13:46:23.28 spid4 Input Buf: RPC Event: [REDACTED]
2003-01-07 13:46:23.28 spid4 SPID: 67 ECID: 0 Statement Type: UPDATE Line #: 354
2003-01-07 13:46:23.28 spid4 Owner:0x1cf86280 Mode: X Flg:0x0 Ref:0 Life:02000000 SPID:67 ECID:0
2003-01-07 13:46:23.28 spid4 Grant List:-
2003-01-07 13:46:23.28 spid4 KEY: 8:966450667:1 (e0003dcf0911) CleanCnt:1 Mode: X Flags: 0x0
2003-01-07 13:46:23.28 spid4 Node:1
2003-01-07 13:46:23.28 spid4
2003-01-07 13:46:23.28 spid4 Wait-for graph
2003-01-07 13:46:23.28 spid4
2003-01-07 13:46:23.28 spid4 ...
2003-01-07 13:46:23.28 spid4
2003-01-07 13:46:23.28 spid4 Cycle:□ RestType:LockOwner Stpye:'OR' Mode: Range-S-S SPID:65 ECID:0 Ec:(0x25673
2003-01-07 13:46:23.28 spid4 Node:2□ RestType:LockOwner Stpye:'OR' Mode: IX SPID:67 ECID:0 Ec:(0x25bf3558) Va
2003-01-07 13:46:23.28 spid4 Node:1□ RestType:LockOwner Stpye:'OR' Mode: Range-S-S SPID:65 ECID:0 Ec:(0x25673
2003-01-07 13:46:23.28 spid4 Deadlock cycle was encountered .... verifying cycle
2003-01-07 13:46:23.28 spid4
2003-01-07 13:46:23.28 spid4 -----
```

select object_name(918450496)
will give you the table name to help
identify the possible problem
statement to start looking for in the
batch of SQL being executed.

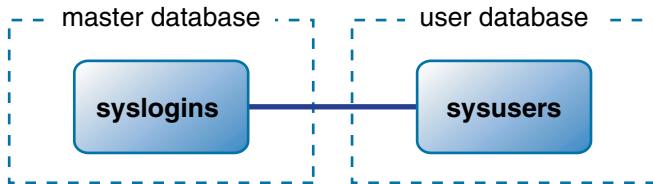
Last command
from buffer.
(stored proc or
DML statement)

Current lock
being held

IX lock wanting
to be taken out

Orphaned Logins

At times, the DBA will restore a database from one instance to another. In doing so, even though the *login* exists for the instance, the SID (varbinary security ID) for the login is different to that in the other instance. This effectively “orphans” the database user from the database login due to this relationship between *master..syslogins* and *mydb..sysusers*.



```
SELECT SUSER_SID('user1')
```

```
0x65B613CE2A01B04FB5E2C5310427D5D5 -- SID of user1 login, instance A  
0x680298C78C5ABC47B0216F035B3ED9CC -- SID of user1 login, instance B
```

In most cases, simply running the command below will fix the relationship and allow the login to access the user database (must run against every database in which the login is valid).

```
exec sp_change_users_login <see books online>
```

This will only work for SQL logins and not fully integrated logins (which is a down right pain). Write your own script to resolve this problem.

If you are still getting errors, consider removing the user from the *sysusers* table in the restored database, and re-add the user.

The DBA can validate NT login accounts via the command:

```
EXEC sp_validateLogin
```



Do not use ALIASES; this allowed the DBA to map a single login to many database users. This is a dated feature that may disappear in newer versions.

Microsoft released a great support document related to the scripting of logins, which includes the original password. The script is not complete in terms of all possible options, but is very handy:

Example: ([http://support.microsoft.com/default.aspx?scid=kb;\[LN\];Q246133](http://support.microsoft.com/default.aspx?scid=kb;[LN];Q246133))

```
/* sp_help_revlogin script
** Generated Nov 17 2002 12:14PM on SECA\MY2NDINSTANCE */

DECLARE @pwd sysname

-- Login: BUILTIN\Administrators
EXEC master..sp_grantlogin 'BUILTIN\Administrators'

-- Login: user1
SET @pwd = CONVERT (varbinary(256),
0x0100420A7B5781CB9B7808100781ECAC953CB1F115839B9248C3D489AC69FA8D5C4BE3B11B1ED1A3
0154D955B8DB)
EXEC master..sp_addlogin 'user1', @pwd, @sid = 0x680298C78C5ABC47B0216F035B3ED9CC,
@encryptopt = 'skip_encryption'

-- Login: user2
SET @pwd = CONVERT (varbinary(256),
0x01006411A2058599E4BE5A57528F64B63A2D50991BC14CC59DB0D429A9E9A24CA5606353B317F4D4C
A10D19A2E82)
EXEC master..sp_addlogin 'user2', @pwd, @sid = 0xF1C954D9C9524C41A9ED3EA6E4EA82F4,
@encryptopt = 'skip_encryption'
```

Orphaned Sessions – Part 1

I rarely have to deal with orphaned sessions and worry little about them unless of course it's resulting in system degradation or chewing server CALs. The trick here is the identification of the database session. This can be confusing, especially with connection sharing via COM+ components—database process that you *believe* is related to a COM+ connection could be different when you next refresh the screen.

We can assume the session is orphaned or killable by:

1. Processes Status = 'awaiting command'
2. Processes Last Batch date – Getdate() is longer than what we *typically* expect
3. We know the XYZ application and its logins/db connect properties crashed, but processes remain.
4. If the SPID is -2 then assume the session is orphaned

The process information can be gleamed from `sp_who` or `sp_who2` or querying `sysprocesses`; and the utmost care must be taken. As such, I highly recommend you:

1. Run – DBCC INPUTBUFFER against the spid to determine its last statement of work
2. Run – `sp_lock` to determine if the SPID is locking objects

To kill the SPID, use the KILL command.

The System Administrator may consider altering the TCP/IP keep alive timeout setting in the registry:

`HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\KeepAliveTime`

The default is 2hrs and is measured in milliseconds. As a general guide, consider a lower value in the order of one hour.

Orphaned Sessions – Part 2

An orphaned session has a SPID of -2. Orphaning may be caused by a variety of things (though are rare) and is typically linked to the distributed transaction coordinator (DTC). A DTC related problem will show up in the SQL Server log files with an error such as “SQL Server detected a DTC in-doubt transaction for UOW <value>”. If the transaction issue can not be resolved, then a kill statement can be issued over the UOW code. For example:

```
Kill 'FD499C76-F345-11DA-CD7E-DD8F16748CE'
```

The table `syslockinfo` has the UOW column.



Use Component Services when appropriate to drill into COM+ classes and their instantiations to locate UOW to assist with killing the correct SPID at the database.

Change DB Owner

You can change the database owner with the command:

```
USE MyDatabase
EXEC sp_changedbowner 'MyUser'
GO
```

This requires the sysadmin instance privilege, and the user will logically replace the `dbo` user (`dbo` can never be revoked/removed altogether). You cannot run this command against the system databases. There is seldom any need to use this command.

Transfer Diagrams Between Databases

To transfer a database diagram from one database to another:

1. Determine the Object Id (or diagram) to be copied from database A to B. The id column for *dtproperties* is an identity column. I have two diagrams, one called “CKTEST” and the other called “Another Diagram”.

```
select objectid, value from dtproperties
```

objectid	value
1	NULL
1	{EA3F6268-D998-11CE-9454-00AA00...
1	CKTEST
1	690
1	NULL
1	17920
1	NULL
8	NULL
8	{EA3F6268-D998-11CE-9454-00AA00...
8	Another Diagram
8	348
8	NULL
8	8704
8	NULL

2. To transfer the diagram, we need to ensure the objectid column is unique. If not, we can set the value manually to whatever value we like.

```
INSERT INTO B.dbo.dtproperties
(objectid, property, value, lvalue, version)
SELECT objectid, property, value, lvalue, version
FROM A.dbo.dtproperties
```

The databases should be identical of course. If not, opening diagrams will result in the loss of objects within the diagram OR you will simply see no diagrams listed at the destination.



No diagrams in the destination database? The *dtproperties* table will not exist and will, of course require changes to the steps above.

Transfer Logins Between Servers

To transfer logins between servers and retain the logins' passwords (SQL Logins), consider utilizing the DTS task to transfer logins between servers, or use the following SQL statement:

```
select 'sp_addlogin @loginame = ' + name + ', @passwd = "' + password  
+ '", @encryptopt = skip_encryption, @deflanguage = "' + language + ''''  
+ char(13) + 'go' from sysloginswhere name in ('user1', 'user2')
```

The key to this script is the *skip encryption* option. Note that we still need to:

1. setup the login database user relationship (sp_adduser)
2. assign database user privileges

Killing Sessions

Within SQL Server, each connection (physical login) is allocated a *spid* (system server process identifier and worker thread). To identify them we can execute the system stored procedures:

```
exec sp_who  
- OR -  
exec sp_who2
```

The DBA can also use the *current activity* option under the Management Group folder and select Process Info.



Be warned. I have experienced major performance problems when forcing the refresh of current activity via Enterprise Manager to a point where the CPU's hit a solid 50% for 5 minutes before returning back to me. This is not acceptable when running against a hard working production system.

Once the SPID has been identified, we use the KILL command to remove the session. For example:

```
select @@spid          -- get my sessions SPID, in this case its 51  
exec sp_who2           -- from output, determine SPID to be killed  
Kill 55                -- issue kill request to DBMS  
Kill 55 with statusonly -- get status of the request
```

*SPID 55: transaction rollback in progress. Estimated rollback completion: 100%.
Estimated time remaining: 0 seconds.*

The DBA should reissue sp_who2 to monitor the SPID after the kill to ensure success. Also consider looking at and joining over the tables:

- sysprocesses
- syslock
- syslockinfo

You cannot kill your own processes. Be very careful you do not kill system processes. The *processadmin* fixed system role will allow a user to kill SQL Server sessions.



Killing SPIDs running extended stored procedures or which did a call-out to a user created DLL may take some time to kill and, in some cases, seem to have stopped but remain as a running process.

The ALTER Statement

An alternative method to the KILL command is the *alter database* statement. The only issue here is the alter database statement requires the database status to be altered, you cannot simply kill user connections without making the database read only, for DBO use only for example. Therefore, it may not suite your specific requirements.

Here is a practical example of its use:

```
alter database northwind set restricted_user with rollback immediate
```

The termination clause will perform the session removal and rollback of transactions. If the termination clause is omitted, the command will wait until all current transactions have been committed/rolled back.

See BOL for more information about the command. Try and use this command over KILL when you need to disconnect all databases sessions AND change the database status.

How Do I Trace the Session Before Killing It?

To get further information about a session, consider the command:

```
DBCC PSS (suid, spid, print-option)
```

For example:

```
--Trace flag 3604 must be on  
DBCC TRACEON (3604)
```

```
--Show all SPIDs  
DBCC PSS
```

```
DBCC TRACEOFF (3604)  
GO
```

Another option apart from utilizing profiler is:

```
exec sp_who2                                -- view all sessions  
dbcc inputbuffer (52)                         -- get SQL buffer for 52  
exec sp_MSget_current_activity 56,4,@spid=52   -- get extended locking  
information
```

Taking this a step further, adapt the code fragment below to iterate through SPIDs and capture their event data returned from INPUTBUFFER for further checking.

```
DECLARE @ExecStr varchar(255)  
CREATE TABLE #inputbuffer  
  
(  
    EventType nvarchar(30),  
    Parameters int,  
    EventInfo nvarchar(255)  
)  
SET @ExecStr = 'DBCC INPUTBUFFER(' + STR(@@SPID) + ')'  
  
INSERT INTO #inputbuffer  
EXEC (@ExecStr)  
  
SELECT EventInfo  
FROM #inputbuffer
```



DBCC INPUTBUFFER only shows a maximum of 255 characters and the first statement only if the process is executing a batch. Consider ::fn_get_sql() instead.

As of SQL Server 2000 SP3 (or 3a), the DBA can use the system function ::fn_get_sql() in association with the **master..sysprocesses** table and its three new columns:

- sql_handle – handle to the currently running query, batch or stored proc, a value of 0x0 means there is no handle
- stmt_start – starting offset within the handle
- stmt_end – end of the statement within the handle (-1 = end handle)

For a single script to save your time try this site:

http://vyaskn.tripod.com/fn_get_sql.htm

Setting up and Sending SQL Alerts via SMTP

This section will show you, through code, how to enable SQL Agent alerts and using a simple stored procedure and DLL, send emails via an SMTP server rather than using SQL Mail. We will cover these items:

- SQL Agent Event Alerts
- SQL Agent Tokens
- Calling DLL's via *sp_OA* methods
- Using RAISEERROR
- The SMTP DLL

Here is the code for our *simplecdo* DLL (some items have been cut to make the code easier to read). The routine is coded in VB and makes use of the standard CDO library:

```
Public Function SendMessage(ByVal ToAddress As String, _
    ByVal FromAddress As String, _
    ByVal SubjectText As String, _
    ByVal BodyText As String, _
    ByVal Server As String,
    ByRef ErrorDescription As String) As Long
    'This is the original function (no attachments).
    '

    Dim lngResult As Long

    lngResult = Send(ToAddress, FromAddress, SubjectText, BodyText, Server, "", 
    ErrorDescription)

    SendMessage = lngResult

End Function

Private Function Send(ByVal ToAddress As String, _
    ByVal FromAddress As String, _
    ByVal SubjectText As String, _
    ByVal BodyText As String, _
    ByVal Server As String,
    ByVal AttachmentFileName As String,
    ByRef ErrorDescription As String)
    'Simple function for sending email from an SQL Server stored procedure.
    'Returns 0 if OK and 1 if FAILED.
    '

    Dim Result As Long
    Dim Configuration As CDO.Configuration
    Dim Fields As ADODB.Fields
    Dim Message As CDO.Message
```

```

On Error GoTo ERR_HANDLER

'Initialise variables.
Result = 0
ErrorDescription = ""

'Set the configuration.
Set Configuration = New CDO.Configuration
Set Fields = Configuration.Fields

With Fields
    .Item(CDO.CdoConfiguration.cdoSMTPServer) = Server
    .Item(CDO.CdoConfiguration.cdoSMTPServerPort) = 25
    .Item(CDO.CdoConfiguration.cdoSendUsingMethod) = CdoSendUsing.cdoSendUsingPort
    .Item(CDO.CdoConfiguration.cdoSMTPAuthenticate) = CdoProtocolsAuthentication.cdoAnonymous
    .Update
End With

'Create the message.
Set Message = New CDO.Message
With Message
    To = ToAddress
    .From = FromAddress
    .Subject = SubjectText
    .TextBody = BodyText
    Set .Configuration = Configuration
    'Send the message.
    .Send
End With

EXIT_FUNCTION:

'Clean up objects.
Set Configuration = Nothing
Set Fields = Nothing
Set Message = Nothing

Send = Result

Exit Function
ERR_HANDLER:

Result = Err.Number

ErrorDescription = "Number [" & Err.Number & "] Source [" & Err.Source & "] Description
[" & Err.Description & "]"
Me.LastErrorMessage = ErrorDescription

GoTo EXIT_FUNCTION
End Function

```

Copy the compiled DLL to your DB server and run the command below to install:

regsvr32 simplecdo.dll

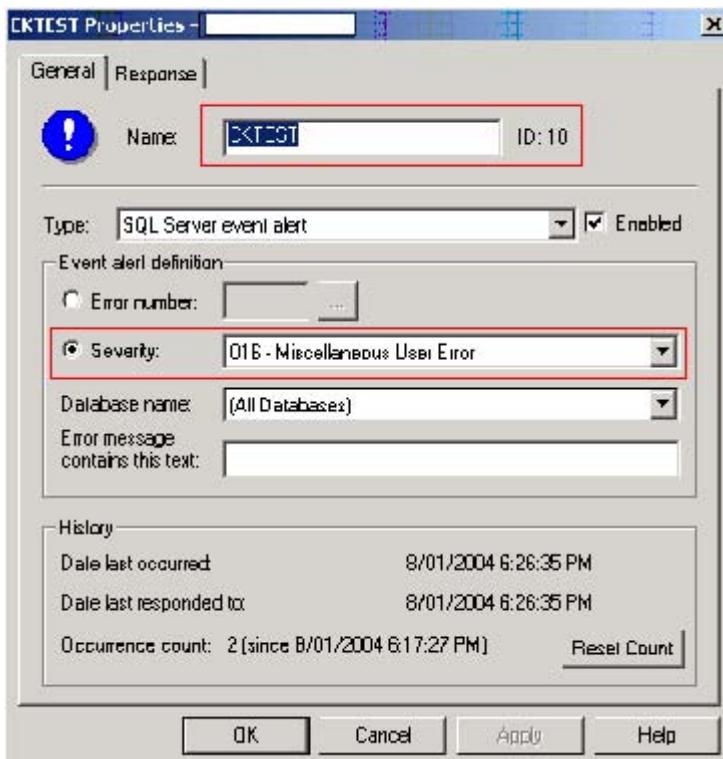
The Stored Procedure

The routine below makes the simple call to the SMTP email DLL. We have hard coded the IP (consider making it a parameter). I was also sloppy with the *subject heading* for the email. Again this should be a parameter or better still a SQL Agent Token (see later).

```
CREATE PROCEDURE usp_sendmail (@recipients varchar(200), @message varchar(2000)) AS  
declare @object int, @hr int, @v_returnval varchar(1000), @serveraddress varchar(1000)  
set @serveraddress = '163.232.xxx.xxx'  
exec @hr = sp_OACreate 'SimpleCDO.Message', @object OUT  
exec @hr = sp_OAMethod @object, 'SendMessage', @v_returnval OUT, @recipients,  
@recipients, 'test',  
  
@message, @serveraddress, @v_returnval exec @hr = sp_OADestroy @object  
GO
```

Creating the Alert

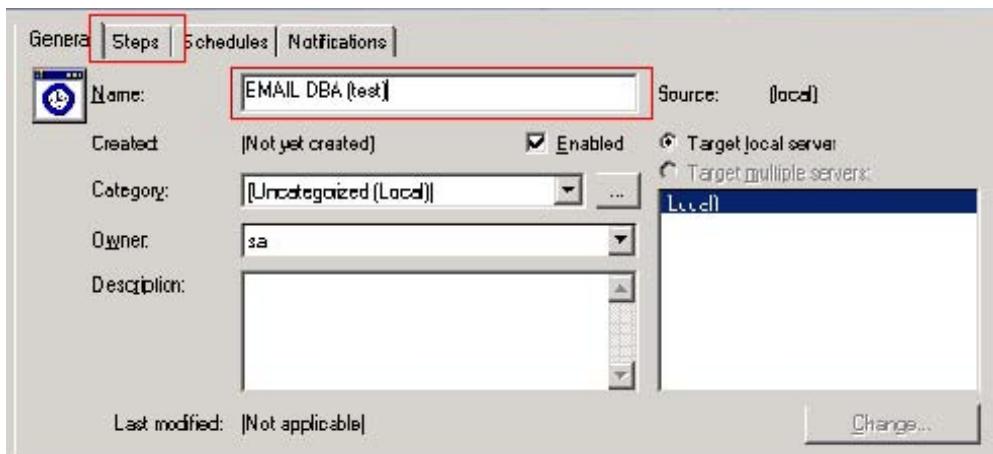
Run Enterprise Manager, under the Management folder expand SQL Agent and right click *Alerts -New Alert*.



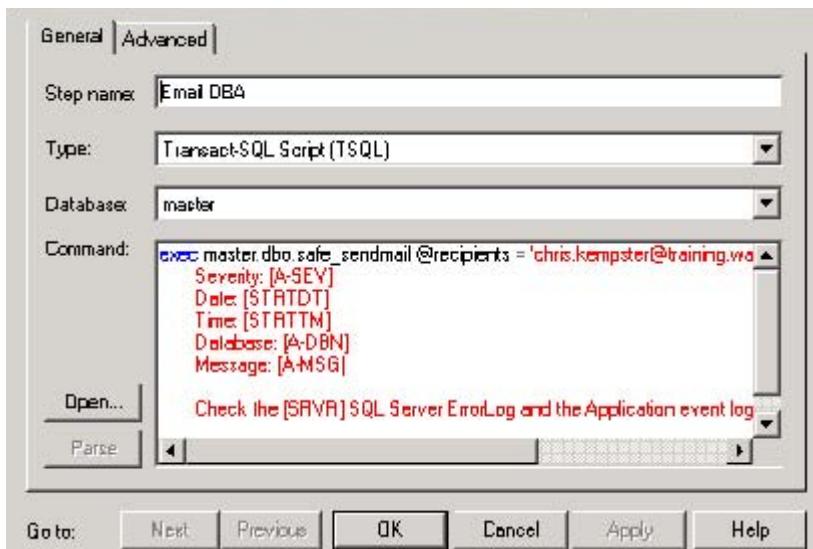
In this case our alert is called **CKTEST**. We are going to send the DBA an email whenever a logged severity 16 message occurs for any databases (not really practical, but this is just an example).

Click on the *Response* tab next.

Uncheck the *email*, *pager* and *net send* options (where applicable for your system). Check the *execute job* checkbox, drop down the list box and scroll to the top, and select <New Job>.



Enter the Name of the new SQL Agent Job, then press the *Steps* button to create a step that will call our stored procedure.



Here we enter the step name, it is a t-sql script of course, and the command which is:

```
exec master.dbo.usp_sendmail @recipients = 'support@chriskempster.com', @message = '
Error: [A-ERR]
Severity: [A-SEV]
Date: [STRDT]
Time: [STRTTM]
Database: [A-DBN]
Message: [A-MSG]
```

Check the [SRVR] SQL Server ErrorLog and the Application event log on the server for additional details'

This is where the power of Agent Tokens comes into play. The tokens will be *automatically filled in*. There are numerous tokens you can leverage; here are some examples:

[A-DBN]	Alert Database name
[A-SVR]	Alert Server name
[DATE]	Current Date
[TIME]	Current Time
[MACH]	Machine name
[SQLDIR]	SQL Server root directory
[STRTDT]	Job start time
[STRTTM]	Job end time
[LOGIN]	SQL login ID
[OSCMD]	Command line prefix
[INST]	Instance name (blank if default instance)

Click OK twice. The Job and its single step are now created. In the Response windows press *Apply*, then press OK to exit the Alert creation window and return back to enterprise manager.

Goto Jobs under SQL Server agent to confirm the new Job and its step we have just created.

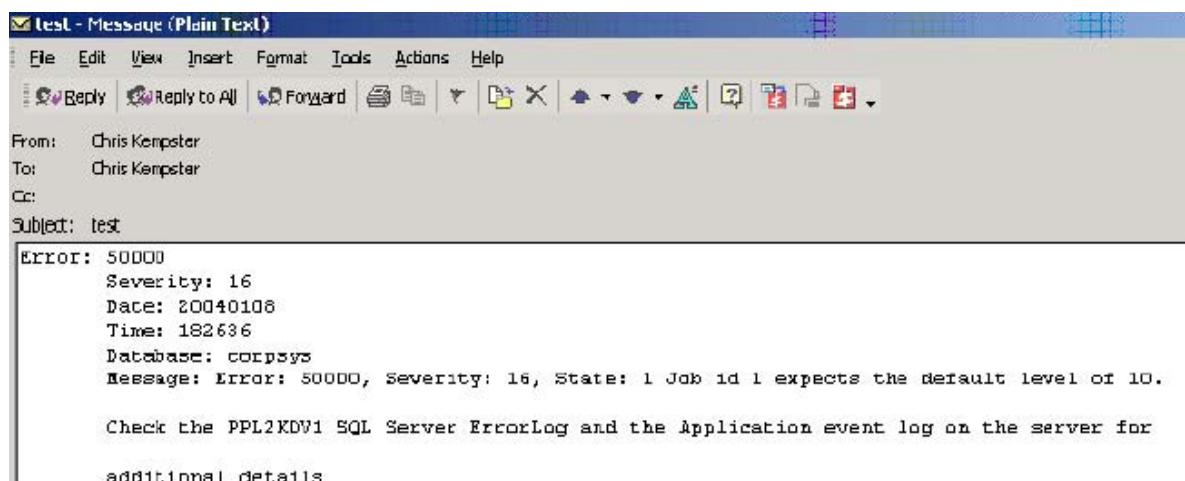
Testing the Alert

Run Query Analyzer and run the following:

```
RAISERROR ('Job id 1 expects the default level of 10.', 16, 1) WITH LOG
```

The *with log* clause is important. The alert will not fire without it.

Shortly I receive the following email:



Recommended Backup and Restore Alerts

The following alerts are highly recommended for monitoring SQL Server backups and recovery (59):

1. Error – 18264, Severity – 10, Database successfully backed up
2. Error – 18204 and 18210, Severity – 16, Backup device failed
3. Error – 3009, Severity – 16, Cannot insert backup/restore history in MSDB
4. Error – 3201, Severity – 16, Cannot open backup device
5. Error – 18267, Severity – 10, Database restore successfully
6. Error – 18268, Severity – 10, Database log restored successfully
7. Error – 3443, Severity – 21, Database marked as standby or read-only but has been modified, restore log cannot be performed.

The DBA can alter the text, and include the alert tokens as necessary.

HIGH AVAILABILITY

One of the most important issues for many organizations revolves around disaster recovery (DR), which goes hand-in-hand with the topic of high availability.

When we talk about high availability, we are primarily focused on almost seamless failover of our servers hosting the applications they are running; and the technologies to support the continuation of service with as little interruption to the business as possible. The solution you come up with will be dictated by the realization of its:

1. value-add to the business and customer expectations (latent and blatant)
2. cost of system downtime and manual cutover
3. issues of business continuity and system reliance

The problem you tend to have is that systems typically grow into this realization rather than being born with it. As such, the DBA and system architects must carefully consider the overarching issues with application state, server configuration, OS and DBMS editions purchased, technologies being used (clusterable?) and to some degree, “brace the future environment” in which the application will live and breath.

Throughout this chapter, we will compare and contrast the high availability options to support your DR plan and identify issues in advance. We also cover clustering with VMWARE so you can repeat the same tests using your home PC.

Purchasing the Hardware

So What Hardware Should I Buy?

The selection of hardware, be it for a cluster or not is a tough job. The DBA relies on past performance indicators, systems growth, wisdom of yourself and fellow IT professionals and of course, restrictions outlined by enterprise architecture initiatives (and how much of the budget you have). These and other factors can turn a great system into one that has mediocre performance with great disaster-recovery, visa versa or none of these. This section discusses some of many issues related to hardware selection.

Here are some general “rules”:

1. Enterprise quality applications require hardware from enterprise hardware vendors (HP/COMPAQ, IBM, DELL etc)—but shop around.
2. Try and picture your future production environment and the business applications hosted within it. Attempt to marry it with your enterprise architecture and its possible visions for systems consolidation, this may also assist in building your dev/test environment. Looking forward is an essential planning skill that takes some time to master, especially with IT.

3. Consider cheaper “components”, i.e. RAM, prices can be hugely inflated from the larger hardware vendors. An absolute minimum for RAM is 2Gb ECC. Be very careful with desupport dates on hardware, and again, looking into the future through research may save you thousands of dollars.
4. Check OS compatibility lists carefully; you may find Windows 2003 is not on the supported OS list for example – never take the risk.
5. Plan to use RAID in your DEV/TEST servers—but watch out for the number of free drive bays, size your disks carefully!
6. Be pragmatic with your RAID configuration, instances with multiple databases will not mean a single RAID-1 set for each transaction log. Be concerned with logical/physical read/writes and the tuning of SQL to reduce overall system load. Be aware that RAID-5, with suitable HBA backup cache, and read/write cache will meet most expectations.
7. Go SCSI-320 for internal disks. Read the next section on RAID and Storage for a further insight. In production we should try a leverage enterprise class shared mass storage devices (SAN's) over direct attached storage solutions or very large internal system storage.
8. Plan to hook into your enterprise backup software rather than buying the hardware and tapes—networking issues? size of backups? impact on the business? time to backup and restore? responsibilities and accountabilities? agent OS compatibility? Tape drive per server is a very costly solution in the longer term.
9. When you make a decision on the hardware specs take time research the choice—any issues? compatibility problems? potential install nightmares?
10. Always pay for longer terms parts/labour warranties – but read the conditions and turn around times very carefully.
11. Consider 64bit computing as the end-game for future hardware infrastructure.

Taking this a little further, here is a simple list to review and add to when selecting hardware:

SERVER CHECK LIST

Operating systems supported by the hardware have been checked? (and you know the OS requirements of the services to be run?)

Existing order will cover the services to be provisioned?

Enterprise backup agents/software supported on the chosen HW and OS?

SAN connection required? Dual HBA's? and is there a standard HBA to suit existing switch technology?

Existing RACK's can house proposed servers (in terms of free space and existing rack dimensions)

Warranty and support has been factored in? restrictions based on distance, public holidays, where parts are located etc

SERVER CHECK LIST

CPU, RAM (including stick configurations and free slots), HBA's, NIC's, Power (swappable?)

BTU and power draw?

Installation and shipping included? Insurance covered?

Monitor/CD-ROM/USB/Mouse required?

SAN bootable? (any specific restrictions?)

Additional RAID card to achieve RAID 5?

Power slots required?

Management network interface?

Specific cluster support/restrictions i.e. ex public holidays, parts are stored on the other side of the country?

Can the OS you plan to load onto the hardware support the features you plan to purchase? ie. all the RAM/CPU's, multi clustered nodes etc.

What is the desupport date for the hardware? Is it old technology?

Do you plan to use virtualization technology? and if so, will the vendor support it? What are the limits of a virtual machine that will adversely effect the services delivered on them?

Is vertical scalability important? what criteria does this impose?

Can the power requirements of the server or rack be met in terms of its placement?

Can the equipment be moved to the location? (physical restrictions)

UPS required?

Dual Ethernet and HBA cards required? Communications team have standardized on specific models?

So you are ordering a 4 cpu machine – what are the database licensing implications? Will you purchase software assurance?

Hardware listed in the Windows HCL? (discussed in the next section)

I often see the consolidation of servers (incl. storage) and their hosted applications onto one of the following architectures:

1. Medium to large scale co-located servers in clusters within a data center; this typically represents the same set of servers for each application but with their co-location to a single managed environment.
2. SAN (fiber or iSCSI) mass storage, leveraged by racked 1U and 2U blade servers. The blade servers are of course co-located servers into a data center; this is similar to 1) but sees the introduction of storage consolidation and commodity blade servers.

3. SAN (fiber or iSCSI) mass storage, leveraged by consolidated large scale servers hosting numerous virtual servers using virtual server software/or alternatively virtualized at the BIOS/OS level where supported (typically Unix implementations). This is taking 1) a step further with the reduction in the total number of servers into clustered large-scale enterprise servers that virtually host a reduced number of environments where possible.
4. SAN with VMWARE or other server virtualization technology (such as LPAR) on highly scalable server infrastructure.

See *Appendix A* for further information on SAN, NAS, iSCSI, RAID and TAPE issues. We also cover the basics of data center based hardware such as racks and blade servers.

No matter the project's budget or the timeframe for production rollout, make every attempt to align with your enterprise vision for hosted production systems and seek active buy-in and support from executive management.

What is the HCL or the “Windows Catalog”

The Quality Online Service offered by Microsoft is a quality endorsement policy and procedure that allows hardware and software vendors to use the “*designed for windows*”, “*certified for windows*” and “*.net connected*” logos. The vendor must pass a range of tests, typically managed by a third party company which at last view was managed by *VeriTest* (www.veritest.com). Apart from the fact that software and hardware is retested independently, the certification typically means you will (potentially) have fewer issues when purchasing the goods and will be better supported by Microsoft. This is particularly important in clustered server implementations.

As a general recommendation, read before you buy! Do not purchase hardware without some prior understanding of HCL (hardware compatibility list) and the Microsoft Windows Catalog support; especially in large scale enterprise solutions and associated operating systems (like data centre server). I personally do not believe Microsoft would provide any different support to that given to HCL hardware/software buyers, but will certainly make your life harder if you do have serious problems.

A classic case in note was iSCSI (discussed later) support within Windows 2003 and MS Exchange. Although fully supported, the HCL was very sparse in terms of vendors compliance as at March 2004.

Read more at: <https://winqual.microsoft.com/download/default.asp>

High Availability using Clusters

In my previous e-book, “SQL Server 2k for the Oracle DBA”, I discussed SQL Clustering in some depth but not much about recovery in this environment. This chapter will cover a large number of day-to-day issues you may experience using this technology, and provide a walkthrough using VMWARE.

Please download the free chapter on High Availability from my website for more information on SQL Clusters.

Let us clarify first the following availability terminology:

1. Hot Standby (passive SQL Cluster) – immediate restoration of IT services following an irrecoverable incident. The delay will be less than 2-4hrs.
2. Warm Standby (bring online passive DR servers) – re-establishment of a service within 24 to 72hrs, be it local or remote in nature. The full service is typically restored.
3. Cold Standby (establishment of new servers/environment) – longer than 72hr restoration of full IT services.

Identification of SPOF (single points of failure) and their risk/mitigation and contingencies strategies is of key importance. The design for high availability needs to consider the elimination of single points of failure or provision alternate components to minimize business impact. The ITIL framework suggests these definitions:

1. High Availability – mask or minimizes the effect of IT component failure.
2. Continuous Operation – mask or minimize the effect of planned downtime.
3. Continuous Availability – mask or minimize the effect of ALL failures.

From a SQL perspective, the DBA can utilize a number of strategies to mitigate the risks:

1. Windows Clustering and SQL Server cluster technology;
2. Log Shipping;
3. Replication – can be complex to configure and manage. Also remember that changes are not automatically provisioned into the published replica(s), requiring ongoing administrative effort and careful change control practices. Due to this, I do not regard it as a overly effective form of high availability; and
4. Federated Databases – primarily for performance over availability.

This chapter will cover a) and b) only. There are many third party high availability solutions not covered in this ebook, including:

1. Legato Costandby Server
2. PolyServe Matrix HA
3. SteelEye LifeKeeper
4. Veritas Clustering Service (VCS) and ClusterX products

Using VMWARE we will cover SQL Server Clustering and troubleshooting techniques.

VMWARE SQL Cluster – by Example

The VMWARE software from VMWARE the company (www.vmware.com) allows us to run *virtual machines* (VM) over an existing operating system, creating an interface layer between physical devices and allowing the user to define *virtual hardware* from it. The software goes further though, allowing us to build new devices that physically do not exist, such as new disk drives, network cards etc.

The version we will be using is called *VMWARE Workstation v4.0*, check the vendors website for new editions. The direct competitor to VMWARE in the Microsoft space is Microsoft Virtual PC (and server when it is released). Although a good product, I found it slow and did not support SQL clustering during the writing of this particular chapter.



Installing VMWARE virtual machines will NOT replace your existing host operating system in any way. The VMWARE itself depends on the host OS itself to run.

The aim here is to:

1. allow the DBA to replicate SQL Cluster testing with a relatively low spec PC—the machine used for this example is a AMD 2.6Ghz, 1Gb RAM, single 80Gb drive;
2. catalyst to discuss cluster design issues; and
3. allow DBA's to realize the benefits of virtual server environments as another form of high-availability and server consolidation.

Using VMWARE in Production?

To be right up front, I am a BIG advocate for VMWARE, not only in your development/test environments, but also in production. As an example, I worked with a customer that ran mission critical applications in high-scalable IBM x445 servers (16 CPU's) over VMWARE ESX.

Four servers were provisioned within a geographically dispersed environment connected to large SAN's via fiber interconnects. The servers ran a mix of Linux and Windows 2003 operating systems, all provisioned from *golden templates* (discussed later) running as SQL clusters and a variety of other smaller applications.

The essential value added extras with the virtualized server environment were:

1. speed in which new virtual servers (and application services) can be provisioned;
2. virtual servers could be moved in near realtime (whilst running) between physical hardware through VMOTION technology;
3. test environments could be establish as mirrors of production within a matter of minutes;
4. VMWARE Virtual Center provides a single unified management interface to all systems, no matter their underlying OS; and
5. large capacity (and relatively cheap x86 architecture) hardware can be better utilized.

Step 1. Software & Licensing

In the following steps we are using VMWare Workstation 4.0.5; you can download a trial from www.vmware.com. Install the software as per installation guide. As a general idea I run Windows XP Pro with 1Gb RAM on a 2.6Ghz AMD chip; reserve approximately 1.8Gb of HDD space per server then add around 500Mb per SCSI disk in the fake disk array thereafter (we install 3 non-raid disks in the array).

At the same time, think about the server operating system you plan to install. We require three nodes in a virtual network, being:

1. Domain Controller
2. Cluster Node 1
3. Cluster Node 2

Take care with this step. The OS must support the cluster service (of course). The following example is using Microsoft Windows 2000 Advanced Server edition. Check carefully with your local Microsoft vendor/re-seller regarding personal use licensing issues, as we are installing three servers.

The DBA should be aware of the following limits:

Windows 2000 Advanced Server	Max 2 Nodes, 8 CPU's, 8 Gb RAM
Windows 2000 Data Center Server	Max 4 Nodes, 32 CPU's, 64 Gb RAM
Windows 2003 Enterprise Ed	Max 8 Nodes, 8 CPU's, 32 Gb RAM
Windows 2003 Data Center Ed	Max 8 Nodes, 32 CPU's, 64 Gb RAM

Paul Thurrott has a great website for feature comparisons and Windows in general—

http://www.winsupersite.com/showcase/winserver2003_editions.asp

http://www.winsupersite.com/reviews/win2k_datacenter.asp



"Microsoft does not support issues that occur in Microsoft operating systems or programs that run in a virtual machine until it is determined that the same issue can be reproduced outside the virtual machine environment.", See MS Support #273508 for the full support note. Do note that many corporate resellers of VMWARE will provide this level support.

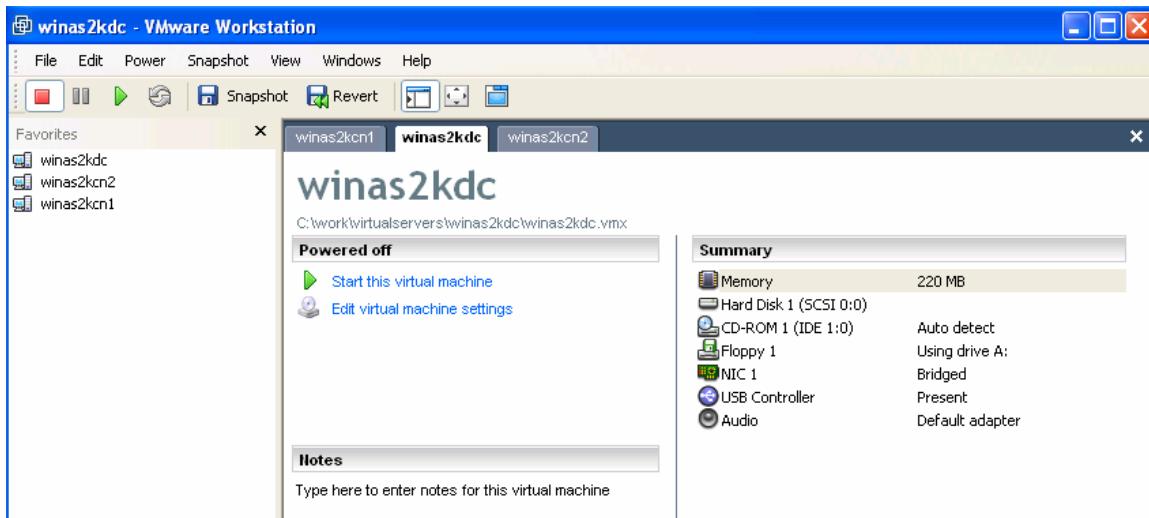
Step 2. Create the Virtual Servers

Run VMWARE, select File → New Virtual Machine, select *typical* machine configuration and from the drop down list we pick Windows 2000 Advanced Server. Enter the name of the virtual server (it makes no difference to the host name of the server itself) and the location of you server on disk. I have placed all virtual servers at:

C:\work\virtualservers\

VMWare will pick up all current PC hardware and provide a summary, along with RAM (memory) it plans to use on the virtual machines startup. Generally no changes are required. Do not alter the network unless you are very comfortable with vmware network bridging.

Below is a screen shot of the server before startup: You can create all of your virtual servers now if you like to save you repeating the process, but we only start off with the domain controller. The three servers I have created are:



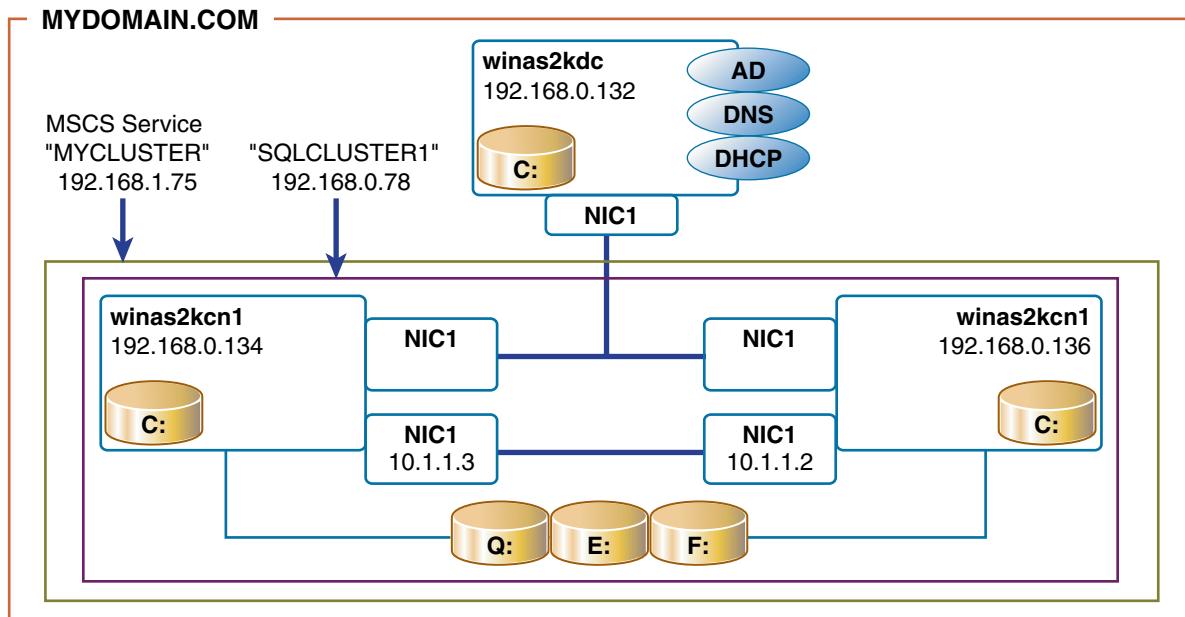
1. winas2kdc – domain controller
2. winas2kcn1 – cluster node 1
3. winas2kcn2 – cluster node 2



We do not use VMWARE server templates (also known as golden templates) as it requires extensive coverage by system administrators (not me!). The golden template is a “known image” (or copy) of a virtual server at some point in time. The image includes the base software any of your normally provisioned servers would include and is carefully crafted to remove networking issues when new servers are created from the template and put online.

Click on *edit virtual machine settings*, and reduce the Guest Size (MB): value down to around 150 to 220Mb of RAM. No other change is required.

The “end game” is this configuration:



My naming conventions for the servers—please do not take them to heart; they are only examples and care must be taken using the OS installed as part of the hostname for obvious reasons.

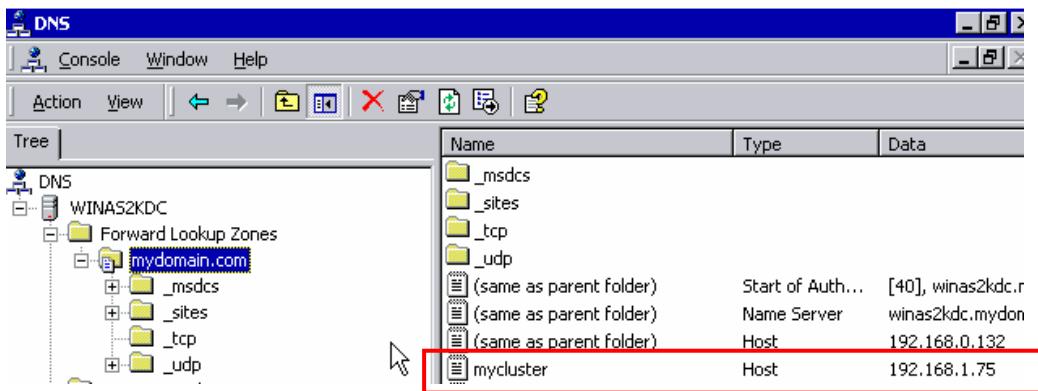
Step 3. Build Your Domain Controller

Place the Windows server disk in the cd-rom and start the virtual machine. From here complete a standard installation of the operating system. The VMWare software will not affect your current PC operating system, so don't concern yourself about the steps related to formatting disks. Ensure NTFS is selected for the file system.

- When asked, you have no domain or work group to join. If prompted to enter a new one, enter the word: MYDOMAIN.
- Leave all network options standard (typical). Do not alter them yet.
- When the install is complete, login to the server as *administrator*.

By default, the Windows 2000 Configure Your Server dialog is shown with a range of options. First step is to install active directory. Select this option and follow the prompts. When asked for the domain name, enter MYDOMAIN.COM and carry on with the defaults from there.

Run the DNS management tool. We need to configure a zone for *mydomain.com*. Expand the tree in the left hand pane and right click properties on Forward Lookup Zones and select *new*. I have selected *active directory integrated*. Once created, select the zone, right click for properties and click *add host*. We are going to add a host entry representative of the virtual IP of the cluster itself. This host/server will be called *mycluster* (or *mycluster.mydomain.com*). One of the member servers that are active at the time in the cluster will take on this IP as we will see later. The IP is 192.168.1.75

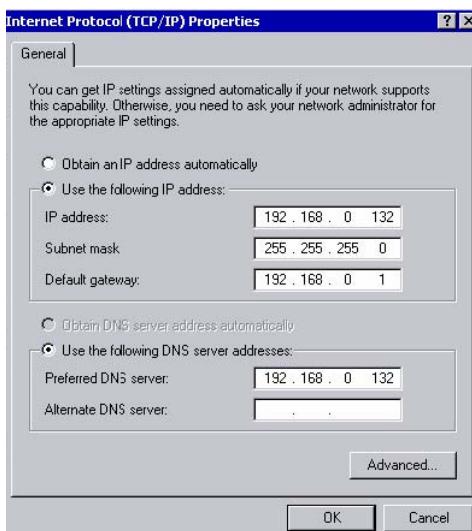


Exit from the DNS management utility.

Before we continue, minimize all windows, and on the desktop, select properties of the *My Network Places*, I have only one NIC defined for this virtual server so I see:



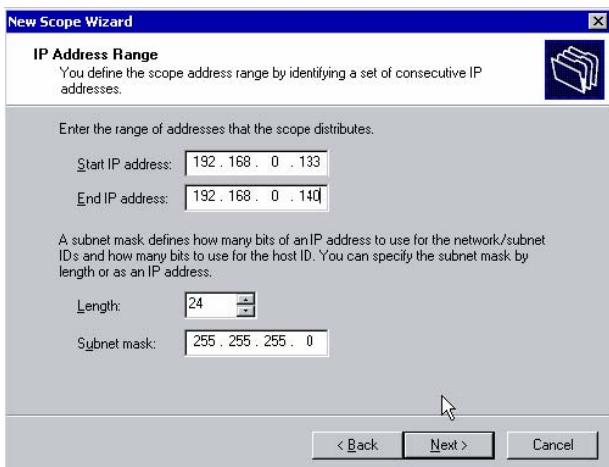
Select properties of the LAN connection, my TCP/IP properties are as follows:



The default gateway was actually picked up from my local PC settings, and this is the gateway properties used for my internet connection sharing with another PC on the network. Note the IP address and the DNS entries, as expected for the role of the server.

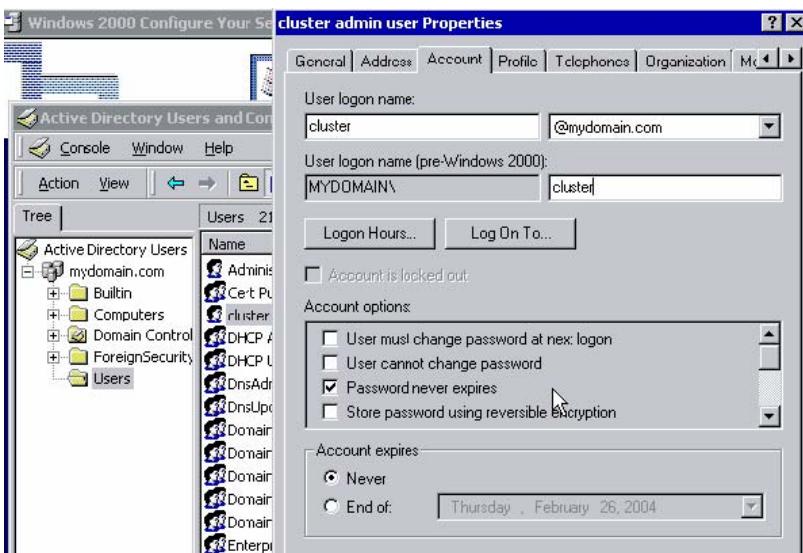
Under the networking option in Windows 2000 Configure Your Server installs DHCP in order for the domain controller to allocate IPs to the two member servers when they come online. The IPs will be reflected in the DNS as the servers are built and come online into the domain, as one would expect of the DNS.

Create a new scope within DHCP, I called it mynetwork. I have allocated a DHCP address pool with a range between 192.168.0.133 to 192.168.0.140 (any server member server will get a IP from this range).



Clicking *next* will ask for any exclusions (or reservations). Ignore this, and retain the default lease period of 8 days to complete the DHCP configuration.

Close the DHCP management utility. Go back to Windows 2000 Configure Your Server and select Active Directory and Manage. We need to create another user with administrative rights. This is our *cluster admin* user:



The username is *cluster* (for want of a better word), and is a member of the administrators group. Exit once this is done.



Using DHCP for your cluster nodes is NOT recommended and is far from best practice. I have used it as a demonstration; revert to fixed IPs and remember to update the DHCP IP range as need be.

Step 4. Build member server 1 (node 1 of the cluster)

Leave the domain controller up and available. We now configure a member server. This will be node 1 of our cluster. Before we start this virtual server and carry on with the installation, we need to setup a number of fake SCSI disks (to simulate a disk array in which the cluster will add as disk resources) and another NIC for the clusters private network (remember I only have 1 physical NIC in my server, so I need to add a virtual one in VMWARE).

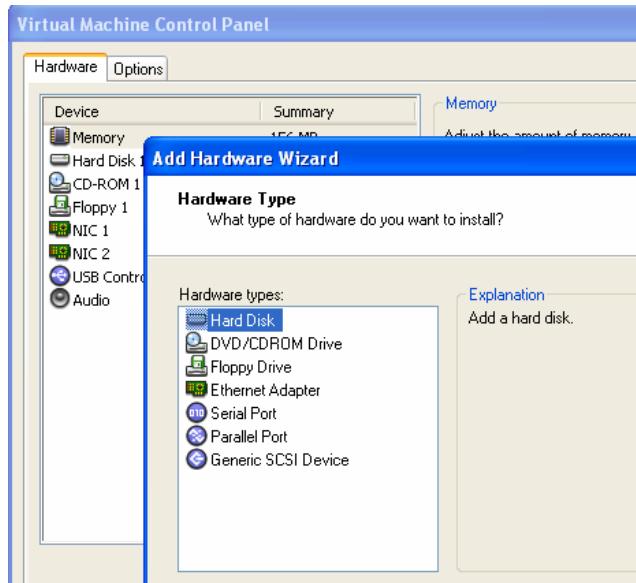


try and keep installation directories and servers identical in the cluster node installation to avoid any later issues.

Adding SCSI Disks

To simulate a disk array/SAN, or a “bunch of disks” for the cluster, VMWARE allows us to create virtual disks (simply files in a directory) and add them to both servers in the cluster via a disk.locking=false clause in the .VMX file for each node.

Before we continue, the solution presented here does not use VMWARE edit server options to add a hard disk:



After some effort, I found the VMWARE SCSI drivers did not load on the node, and could not re-install them. To get around this, visit Rob Bastiaansen's website to download *plain disks*:

<http://www.robbastiaansen.nl/tools/tools.html#plandidisks>

And extract the disks into a folder called:

C:\work\virtualservers\disks

Edit the .PLN file for the disk and change the path to the disk (.DAT) file; do not alter any other setting in the .PLN file. I copied the file twice, giving me three 500Mb plain disks.

Name	Size	Type
plainscsi1500mb	1 KB	PLN File
plainscsi2500mb	1 KB	PLN File
plainscsi3500mb	1 KB	PLN File
plainscsi1500mb1	512,000 KB	DAT File
plainscsi2500mb	512,000 KB	DAT File
plainscsi3500mb	512,000 KB	DAT File

For each .VMX file on our virtual server cluster nodes (not our DC):

1. shutdown the node
2. locate and edit with notepad the .VMX file for the node

 winas2kcn1 2 KB VMware Configuration File 26/0:

3. Add the following:

```
<after the memsize option add..>>
disk.locking = "FALSE"
<<add these near the bottom of the file>>
scsi1.present="true"

scsi1:0.present= "true"
scsi1:0.deviceType= "plainDisk"
scsi1:0.fileName = "C:\work\virtualservers\disks\plainscsi1500mb.pln"

scsi1:1.present= "true"
scsi1:1.deviceType= "plainDisk"
scsi1:1.fileName = "C:\work\virtualservers\disks\plainscsi2500mb.pln"

scsi1:2.present= "true"
scsi1:2.deviceType= "plainDisk"
scsi1:2.fileName = "C:\work\virtualservers\disks\plainscsi3500mb.pln"
```

Before we start the first server of our cluster, we need to add another NIC.

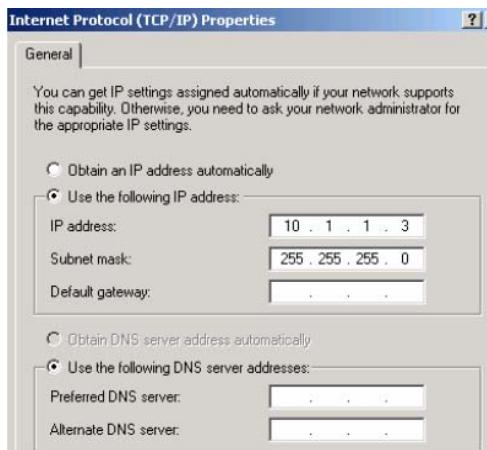
Adding another NIC for the Private Network

For each server that will be a node in the cluster, we require two NIC's:

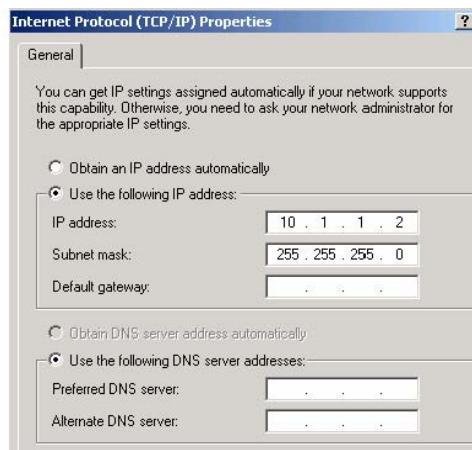
1. For the private network between cluster nodes
2. For the public facing communication to the node

I assume that most people have at least one NIC, if not, use VMWARE and the edit hardware option to add the two NIC's. Leave options as default. For each node when booted, set up IP properties as such:

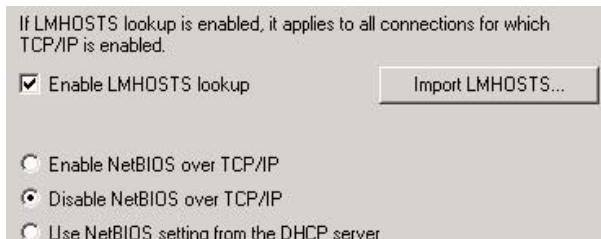
Node 1



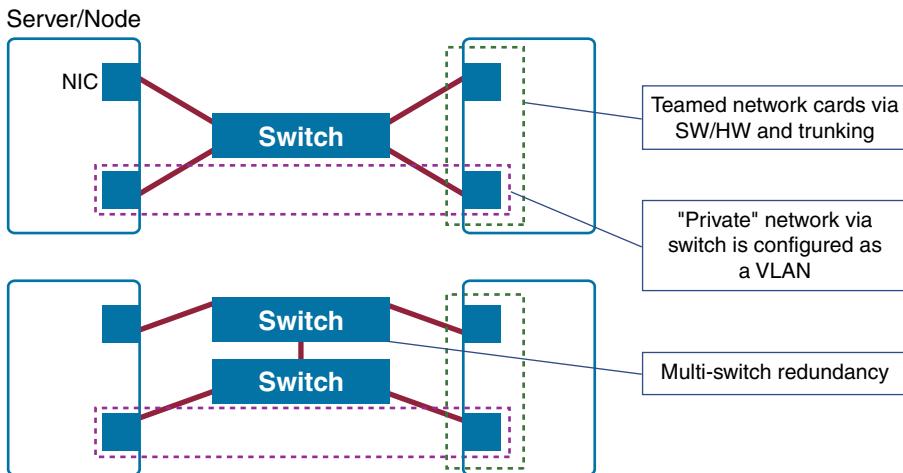
Node 2



For each node, disable netbios over TCPIP for DHCP:



The actual networking and connection of the server's NICs can be varied, and I suggest that the Network Administrators/Communications Specialists be approached well before the purchase of any hardware. At a high level, we tend to this sort of configuration:



Also, be aware of the following:

"By agreement with the Internet Assigned Numbers Authority (IANA), several IP networks are always left available for private use within an enterprise. These reserved numbers are:

- 10.0.0.0 through 10.255.255.255 (Class A)
- 172.16.0.0 through 172.31.255.255 (Class B)
- 192.168.0.0 through 192.168.255.255 (Class C)

You can use any of these networks or one of their subnets to configure a private interconnect for a cluster. For example, address 10.0.0.1 can be assigned to the first node with a subnet mask of 255.0.0.0. Address 10.0.0.2 can be assigned to a second node, and so on. No default gateway or WINS servers should be specified for this network."

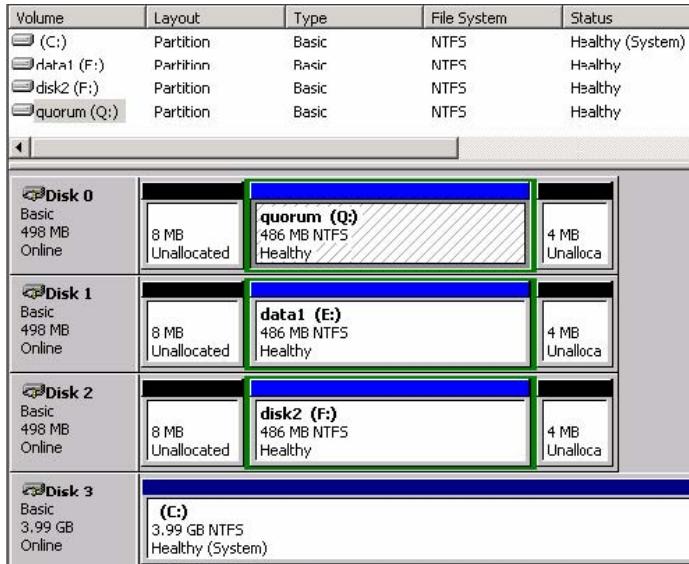
Prepare your SCSI disks

With the disks and NICs added, we need to partition and format the disks.

1. Start your DC server first.
2. Start server 1 (winas2kcn1), leave the other server down for the time being.
3. When booting the server, notice in the bottom right hand corner the list of resources for this server (ignore floppy disk drive messages):



4. Login to the server using the domain administrator account e) When you login, the server will detect new hardware and install the VMWARE SCSI drivers for you.
5. My computer → Manage.
6. Storage → Disk Management.
7. For each of the three disks listed, DO NOT MAKE THEM DYNAMIC DISKS (also known as disk signing). Simply create a partition, and format using NTFS, extended partitions.
 - a) My disks are mapped to Q:\ (disk1, will be Quorum disk), E:\ and F:\



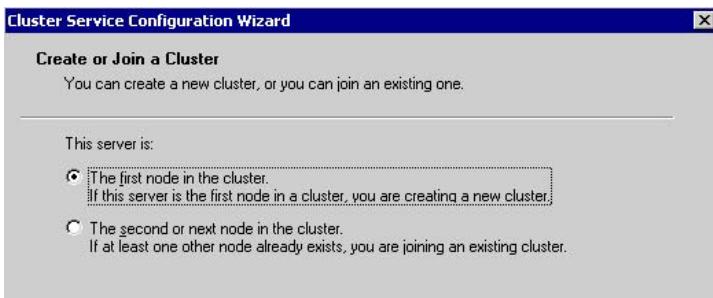
8. Repeat this for the other cluster node – winas2kcn2.
9. Once done, shutdown server 2 – winas2kcn2.

Install Cluster Services on Server (node) 1

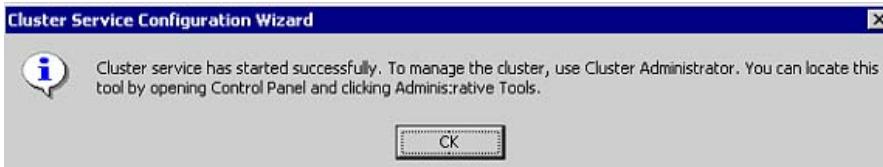
Now the disks have been added and prepared, and we have two NIC's defined for our server nodes:

1. Start your DC server first – already started from previous step.
2. Start server 1 (winas2kcn1) – already started from previous step.
3. Leave server 2 (winas2kcn2) down.
4. Login with your domain administrator account.
5. Navigate to control panel, add/remove programs, add/remove windows components. The cluster services option will be shown. Run this option.

6. This will be the first node in the cluster:



7. Enter the name of your cluster: MYCLUSTER (the virtual host entry we made in DNS).
8. Enter the username and password of our cluster administrator user. We called it "cluster", and the domain name MYDOMAIN.
9. Your SCSI disks are shown. We will retain all three disks for this managed cluster resource.
10. Pick our Q:\ as the quorum drive (holds all cluster checkpoint and log files essential to the cluster resource group).
11. Next we are asked about our network configuration. At the first prompt pick your private network card and select the radio button option "internal cluster communications only" (private network).
12. Then we are asked about our public network. Select "all communications (mixed network)" in the radio button. Addressing is self explanatory.
13. Click through and the cluster service should be up and running:

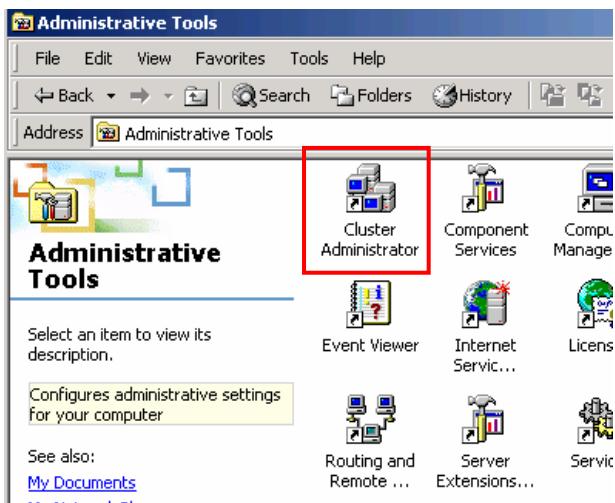


The following examples are using Windows 2000. Do note that in Windows 2003, if no shared disks are detected, then a local quorum will be automatically created; you can also create a local quorum resource after cluster installation. The local quorum information will be stored in %systemroot%\cluster\MSCS. Of course, being local it will remain a single node cluster installation.

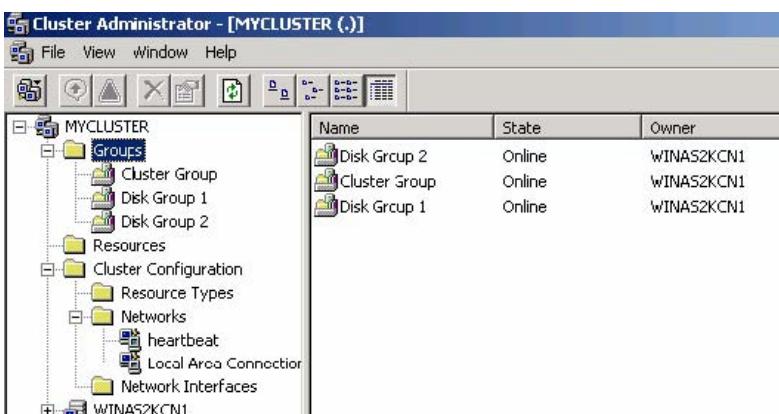
To create the quorum with a local resource, use the /fixquorum switch. The creation is also covered in MS Support article #283715.

Validate Node 1 in the cluster via Cluster Administrator

Open the cluster administrator program:



And we see this. Notice our node 1 (winas2kcn1) is the owner of the resources? Click through options to familiarize yourself with the cluster.



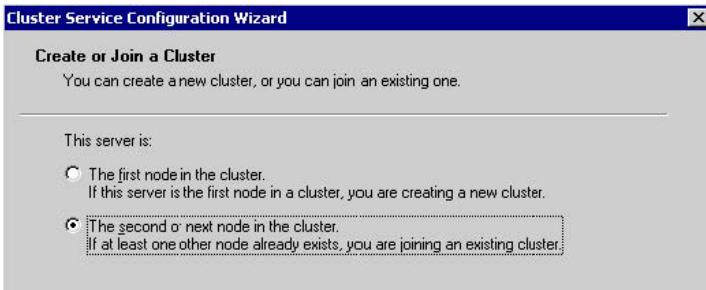
Run IP config (*ipconfig* command via the DOS prompt), notice that this server has the IP address of *mycluster* (as defined in DNS).

```
Connection-specific DNS Suffix . : mydomain.com
IP Address . . . . . : 192.168.1.75
Subnet Mask . . . . . : 255.255.255.0
IP Address . . . . . : 192.168.0.136
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.0.1
```

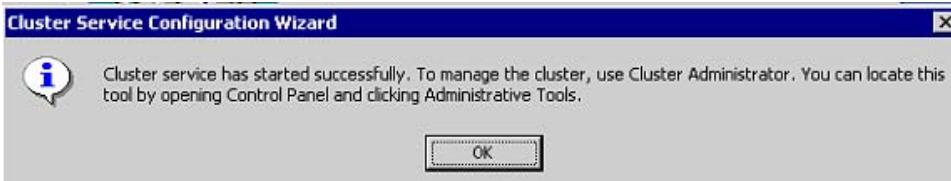
Step 5. Build Member Server 2

With the DC (domain controller) and node 1 online, we have added both disks to both servers and configured their private networks. Now we complete installation by installing cluster services on node 2.

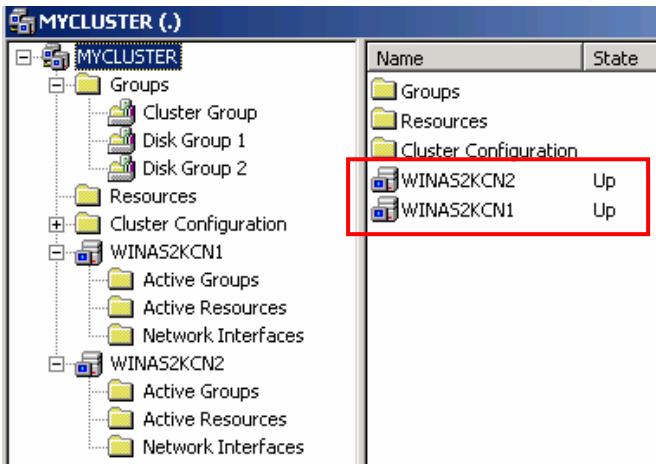
Boot node 2. Using the control panel wizard continue with cluster service installation by selecting the *second or next node* in the *cluster*:



You will be asked the name of the cluster to join. DO NOT USE "mycluster". Use the name of server 1 (winas2kcn1) that currently owns the cluster. Use our cluster user account to authenticate.



Run cluster administrator to verify your node:



Step 6. Install SQL Server 2k in the Cluster (Active/Passive)

Our VMWARE cluster is now up and running. We will install SQL Server 2k in active/passive mode in the cluster (i.e. a single named instance on the cluster, not multiple instances running on both nodes). Remember than only one server (node) in the cluster can open and read/write to/from the database files at any one time, no matter how many nodes you have in the cluster. An active/active cluster simply means two *separate* instances with their OWN disks and database files are running on both nodes of a two server cluster (for failover reasons, DON'T install two default instances on each node—simple but common mistake).



SQL Server 2000 SP3 or higher is only supported under a Windows 2003 cluster.

I will not discuss the wide and varying issues with clusters and sql server 2k installation. Please visit Microsoft Support and MSDN for detailed articles covering a majority of them. Also take the time to visit www.sql-server-performance.com and www.sqlservercentral.com for their feature (and free) articles on clustering sql server.



You can only install SQL Server Enterprise Edition on a cluster.

You will require:

1. SQL Server 2k Enterprise Edition
2. Latest service pack



The setup of COM+ on Windows 2003 is completely different to running comclust.exe on Windows 2000. The administrator must manually create the MSDTC group, including IP, Network Name and the DTC resource via the Cluster Administrator.

Follow these steps to install the DBMS (assumes all servers are up and installed/configured as described earlier).

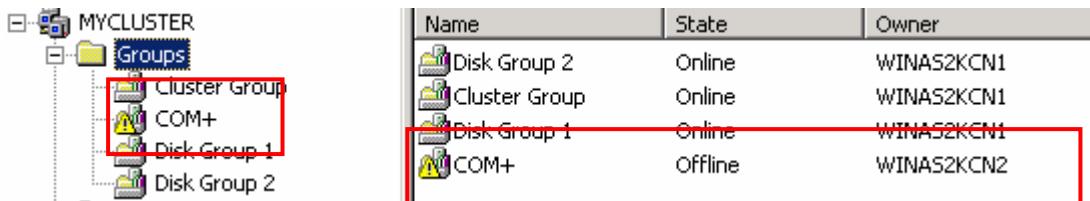
1. On each cluster node, run *comclust.exe*:

```
C:\>comclust
Setting up MS DTC.
Setup has sucessfully populated configuration information to allow MS DTC to run
on this cluster node. Please run setup on all other nodes in the cluster before
continuing.
Setting up: COM+ -- CLB
WARNING: This machine is not a Component Load Balancing server.
The CLB resource will not be added to the COM+ cluster group.

C:\>_
```

2. From the primary node in the cluster, run *cluster administrator*

3. Verify MSDTC is clustered correctly by navigating to the resources folder. Check each node to ensure the *distributed transaction coordinator* service is running. Nodes with a DTC issue are shown in the administrator tool:



My node 2 locked up with the CLB setup. A CTRL-C did the trick and a manual bring resource online via cluster administrator worked without failure. This e-book does not cover such issues.

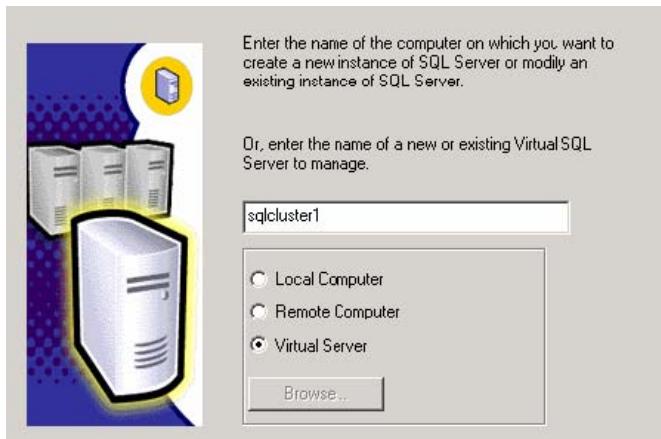
4. It is generally recommended you turn off all services except the following before installation (confirm between OS releases):

Alerter
Cluster Service
Computer Browser
Distributed File System
Distributed Link Tracking Client
Distributed Link Tracking Server
DNS Client
Event Log
IPSEC Policy Agent
License Logging Service
Logical Disk Manager
Messenger
Net Logon
Plug and Play
Process Control
Remote Procedure Call (RPC) Locator
Remote Procedure Call (RPC) Service
Remote Registry Service
Removable Storage
Security Accounts Manager
Server
Spooler
TCP/IP NetBIOS Helper
Windows Management Instrumentation Driver Extensions
Windows NT LM Security Support Provider
Windows Time Service
Workstation

5. Before we install SQL Server 2k – use cluster administrator to offline DTC, right click for resource properties:

Name	State	Owner	Group
Disk Q:	Online	WINAS2KCN1	Cluster Group
Disk F:	Online	WINAS2KCN1	Disk Group 2
Cluster IP Address	Online	WINAS2KCN1	Cluster Group
Cluster Name	Online	WINAS2KCN1	Cluster Group
Disk E:	Online	WINAS2KCN1	Disk Group 1
MSDTC	Offline	WINAS2KCN1	Cluster Group

6. Place your SQL Server disk in the CD-ROM, and begin installation on node 1 (win2kascn1)
7. You are prompted, by default for the virtual server name, we will call this *sqlcluster1*:



8. We are prompted for the virtual IP of our sql server instance within the cluster. The resource will be failed over as required in the cluster. Before I enter the IP here, goto to your domain controller server, run DNS, and add a new Host record for this server. Double check your DHCP settings also to ensure the integrity of the allocated IP:

Host (A)	Security
Parent domain:	
<input type="text" value="mydomain.com"/>	
Host (uses parent domain if left blank):	
<input type="text" value="sqlcluster1"/>	
IP address:	
<input type="text" value="192 . 168 . 0 . 78"/>	
<input type="checkbox"/> <u>Update associated pointer (PTR) record</u>	

Failover Clustering

Enter virtual server information.		
Virtual server name :	SQLCLUSTER1	
IP address:	<input type="text" value="192 . 168 . 1 . 78"/>	
Subnet:	<input type="text" value="255 . 255 . 255 . 0"/>	
Network to Use:	<input type="text" value="Local Area Connection 2"/>	
<input type="button" value="Add"/> <input type="button" value="Remove"/>		
IPAddress	Subnet	Network
<input type="text"/> <input type="button" value="◀"/> <input type="button" value="▶"/>		
<input type="button" value="Help"/> <input type="button" value="< Back"/> <input type="button" value="Next >"/> <input type="button" value="Cancel"/>		

9. Pick our default data file disk, don't use your Q:\ (quorum drive); we will use F:\
10. Our two nodes will be shown. We want all servers to be used; no changes required.

11. Go back to your domain controller. Create a new domain user called "SQLServerAdmin" and add it to the administrators group. This is the service account that will run our SQL Server clustered instance.

Create in: mydomain.com/Users

First name: SQLServerAdmin Initials:

Last name:

Full name: SQLServerAdmin

User logon name: SQLServerAdmin @mydomain.com

User logon name (pre-Windows 2000): MYDOMAIN\SQLServerAdmin

for each node, add this user in the *administrators* group:

Look in: mydomain.com

Name	In Folder
SQLServerAdmin (SQLServerAdmin@mydomain.com)	mydomain.com/Users
Domain Computers	mydomain.com/Users
Domain Controllers	mydomain.com/Users
Schema Admins	mydomain.com/Users
Enterprise Admins	mydomain.com/Users
Cert Publishers	mydomain.com/Users
Domain Admins	mydomain.com/Users

Add Check Names

SQLServerAdmin (SQLServerAdmin@mydomain.com)

General

Administrators

Description: Complete and unrestricted access to the computer/domain

Members:

- Administrator
- MYDOMAIN\cluster
- MYDOMAIN\Domain Admins
- MYDOMAIN\SQLServerAdmin

and continue on with the SQL Server installation with this domain account:

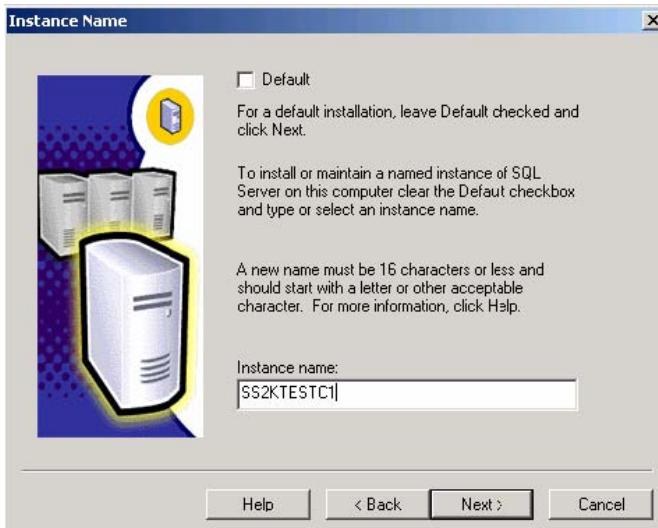
Enter a user name and password that is a valid administrator account for all nodes in the cluster system.

Username: SQLServerAdmin

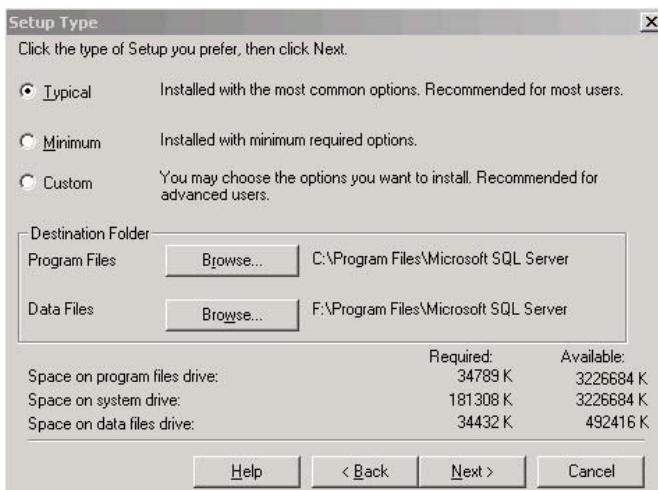
Password: ****

Domain: MYDOMAIN

12. Enter the instance name, we will use a named instance:



13. You are shown the standard summary screen:



We do not install the binaries on the clustered disks. They remain local to the node. Check *Custom* and continue.

14. Select your components. Note that Full Text Indexing is not on the list. This will be installed by default as a cluster resource.

15. Select your service startup account:



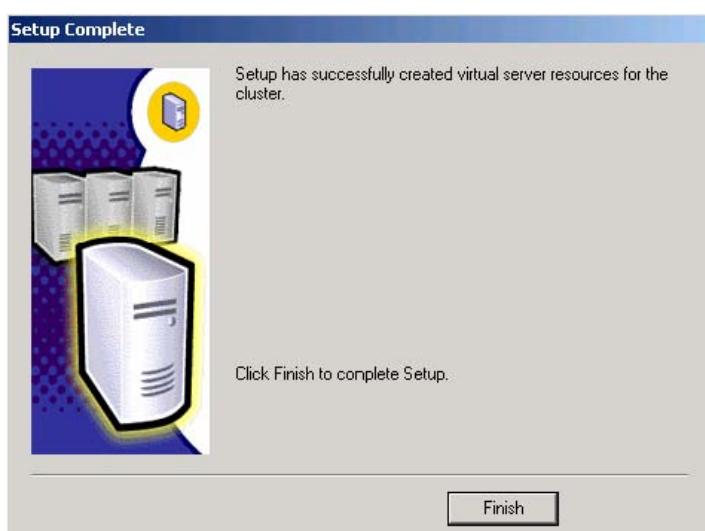
16. Then run through your authentication mode, collation, and network library properties. I leave all default but choose mixed mode.

 For each instance, ensure you fix the PORT properties over TCPIP. Do not let SQL Server automatically assign the port as it can change between nodes in the cluster.

17. Setup begins, node 1 then automatically on node 2:

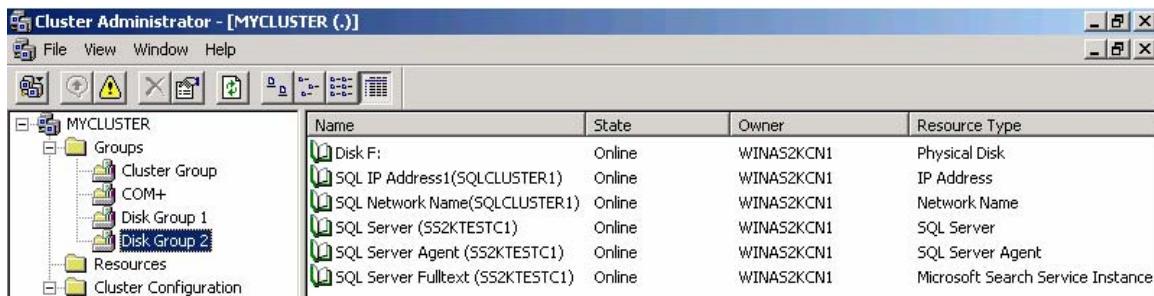


18. Complete



19. Reboot Node 1, then Node 2

20. Run Cluster Administrator on either node to verify the cluster:



Note that all resources are under disk group 2. Move them with caution as all are dependent on one another for a successful failover.

For the active node (node 1), check its IP configuration:

Connection-specific DNS Suffix	: mydomain.com	SQL Server virtual IP
IP Address	: 192.168.0.78	Cluster virtual IP
Subnet Mask	: 255.255.255.0	Node 1's DHCP IP
IP Address	: 192.168.1.75	
Subnet Mask	: 255.255.255.0	
IP Address	: 192.168.0.136	
Subnet Mask	: 255.255.255.0	
Default Gateway	: 192.168.0.1	

You should be able to PING the:

21. server cluster IP address – 192.168.1.75
22. server cluster name (network name of the cluster) – 192.168.0.78



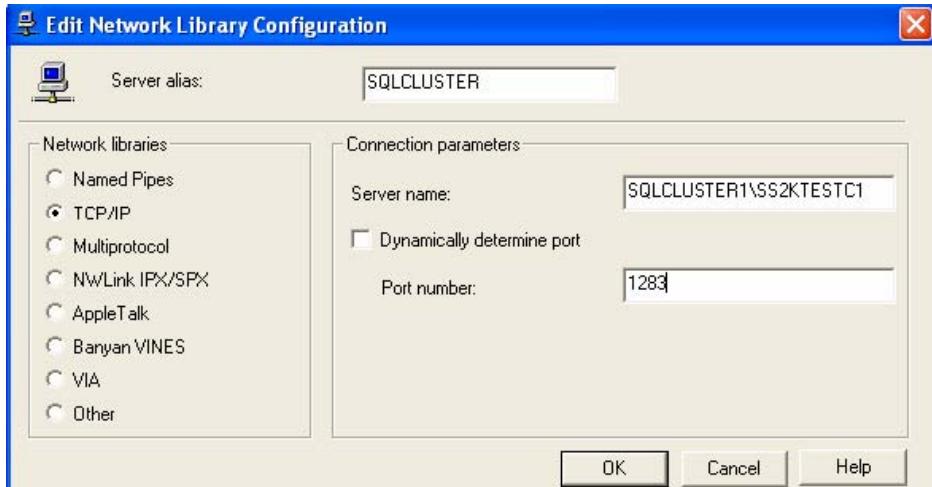
In an active/active cluster, remember that the total memory usage of all instances should be less than the total memory of a single node. If all instances fail to one node then it must be able to support, in terms of memory, all databases. If not you will experience severe paging and very poor performance. Also be reminded that a maximum of 8 instances can run on a single node, and only one default instance may exist within the entire cluster.

Test Connectivity

Our SQL Server Named instance is referred to as: SQLCLUSTER1\SS2KTESTC1

The port and this instance name can be verified by running the server network utility on either node in the cluster.

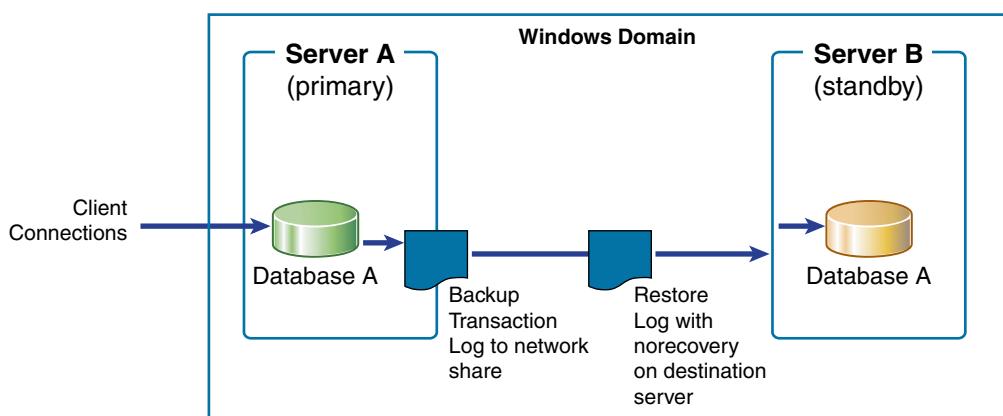
From my PC (that is running VMWARE), install the client tools (client network utility, query analyzer), and create an alias via the client network utility as such:



If you cannot connect, double check the server name, the port, the enabled protocols, attempt to ping the sql server virtual IP and the cluster IP.

High Availability using Log Shipping

The process in SQL Server works at a database level, not at a global instance level as you will see later. The aim here is to create a *warm standby* server in which one or more databases are permanently in recovery mode. The source database *ships* transaction log backup files to the destination server and we restore (with no recovery option) the logs sequentially (in order). If there is a failure on the source server we attempt to recover the last of the transaction logs, ship it to the destination server and complete the database recovery. Once done, we send client connects to the new server and they continue to work with little or no data loss.



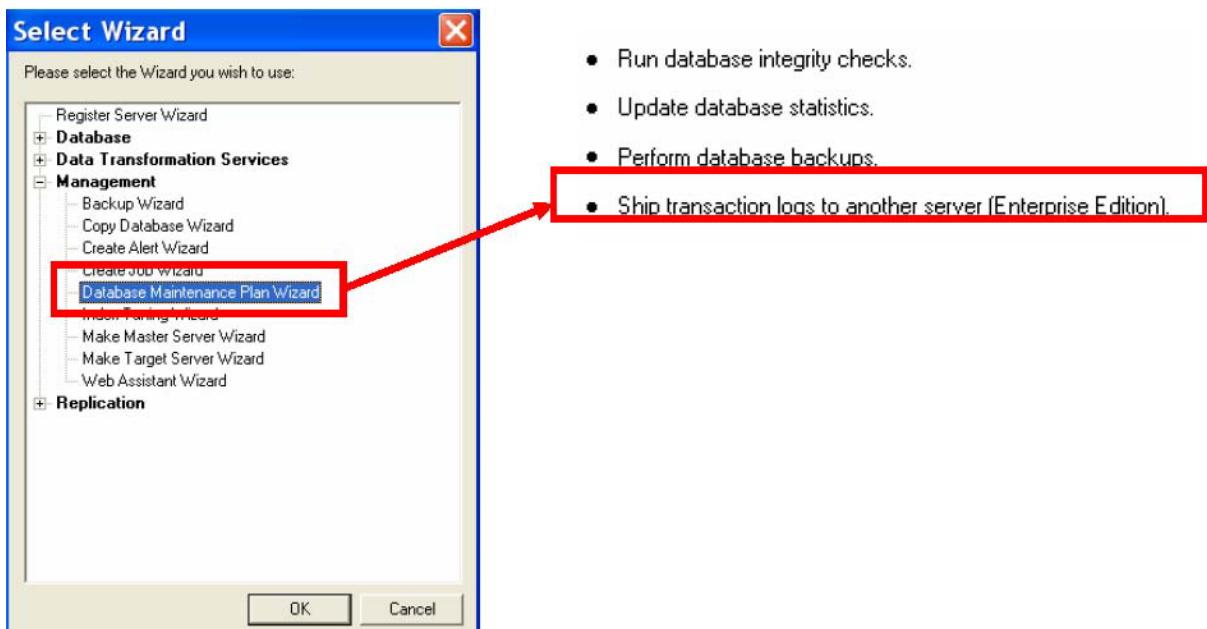
We have not diagrammed the full database backups that must also be log-shipped to start the process.

The DBA can use a custom written script or the SQL Server Wizard to do the shipping. The shipping in most cases will be to a server on the same Windows 2k network domain, but there is no reason why VPN tunnels over large distances or other remote scenarios are not utilized.



You do not require enterprise edition of SQL Server for log shipping.

If you further help over what has been provided in this ebook for custom shipping, then get hold of the SQL Server 2000 Resource Kit from Microsoft.



The supplied wizard provides all necessary steps to setup log shipping for selected databases. Before doing so remember the following:

- document the process before running the wizard
- how will the client application components detect the failover?
- databases must be in full or bulk-logged recovery mode
- ensure instance names are the same on source and destination servers
- pre-determine the log backup schedule for the source database
- pre-setup the transaction log backup directory via a network UNC path (referred to by both source and destination DTS jobs)
- create DTS packages on both servers and use the transfer logins job to facilitate the transferring of login information between shipped servers.
- Must run the wizard using a login that has sysadmin system privileges

Once the log shipping DTS (maintenance plan) has been created, go to the Management Folder in EM and select properties of the log-shipping job to view the process/steps.

See reference (56) for a documented “How-To” for log shipping in a SQL Server 2k environment using the Wizard.

Manual Log Shipping – Basic Example

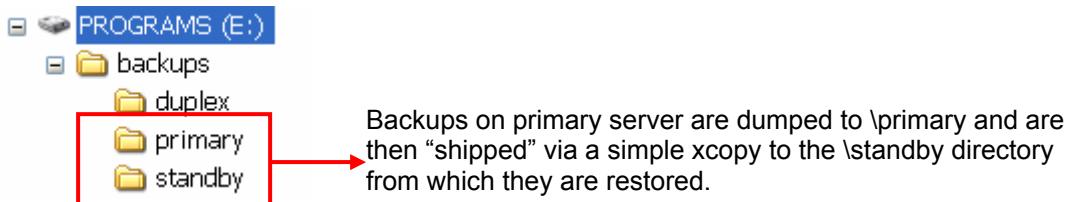
Here we will discuss a working example that was coded using two stored procedures, a linked server, two DTS packages and (optionally) pre-defined backup devices.

This example revolves around a single server with two named instances:

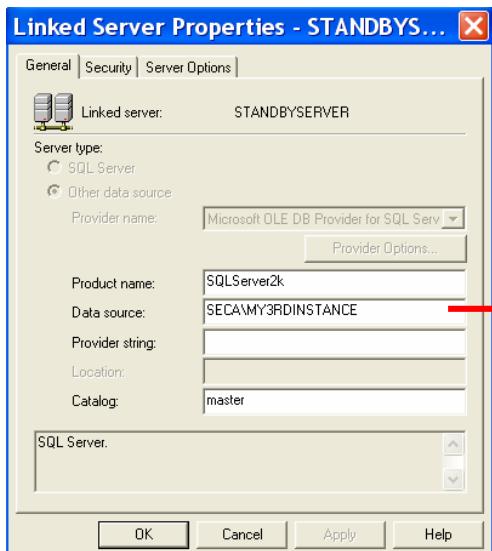
Primary Database Instance – SECA\MY2NDINSTANCE

Standby Database Instance – SECA\MY3RDINSTANCE

1. Setup disk locations for primary and destination backups and test with SQL Server service account.



2. Pre-determine what account will be used for doing the backups, establish the linked server and restore backups on the primary and standby databases. I recommend that you create a new account with *sysadmin* rights on both servers to facilitate this.
3. Setup Linked Server on primary server to destination server



We will link from the primary server over to the standby server via the account specified in step 2. Data source is the server-instance name we are connecting too.

Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password
sa	<input type="checkbox"/>	sa	xxxxxx

For a login not defined in the list above, connections will:

- Not be made
- Be made without using a security context
- Be made using the login's current security context
- Be made using this security context

In this case we are using the SA account which is not best practice, but will suffice for this example.

Option Name	Value
Collation Compatible	<input type="checkbox"/>
Data Access	<input checked="" type="checkbox"/>
RPC	<input checked="" type="checkbox"/>
RPC Out	<input checked="" type="checkbox"/>
Use Remote Collation	<input checked="" type="checkbox"/>
Collation Name	

Apart from the SA account mapping above, no other mapping will be valid.

Ensure remote procedure call options are set to facilitate the calling of t-sql stored procedures on the standby server from the primary server.

* If you are using EM, and have registered the server under an account not mapped in the linked server, then the linked server will not work for you. Check your EM registration before attempting to view tables/view under the linked server within EM.

4. Setup database backup devices on primary server

This is completely optional and really depends on the backup model you are using. SQL Server allows you to pre-create *backup devices*, once created, we can use the *logical name* for the backup device (which maps to a physical file) rather than the using the physical path and filename for the backup. This makes scripting much friendlier and easier to read/change.

Here we create two devices, one for full backups and the other for logs. See *management* folder within Enterprise Manager and the *backup* folder item item to create these:

Name	Physical Location	Device Ty...
logship_primaryserver_full_backup	E:\backups\primary\logship_primaryserver_full_backup.BAK	Disk Backup
logship_primaryserver_log_backup	E:\backups\primary\logship_primaryserver_log_backup.BAK	Disk Backup

5. Check primary server database recovery model. Select properties of the database via EM. Ensure the model is in line with the backups you will be log-shipping.

6. Write two stored procedures that will reside in the master database on the standby server for recovery of the FULL and LOG backups. The standby restore option is what tells SQL Server that the database is in warm standby mode.

```
CREATE PROCEDURE dbo.StandbyServer_restore_full_database_backup AS SET NOCOUNT ON
```

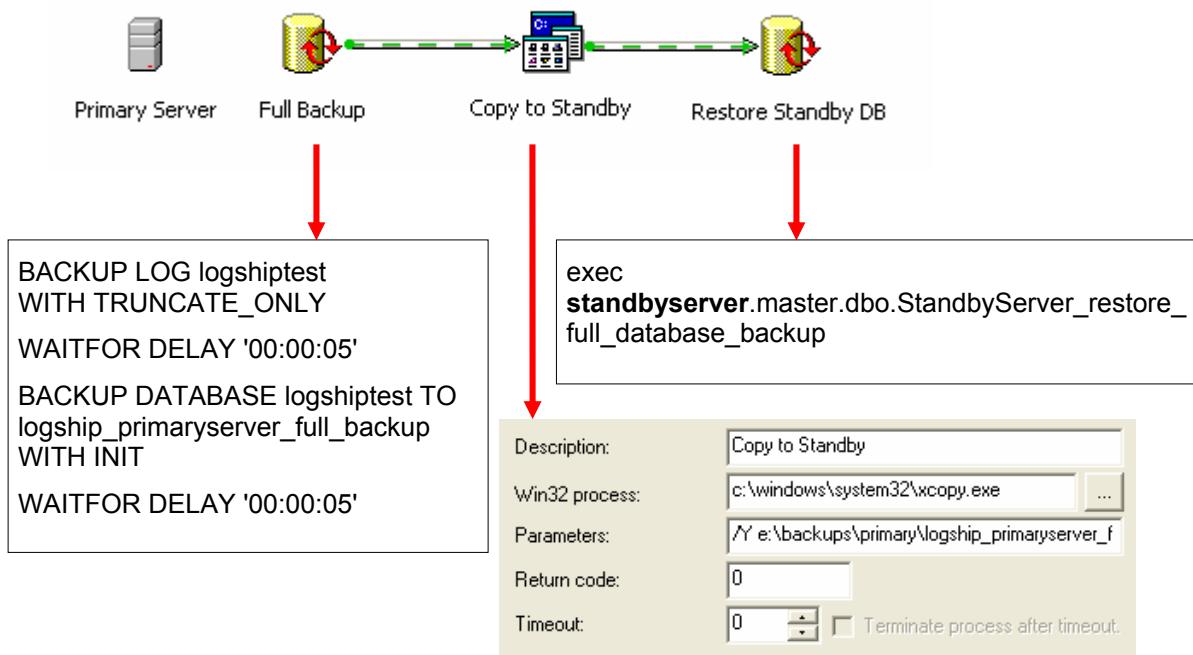
```
RESTORE DATABASE logshiptest FROM DISK =
'e:\backups\standby\logship_primaryserver_full_backup.BAK'
WITH
RESTRICTED_USER, -- leave db in DBO only use
REPLACE, -- ensure overwite of existing
STANDBY = 'e:\backups\standby\undo_logshiptest.ldf', -- holds uncommitted trans
MOVE 'logshiptest_data' TO 'e:\standbydb.mdf',
MOVE 'logshiptest_log' TO 'e:\standbydb.ldf'
GO
```

```
CREATE PROCEDURE dbo.StandbyServer_restore_log_database_backup AS SET NOCOUNT ON
```

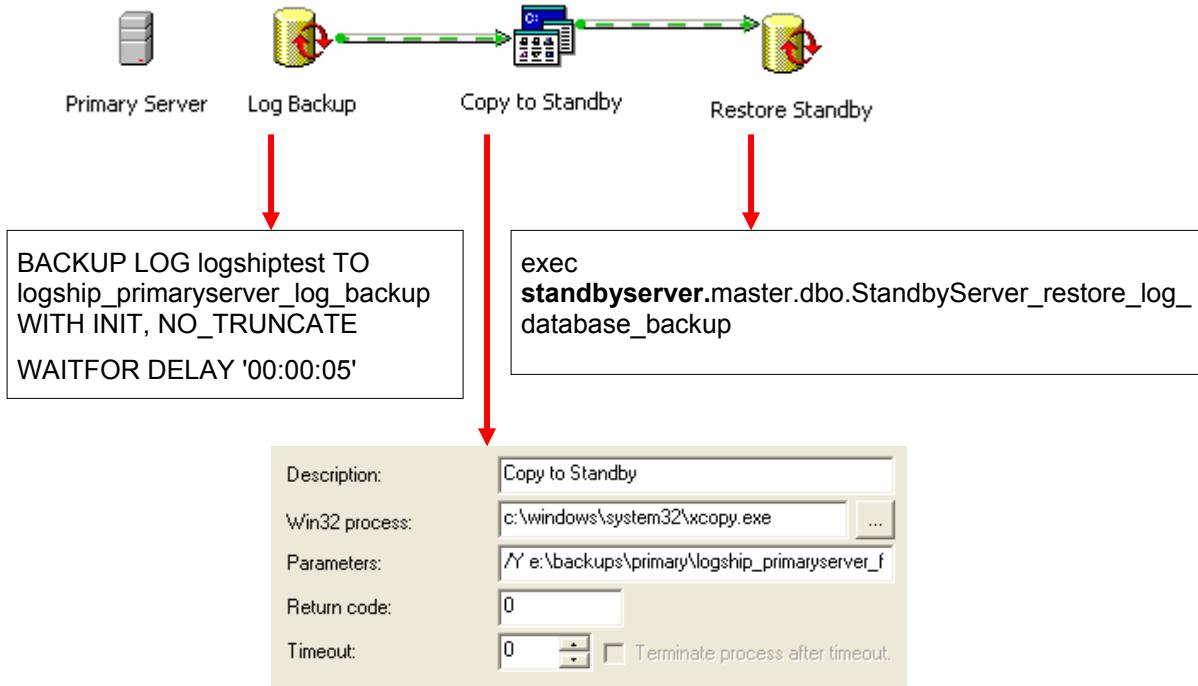
```
RESTORE LOG logshiptest FROM DISK =
'e:\backups\standby\logship_primaryserver_log_backup.BAK'
WITH
RESTRICTED_USER,
STANDBY = 'e:\backups\standby\undo_logshiptest.ldf' -- holds uncommitted trans
GO
```

7. Write two DTS packages on the primary server, one to do a full backup and the other a log backup.

Package 1 – Primary Server, Full Database Backup



Package 2 – Primary Server, Log Database Backup



8. Test Full then Log DTS routines and debug as required.
9. Schedule DTS packages.
10. Monitor.
11. On failure of the primary, do the following.

-- Login to primary server (depends on failure), and attempt to backup last database log file

```
BACKUP LOG logshiptest TO logship_primaryserver_log_backup WITH INIT, NO_TRUNCATE
```

-- Login into standby server

```
restore database logshiptest with recovery
```

```
Deleting database file 'e:\backups\standby\undo_logshiptest.ldf'.
RESTORE DATABASE successfully processed 0 pages in 4.498 seconds (0.000 MB/sec).
```

-- Ensure client connections are connection to the now live "standby" server.

Some thoughts about this setup:

1. The full and log backups are being appended to the same file, consider writing better backup routine on the primary server than produces separate files for each backup with a date/time stamp. Do this in a T-SQL stored procedure on the primary database and consider replacing the DTS copy command with a call to `xp_cmdshell` within the stored procedure.

2. If using a), parameterize the two recovery procedures on the standby server to accept the file path/filename of the file to be recovered. The routine in a) will have all this information and can pass it to the standby server without any problems.
3. Consider email on failure DTS tasks.
4. Consider how you will remove backups N days old?
5. Will the standby ever become the *primary* and effectively swap serve roles?

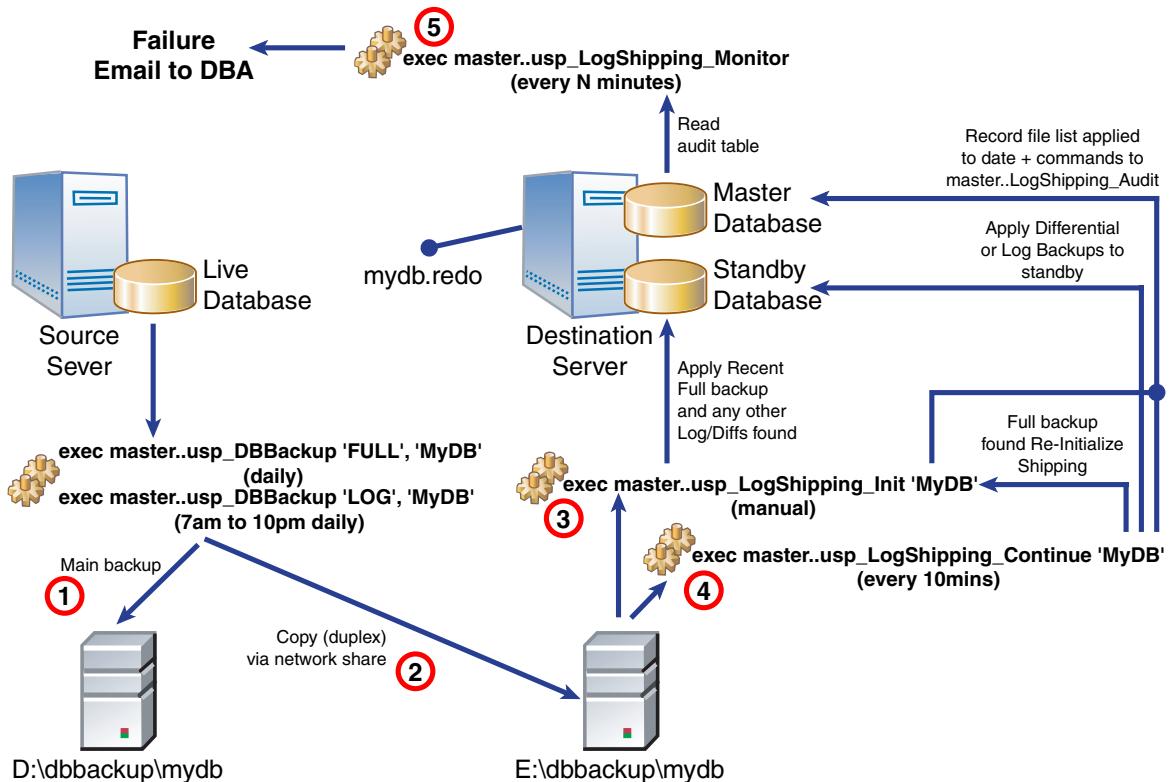
Custom Logshipping – Enterprise Example

The Enterprise Edition of SQL Server 2k includes the ability to configure and run log shipping; I find the process overly complex and a little restrictive in terms of control (i.e. I want to zip files, or FTP them to remote sources – we have no such options in the supplied method).

The scenarios I have implemented the custom log ship routines on are:

1. Heavily used OLTP database hosted on our “source server”
2. We currently backup to two destinations (called duplexing) on disk
3. We have a reporting requirement as well (destination server), and have chosen to *log ship* the database to this server.

The architecture is summarized below:



We take full and transaction log backups via a custom written stored procedure. The routine will dump the files to a disk array on the source server and optionally gzip (compress) them. The stored procedure will then copy the file to the remote server over a network share with appropriate service user privileges for the instance. Any failures are reported to the DBA via email. Also note that the *copy* to the remote server can be easily changed to an FTP and servers must be time-synchronized otherwise you will get the error “*There is a time difference between the client and the server*”.

At the destination server, we manually run the Initialize stored procedure. This routine will search the dump directory for specific pre and post fixed files specified by the procedures incoming parameters. Using xp_cmdshell and the dir command, we build up a list of files in the directory, locate the last FULL (unzip) then restore. We then search for a differential backup, apply this, and carry on applying the subsequent transaction log files. All restore commands used for the database are logged into a user defined master database table.

Finally, we call a simple monitoring stored procedure that looks for errors in the log table every N minutes (as defined by the DBA); emailing errors via CDO-SYS and your local SMTP server.

Advantages

- No requirement for linked servers
- Simply scripts with no complex lookups over the MSDB database, very easy to change and enhance
- Easy to setup and configure
- Will not, by default, force the overwriting of existing database files used by other databases
- Will search for full, differentials and logs and apply in correct order, so long as files copy OK

Disadvantages/Issues

- Requires a network share to copy files (can be a security issue)
- Can't pre-detect missed or missing files (as you could if you utilized the MSDB)
- Can't pre-detect invalid file sizes
- Does not do a quick header and file check and compare DBA's passed in parameters with
- Relies on the DBA to supply *move* commands for the restore as a parameter (see later), does not dynamically pick up database files/filegroups from the backup files themselves
- User sessions must be killed on the log shipped database before attempting the restore command
 - The only way I can see around this is via a physical log reader/parser program and the DBA runs SQL scripts rather than applying the log itself.

Configuration & Installation

All scripts are custom written and are typically stored in the *master* database of the instance.

Source Server (server-1)

The source server utilizes the following database objects:

- **DBBackup_sp** – master database—dumps full, log or differential backups to specified directory on source server, optionally zips files, emails on error, copies (duplexes) backups to destination server via UNC path, delete files older than N days.
- **SendMail_sp** – master database—utilizes *simplecdo.dll* (custom written VB COM that uses CDOSYS to send emails, can use JMail instead) to email the administrator on backup errors.
- **dtdelete.exe** – c:\scripts\—command line executable that will remove files from a directory (and recursively if so desired) that are N days old from the backup destination directory.
- **gzip.exe** – c:\scripts\—command line file compression utility for backup files.

Destination Server (server-2)

The destination server utilizes the following database objects:

- **usp_LogShipping_Init** – master database—run manually (1st time only or on serious error). Searches the incoming backup directory, applies most recent FULL backup, then last differential (if any) and applies subsequent transaction log files. Leaves database in *norecovery* or *standby* mode. Logs all recoveries to its audit table.
- **usp_LogShipping_Continue** – master database—as above, but searches for differentials and transaction logs only. If a full is found then Init is recalled to reapply the full backup again. Logs all recoveries to its audit table.
- **usp_LogShipping_Finish** – master database—manually called by the DBA, will finalise the recovery of a database and make it available for read/write. IMPORTANT - DBA must turn off log shipping jobs on the destination server before attempting this command.
- **usp_LogShipping_Monitor** – master database—reads the audit table below and emails the errors found to the DBA.
- **LogShipping_Audit** – master database—table that to which all recovery attempts are logged.
- **SendMail_sp** – master database—utilizes *simplecdo.dll* to email the administrator on backup errors.
- **gzip.exe** – c:\scripts\—command line file de-compression utility for backup files
- **usp_KillUsers** – master database—kills all users connected to a database for the instance.

Log Shipping Example 1 - Setup and Running

Server 1 (source)

Here we assume the server has been configured, instances and databases all running nicely and the DBA is now ready to sort out the backup recovery path for the database **MYDB**. This database has four file groups (1 data file per group) and a single log file. The database itself is approx 3.6 Gb in size, full recovery mode and requires point in time recovery in 10min cycles.

The DBA creates the following directory on the source server to hold backups:

```
d:\dbbackup\mydb\
```

There is ample space for 4 days worth of backups. LiteSpeed or other 3rd party backup products are not being used. The DBA wants the full backup files zipped, and files older than 4 days automatically removed.

On the remote server, the DBA creates the duplex directory: e:\dbbackup\mydb\. A share is created on the dbbackup directory called **standbydest** for one of a better word and NT security configured accordingly.

The DBA configures the following stored procedure to run 2 x daily for FULL backups via a DTS job:

```
exec DBBackup_sp 'full', 'mydb', 'c:\scripts', 4, 'c:\scripts', 1,  
'd:\dbbackup\mydb\','\\server2\standbydest\mydb\', 'support@chriskempster.com', 'Y'
```

We are running full backups at 6am and 8pm to cover ourselves nicely in terms of recovery (not shown here, in DTS). We chose not to run differentials and are happy with recovery times in general. The script above and its parameters tells the backup where our gzip and dtdelete.exe files are (c:\scripts), the backup destination, and the duplex destination on server-2. We are retaining files less than 4 days old and the value one (1) tells the routine to zip the file created.

Next we schedule the transaction log file backups:

```
exec DBBackup_sp 'log', 'mydb', 'c:\scripts', 4, 'c:\scripts', 0,  
'd:\dbbackup\mydb\','\\server2\standbydest\mydb\', 'support@chriskempster.com', 'N'
```

The script above and its parameters tells the backup where our gzip and dtdelete.exe files are (c:\scripts), the backup destination, the duplex destination on server-2. We are retaining files less than 4 days old and the value one (1) tells the routine to zip the file created. The value zero (0) represents the email, when zero the DBA is only notified on backup failure, not success.

The DBA should actively monitor the backups for a good two or three days, ensuring full and log backups are copied successfully, the backups can be manually restored on server-2, and the deletion of files older than N days is working fine.

Server 2 (destination)

As mentioned in *server 1 (source)* setup, the DBA has already created the duplex directory e:\dbbackup\mydb\ and configured a share on the \dbbackup directory called *standbydest* using NT security. For server-2, we schedule three jobs that execute stored procedure routines to initialize, continue and monitor log-shipping.

Initialize

The main stored procedure is *log shipping initialize*. We supply the routine a number of parameters, being the name of the database to be restored, the location of the backups (remember—files were copied from server-1), the standby redo file, the pre and post-fix file extensions so the routine can build a list of files from disk to restore from, and finally, the *MOVE* command for each database filegroup.

Here is an example:

```
exec usp_LogShipping_Init
'mydb'
,'e:\dbbackup\mydb\' 
,'e:\dbbackup\mydb_standby.rdo' 
,'mydb_'
,'.bak*' 
,'_full.bak' 
,'_dif.bak' 
,'_trn.bak' 
,''
MOVE ''MYDB_SYSTEM'' TO ''c:\dbdata\mydb\mydbstandby_system01.mdf'', 
MOVE ''MYDB_DATA'' TO ''c:\dbdata\mydb\mydbstandby_data01.mdf'', 
MOVE ''MYDB_INDEX'' TO ''c:\dbdata\mydb\mydbstandby_index01.mdf'', 
MOVE ''MYDB_AUDIT'' TO ''c:\dbdata\mydb\mydbstandby_audit01.mdf'', 
MOVE ''MYDB_LOG'' TO ''c:\dbdata\mydb\mydbstandby_log01.mdf''
'
,'c:\scripts'
```

The DBA may consider further customization of this script, namely the building of the *MOVE* statement by reading the backup file header. You can, but I work on the KISS principle and in this scenario we can't go wrong.

The initialize routine is NOT SCHEDULED and is only run manually if we need to force the re-initialization of log shipping from the last full backup.

The *master..LogShipping_Audit* is updated accordingly with the files applied by the routine or any failure/error information.



This routine will locate the last full backup and apply it, then the last differential (if any) and all subsequent transaction logs.

Continue

This is the crux of the log shipping routines and is scheduled to run every two hours from 7am to 10pm. The routine has identical parameters to that of the initialize procedure. When run, the routine will determine if Initialize must be called (missing standby database), or a new full backup file has been found and we need to start from scratch with the full and differentials. This routine is basically the driver for log shipping, in both its initializing and continuing to apply transaction logs as they arrive in the backup directory from server-1.

```
exec usp_LogShipping_Continue
'mydb'
,'e:\dbbackup\mydb\' 
,'e:\dbbackup\mydb_standby.rdo'
,'mydb_'
,'.bak*'
,'_full.bak'
,'_dif.bak'
,'_trn.bak'
,''
MOVE ''MYDB_SYSTEM'' TO ''c:\dbdata\mydb\mydbstandby_system01.mdf'',
MOVE ''MYDB_DATA'' TO ''c:\dbdata\mydb\mydbstandby_data01.mdf'',
MOVE ''MYDB_INDEX'' TO ''c:\dbdata\mydb\mydbstandby_index01.mdf'',
MOVE ''MYDB_AUDIT'' TO ''c:\dbdata\mydb\mydbstandby_audit01.mdf'',
MOVE ''MYDB_LOG'' TO ''c:\dbdata\mydb\mydbstandby_log01.mdf''

'
,'c:\scripts'
```

Monitor

Using a simple-cdo custom DLL, this scheduled job running at the same schedule as *continue log shipping* calls this:

```
exec usp_LogShipping_Monitor 'support@chriskempster.com', 'LOGSHIP', 15
```

The DBA is emailed error rows from the *master..LogShipping_Audit* table. Here is an example of its contents:

4451	SUCCESS	16/11/2003 8:00:4	mydb	LOG	mydb_20031116_0 RESTORE LOG mydb FROM DISK= 'c:\dbbackup\mydb'
4450	SUCCESS	16/11/2003 8:00:4	mydb	LOG	mydb_20031116_0 RESTORE LOG mydb FROM DISK= 'c:\dbbackup\mydb'
4449	SUCCESS5	16/11/2003 6:21:1	mydb	LOG	mydb_20031116_0 RESTORE LOG mydb FROM DISK= 'c:\dbbackup\mydb'
4448	SUCCE55	16/11/2003 6:21:1	mydb	LOG	mydb_20031116_0 RESTORE LOG mydb FROM DISK= 'c:\dbbackup\mydb'
4447	SUCCE55	16/11/2003 6:21:0	mydb	LOG	mydb_20031116_0 RESTORE LOG mydb FROM DISK= 'c:\dbbackup\mydb'
4446	SUCCE55	16/11/2003 6:21:0	mydb	LOG	mydb_20031116_0 RESTORE LOG mydb FROM DISK= 'c:\dbbackup\mydb'
4445	SUCCESS	16/11/2003 6:14:0	mydb	LOG	mydb_20031116_0 RESTORE LOG mydb FROM DISK= 'c:\dbbackup\mydb'
4444	SUCCE55	16/11/2003 6:13:5	mydb	LOG	mydb_20031116_0 RESTORE LOG mydb FROM DISK= 'c:\dbbackup\mydb'
4443	SUCCESS	16/11/2003 6:13:5	mydb	LOG	mydb_20031115_2 RESTORE LOG mydb FROM DISK= 'c:\dbbackup\mydb'
4442	SUCCESS5	16/11/2003 6:13:4	mydb	LOG	mydb_20031115_2 RESTORE LOG mydb FROM DISK= 'c:\dbbackup\mydb'
4441	SUCCE55	16/11/2003 6:13:3	mydb	FULL	mydb_20031115_2 RESTORE DATABASE mydb FROM DISK= 'c:\dbbacku
4440	SUCCE55	16/11/2003 6:00:0	mydb	REINIT	<NULL> Found another full backup, reinitializing log shipping

Log Shipping Example 2 – Finalizing Recovery/Failover

The DBA will attempt the following in order, I say “attempt” as the first steps may fail and must be carefully monitored.

1. Re-evaluate the need to cutover and double check that the situation is such that cut over is required
2. Attempt to run a backup on server-1 using your scheduled DBBackup_sp command or via Query Analyzer
3. Verify backup and file copy, manually copy if required
4. Run usp_LogShipping_Continue (or its scheduled job) on server-2
5. Disable above job
6. Manually run `exec..usp_LogShipping_Finish 'db-name-here'`

Concluding thoughts

The method presented is simple, easy to implement and does not rely on linked servers or numerous system table lookups. One of the big disadvantages with log shipping is more to do with the recovery process (albeit short—consider this when testing), that being *"user sessions must be killed before attempting to recover the database"*. If users need to be kicked from the standby database to apply further logs, then its tough setting a specific recover time if the database is also used for corporate reporting.

INDEX

A

accountability, 23
active/passive mode, 134
authority, 23
Autonomic computing, 39
availability measures, 21, 22

B

Backup schedule, 15
Business Continuity, 6
business priority, 14

C

change control form, 34
change management system, 31
checkpoint, 92
CoBIT, 17
communications manager, 12
Crisis and Disaster Contact Details, 13
crisis manager, 12

D

database administration roles, 25
database diagram
 transfer, 104
DATABASEPROPERTYEX, 81
dbcc inputbuffer, 107
DBCC INPUTBUFFER, 103
dbcc loginfo, 73
dbcc sqlperf, 86
DBCC TRACEON, 93
DBCC TRACESTATUS, 93
Deadlocking, 97
Disaster recovery, 7
disaster recovery planning, 6
DR documentation, 11

E

ECID, 99
emergency response team, 24
error log, 95

F

Full Text Indexing
 cluster, 139

H

HCL, 117
High Availability, 118
hot fixes, 39

I

isql, 81
ITIL, 17

L

License Manager, 82
licensing mode, 83
log writer, 69
LSN, 70

M

Master Recovery Plan, 15
meta data functions, 77
MOF, 19
MRAC, 40

O

OBJECTPROPERTY, 84
orphaned session, 103
orphaned sessions, 102
osql, 81

P

parallel testing, 14
production server, 37
Profiler, 97

Q

Quality Online Service, 117

R

RAID, 115
reconfigure, 84
recovery manager, 12
Recovery strategy, 14
responsibility, 23
restoration history, 89
rollback and redo, 69

S

SAN, 116
SERVERPROPERTY, 82
service level
 metrics, 21
 simulation testing, 14
sp_change_users_login, 101
sp_configure, 84
sp_cycle_errorlog, 97
sp_enumerrorlogs, 96
sp_MSforeachtable, 84
sp_procoption, 86
sp_spaceused, 85
SQL Server Named instance
 cluster, 141
SQL Server Wizard, 143

T

test server, 36
time difference, 149
trace flags, 91
Tracing
 black box, 86
transaction manager, 74
transfer logins, 105

U

User
 database ownership, 103
 orphaned, 101, 102

V

VeriTest, 117
virtual IP, 137
VSS, 32, 42
 .Net, 63
 branching, 50

W

walk-through, 14
warm standby, 142
write log entry, 71
xp_fixeddrives, 86
xp_readerrorlog, 96