Pro PerformancePoint Server 2007: Building Business Intelligence Solutions

Copyright © 2008 by Philo Janus

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

ISBN-13 (pbk): 978-1-59059-961-7

ISBN-10 (pbk): 1-59059-961-6

ISBN-13 (electronic): 978-1-4302-0588-3

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Lead Editor: Jeffrey Pepper
Technical Reviewers: Dana Hoffman, Phillip Taylor
Editorial Board: Clay Andres, Steve Anglin, Ewan Buckingham, Tony Campbell, Gary Cornell, Jonathan Gennick, Matthew Moodie, Joseph Ottinger, Jeffrey Pepper, Frank Pohlmann, Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh
Project Manager: Richard Dal Porto
Copy Editors: Damon Larson, Heather Lang
Associate Production Director: Kari Brooks-Copony
Production Editor: Katie Stence
Compositor and Artist: Van Winkle Design Group
Proofreader: April Eddy
Indexer: Broccoli Information Management
Cover Designer: Kurt Krames
Manufacturing Director: Tom Debolski

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail orders-ny@springer-sbm.com, or visit http://www.springeronline.com.

For information on translations, please contact Apress directly at 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705. Phone 510-549-5930, fax 510-549-5939, e-mail info@apress.com, or visit http://www.apress.com.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales—eBook Licensing web page at http://www.apress.com/info/bulksales.

The information in this book is distributed on an "as is" basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is available to readers at http://www.apress.com. You may need to answer questions pertaining to this book in order to successfully download the code.

CHAPTER 1

Business Intelligence

Before we can talk about implementing business intelligence solutions, it's important to have an understanding from the business perspective on what our users are trying to accomplish. A repeating theme throughout this book will be that business intelligence is about solving business problems. So it's important that the business intelligence architect, DBA, developer, and so on be in tune with the problems involved on the business side as much as the technical side.

What Is Business Intelligence?

The IT industry has spent the last two decades getting data off of desktops and out of filing cabinets, and into relational databases. We've been very successful, and most business processes today are run from electronic data stores.

Unfortunately, as data has been moved by various initiatives and different groups into products by various vendors and integrators, we've ended up with huge collections of transactional silos. The data serves those who use the system—the warehouse can generate pick lists with bin numbers from orders, the financial group can generate invoices and checks, HR can manage employee records, and so on. But what about managers and executives who need an "all-up" perspective on their organization? They need to see current staffing levels and how they may compare to shipping times, order error rates, and stock levels, and how those numbers relate to truck loading rates and fuel usage. They will ask questions about how numbers interrelate, and will also want to perform analysis on relationships among data that may not be intuitive or obvious (data mining).

The problem domain can be summed up very simply: "I have several piles of data, and I want to get some value out of them."

BUSINESS INTELLIGENCE VS. PERFORMANCE MANAGEMENT

A lot of words have been written about the difference between business intelligence (BI) and performance management (PM), regarding where and how they overlap. One general theory is that BI is about delivering information while PM is about acting on that information.

My personal take on this: who cares? I feel that it's an esoteric exercise that doesn't really deliver any value to the people who need to run their companies. Whether you call a dashboard BI, BPM, or a dashboard, the important point is that it delivers the information necessary for people to monitor their business and make decisions on how to run it.

As I said, this is just my opinion.

Scorecards vs. Dashboards

The root of a BI solution (and often the impetus for a BI initiative) is the scorecard. There is often some confusion about when to use a scorecard and when to use a dashboard. Again, a lot of opinions exist about which is which, so let me share mine.

A *scorecard* (Figure 1-1) is a small, direct application that tracks a collection of *key performance indicators* (KPIs), and shows current actual values and target values, and a score for the KPI. KPIs may then be aggregated into *objectives* and/or *perspectives* with scores rolled up in either an average, a weighted average, or a bubble-up exception (showing the worst child score for a parent). Scorecards are strategic—they show long-term values, goals, and trends. Data in a scorecard should not be the type of data you would want to see in real time, but rather data that you monitor on a weekly or even monthly basis.

	Q1-06		Q2-06		Q3-06		Q4-06	
	Actual	Target	Actual	Target	Actual	Target	Actual	Target
⊟ **Financial Objectives**								
⊟ **Revenue**								
Sales Amt	$25,663,364		$34,366,793		$48,122,573		$72,000,830	
Units	14,005		18,752		24,057		37,164	
⊟ **Margins**								
Gross Margin %	19.89%		19.84%		16.52%		17.37%	
Gross Profit %	24.83%		24.76%		19.78%		21.02%	
⊟ **Costs**								
Cost	$20,558,319		$27,546,799		$40,174,195		$59,493,287	

Figure 1-1. *A scorecard*

A *dashboard* (Figure 1-2), on the other hand, is more tactical. This is where you'll see your near-real-time data. You'll want charts and graphs that show data changing over hours, and how the data interrelates. A scorecard may be part of a larger dashboard as a means of giving an overarching perspective to the more tactical data displayed in the dashboard.

A good analogy is that a dashboard in a car shows real-time data: oil pressure, speed, RPMs, and so on; while a GPS display and maintenance record are similar to a scorecard—showing where you've been and the long-term trends of your performance.

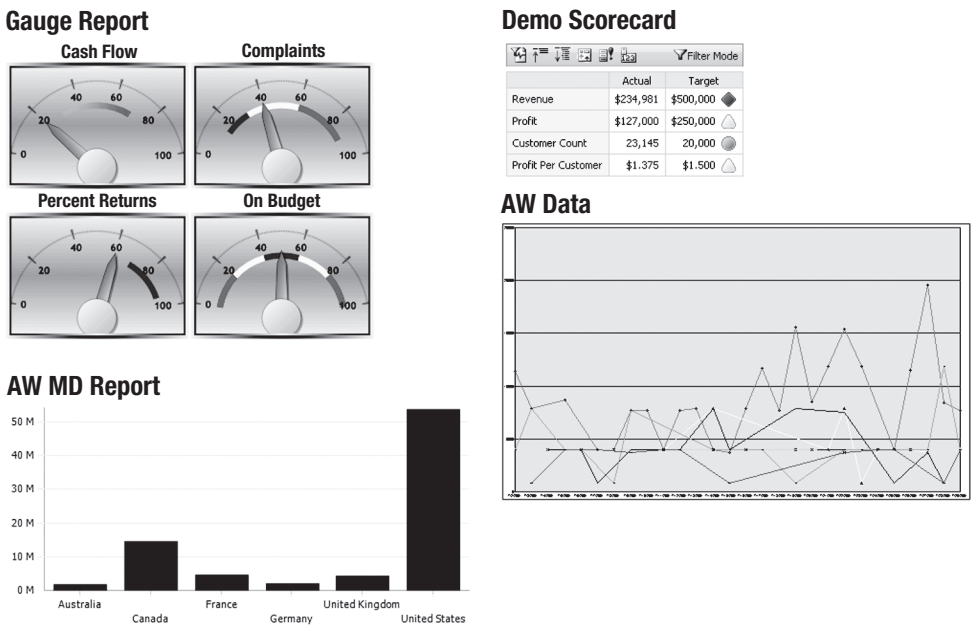


Figure 1-2. *A dashboard*

THE BALANCED SCORECARD

You will also hear about a special case of scorecard called a *balanced scorecard*. The balanced scorecard was created by Robert Kaplan and David Norton in 1992. Their goal was to pull the focus of management off the balance sheet (profit/loss) and pay attention to those factors outside finance that are indicative of the health of the company. To do this, they created what they called *perspectives* to aggregate KPIs and/or *objectives* (collections of KPIs).

Following are the four perspectives in a balanced scorecard:

Financial: Standard profit-and-loss type data

Customer: Measures indicative of customer satisfaction

Internal business processes: The health of the company's processes

Learning and growth: Primarily focused on employee quality and satisfaction

The goal of a balanced scorecard is to identify factors that are critical to the success of a business (and that will affect profit and loss down the line) *before* they become critical problems. For example, excessive employee attrition and turnover will eventually show up in decreased customer satisfaction, increased loss rates, and ultimately lower profits. Instead of waiting for it to become such a problem that it shows up on the balance sheet, by measuring attrition directly, management will get an advanced "heads up" when it becomes a problem.

Note that from a technical point of view, a balanced scorecard doesn't have any special requirements—it is simply a special case of scorecard. PerformancePoint Server allows you to build balanced or "unbalanced" scorecards.

For more information about balanced scorecards, check out the Balanced Scorecard Institute at `www.balancedscorecard.org/`.

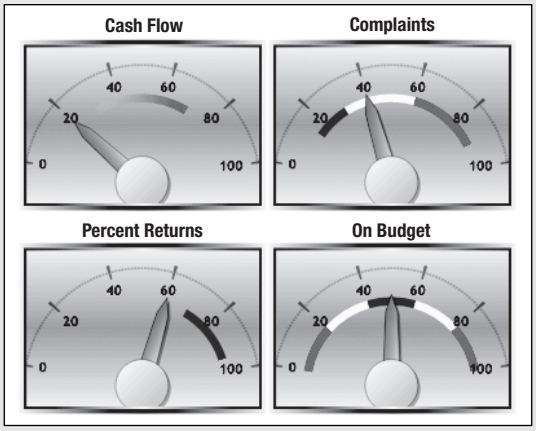
Wayne Eckerson presents a straightforward chart comparing scorecards and dashboards in *Performance Dashboards* (Wiley, 2005) (shown in Table 1-1).

Table 1-1. *Comparing a Dashboard to a Scorecard*

	Dashboard	Scorecard
Purpose	Measures performance	Charts progress
Users	Supervisors, specialists	Executives, managers, staff
Updates	Right-time feeds	Periodic snapshots
Data	Events	Summaries
Display	Visual graphs, raw data	Visual graphs, text comments

GRAPHS AND GAUGES

Note the gauges in Figure 1-2. How helpful are they? Now look at the gauges in the following illustration:



Note the labels—they may work well as reminders, but they are not very descriptive as to what the gauge is measuring, or how. If the gauges were self-describing (as we usually like graphical indicators to be), the labels would be incidental to what the gauges meant. Cash Flow is 20 and out of the red, but what does that mean? And what is the trend of the value? Is the Percent Returns gauge moving into the green or out of it?

If you'd like to really dig into maximizing the value from visual representations of data, I recommend *Information Dashboard Design*, by Stephen Few (O'Reilly, 2006), which walks through a number of dashboard designs by various vendors (sadly published before PerformancePoint was available), and discusses pros and cons of each design.

Once you have dashboard design down, dig into your indicators and charts with *Show Me the Numbers*, by Stephen Few (Analytics, 2004), which picks apart the various ways of representing data (including our favorite—the gauge).

Key Performance Indicators

We mentioned KPIs before, but what are they?

A KPI is where the business meets the data (see Figure 1-3). While a scorecard can have perspectives and objectives as business drivers, the actual metrics—the KPIs—are going to be data driven. The underlying principle originating here is, "You cannot manage what you cannot measure." So, while we may want happy customers and content employees and satisfied shareholders, those subjective concepts won't help us run our business.

Figure 1-3. *The center of a scorecard—the KPI*

So we must identify the data-driven "things" that will help us guide our business decisions. A standard mnemonic that is used to evaluate KPIs is *SMART*. Spelled out, indicators must be

- Specific
- Measurable
- Achievable
- Realistic
- Timely

Some examples of good KPIs include customer attrition, cost of acquiring a customer, employee retention, percent of late deliveries, average wait times, and revenue growth. These may seem intuitive, but you should recognize the various pitfalls associated with each.

"Customer attrition" seems like a great metric; however, if you are a retail store, how do you define a lost customer? Just because a customer hasn't visited the store in a month doesn't necessarily mean they've decided to never visit again. (If you're a tire store, you may only see customers once a year.) In addition, if you're a brick-and-mortar store, how do you even track customers (this explains affinity cards, doesn't it?)?

"Employee retention," by comparison, is pretty straightforward—you want to just look at employee turnover. However, the danger here is assuming one metric can serve the whole company. You have to be cautious to set baselines—it may turn out that while turnover in accounting is very low, the shipping dock is always churning employees. Before you try to hold the shipping dock to the standards set by the accounting department, do some research—it may turn out that shipping departments always have high turnover rates; it's the nature of the work. In that case, you can work on getting turnover *lower*, but you don't want to set the accounting department's numbers as a goal if it's unrealistic.

"Percent of late deliveries" is pretty much a line drive down the middle. All you have to be sure of is that you have the data to measure what a late delivery is.

In addition, since we're discussing wiring scorecards to data, keep in mind that a KPI must have the data to drive it. (You could factor this under "measurable" and "achievable.") An interesting aspect of this approach is how it can help keep KPIs honest—when an indicator is suggested, identify where the data is going to come from. If that data doesn't currently exist, you have to ask a series of questions:

- Why isn't the data currently being captured?
- Should it be captured?
- If not—if this data isn't worth building a business system for—should we be driving our business from it?

In other words, the need to have the data acts as something of a sanity check on the KPI itself.

As you start to look at KPIs, you may be quickly overwhelmed, especially if the organization does not currently have a scorecard or has never evaluated performance metrics.

KPIs and Business Process

David Parmenter, in his book *Key Performance Indicators: Developing, Implementing, and Using Winning KPIs* (Wiley, 2007), recommends a 12-step process that covers major success factors, such as stakeholder buy-in, organic growth, and iteration instead of "get it right the first time." His 12 steps for identifying and implementing KPIs are as follows:

1. Senior management team commitment
2. Establishing a "winning KPI" project team
3. Establishing a "just do it" culture and process
4. Setting up a holistic KPI development strategy
5. Marketing a KPI system to all employees
6. Identifying organization-wide critical success factors
7. Recording performance measures in a database
8. Selecting team-level performance measures
9. Selecting organizational winning KPIs
10. Developing the reporting frameworks at all levels
11. Facilitating the use of winning KPIs
12. Refining KPIs to maintain their relevance

The reason I list these is to drive home the point that KPIs, objectives, scorecards, and dashboards constitute a *business problem*. They will require significant effort by business stakeholders to get right, and they will require maintenance in the long term to continually reevaluate the indicators and ensure they are guiding the business appropriately. I highly recommend Parmenter's book as a good foundation of how to build a solid collection of KPIs.

The Law of Unintended Consequences

One final warning regarding KPIs is to be wary of creating unexpected behaviors. Since you can't always anticipate how people will react to metrics, this again points toward the issue that you cannot create KPIs and walk away—reevaluation of the intent and effects of KPIs must be part of the scorecard business process.

Here are a couple examples of unintended consequences:

> A company has a metric of "number of cases held over 20 days." The net result of this metric is that when your case is 18 days old, you'll see a flurry of activity, but when you hit the 3-week mark, it will suddenly go dead. Why? Because there is no metric to differentiate between a case that's 21 days old and one that's 90 days old. Once you're past the magic 20-day mark, there is no incentive to work on your case.

> One computer manufacturer implemented a metric on its support line counting "number of calls lasting more than 10 minutes." Their cost of support skyrocketed. When they dug into the background, they found that their support technicians would work hard to help customers for 9 minutes. As the clock entered that ninth minute, they would simply offer to ship the customer a new system to get them off the phone.

A classic example of unintended consequences is counting lines of code. Many development managers come to the conclusion that a good metric for developers is counting the lines of code they write every week. In the initial part of a development project, this may even render what appears to be good performance data.

However, there are a number of factors to consider that pretty much invalidate the use of "lines of code" as a metric:

- A lot of development is about solving a problem, so a developer may go a whole day and write four lines of code, but those four lines may be a very tight loop that fixes a performance bug.
- Other optimizations may involve deleting large chunks of code and replacing them with a few lines, for a net negative.
- A lot of development now is template-based—if a developer spends a day just setting up form templates where a tool generates 5,000 lines of code, does that count?
- While I would be loathe to suggest that developers often game the system, there are a lot of ways to write code such that what should be one line of code comes out as ten. Is that a desired outcome?

So, the unintended consequence of measuring "lines of code" as a developer metric is that you're rewarding developers that just stamp out template code or find ways to game the system, while you're penalizing the superstars who have a negative metric.

Another example of a unintended code-related consequence is in counting bugs. If "bugs reported" is used as a metric, with lower numbers being better, what you end up with are fights between the developers and testers over every bug reported as to whether it's really a bug. Developers should not have a vested interest in hiding bugs.

A better metric might be "function points delivered" or accuracy of project estimates (too many days over *or* under yielding a bad metric). There are a number of essays and commentaries about using metrics on software development projects. Here are some examples:

- "In pursuit of code quality: Monitoring cyclomatic complexity," by Andrew Glover (`www.ibm.com/developerworks/java/library/j-cq03316/index.html`)
- "Lines of code," from the c2 wiki (`http://c2.com/cgi/wiki?LinesOfCode`)
- "Hitting the high notes," by Joel Spolsky (`www.joelonsoftware.com/articles/HighNotes.html`)

To sum up, determining the KPIs for your organization is a nontrivial problem. If there are no KPIs currently, then there's a lot of work to be done on the business level. Even if your organization already has a scorecard and KPIs, but it's driven manually, you will find that many things will have to shift as you try to move the scorecard to a data-driven environment (e.g., the first time a reported green KPI goes red when the real data is hooked up).

Do *not* quote the implementation time or development time as a timeframe for scorecard implementation. Be sure that the proper business process analysis and implementation is being considered or else you'll be a software project manager being held up by a business process you have no control over.

Strategy Maps

Strategy maps were created by Robert Kaplan and David Norton (yes, the balanced scorecard guys). Kaplan and Norton, while working on balanced scorecard implementations, noticed that successful business implementations were the result of *focus* and *alignment*.

We're all familiar with the concept of a mission statement—most businesses have some form of mission or vision. They are traditionally the butt of many jokes, as they are often perceived as fluffy or obvious. What is often missing is the linkage between a company's mission statement and what the company actually does. It's easy to have a mission statement of "Provide valuable services to our customers," but this begs the question of how?

Strategy maps are designed to link a company's high-level goals (*perspectives*, in balanced scorecard parlance) to the KPIs that measure how the company is performing on the measures that drive the business. A strategy map shows how KPIs relate to objectives and then to perspectives (Figure 1-4).

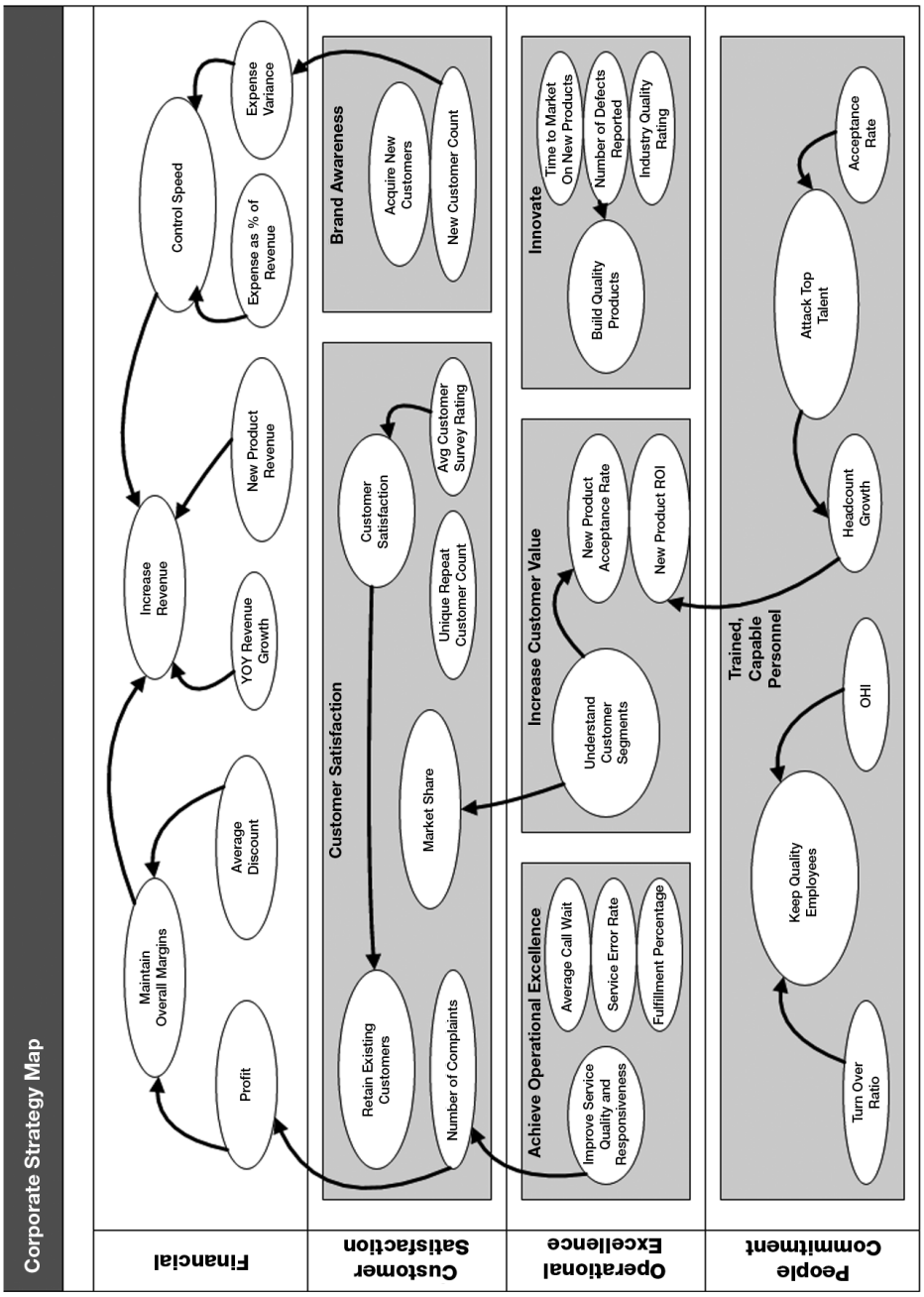


Figure 1-4. *A strategy map*

Very often, a company will build a strategy map before attempting a scorecard, as a way to formalize the company's strategy and how these factors interrelate. Strategy maps are generally drawn in a simple drawing package (like Microsoft Visio), but are increasingly included in a scorecard implementation to actually show the relationships between KPIs and the corporate strategy.

Data Silos

The root of the problem, again, is that we have a lot of data, and it's all in silos. The way we get data out of silos is generally through transactional reporting—we'll get reports from this data source or that data source, and on rare occasions we may get a report that pulls data from two or more data sources. But all the information lives in isolation—it's rare that we can actually view information from disparate back-end systems in a way that reflects how we do business (see Figure 1-5).

Very often, reports are actually named for the system they come from—for example, "The Warehouse Picklist Report" or "The FIPS Report" (where FIPS is a system written by some guy 10 years ago). The systems are driving reporting, not our business. What we get instead are piles of reports that nobody ever reads.

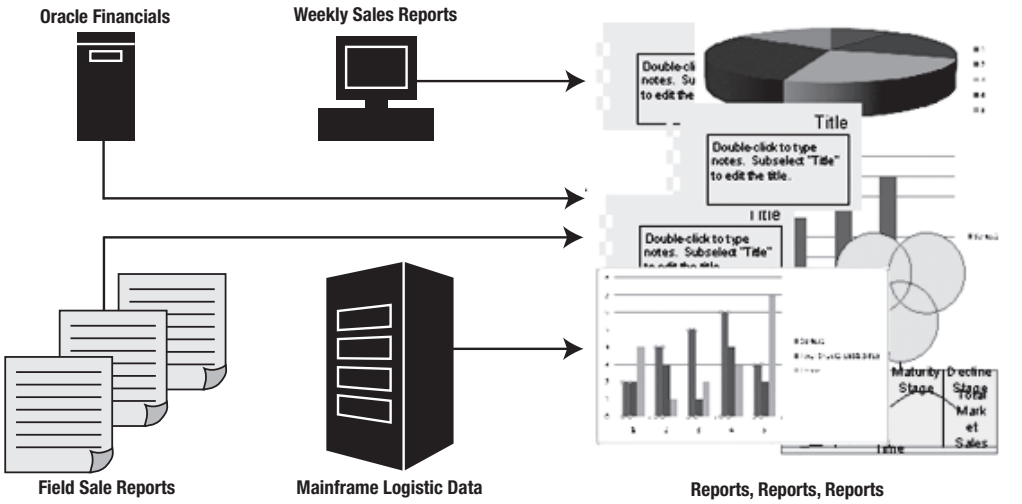


Figure 1-5. *Information silos result in piles of reports nobody reads.*

These reports generally become references—decision-makers dig into them when they are looking for a specific answer. It's much rarer that reports are referred to on a regular basis to indicate any kind of status. And of course, when there's an excess of reports, then they simply accumulate in a virtual bin somewhere.

Data Marts

What we want is a way to pull this stuff into one place, in a way that makes sense. The industry solution to this problem is the *data mart*. You may also hear references to OLAP (online analytical processing, as coined by E.F. Codd & Associates in 1994) or dimensional data storage. The fundamental idea is that we are starting to break down the system barrier and look at our information in ways that make sense to our business.

■ **Tip** I prefer to use the term *data mart* or *cube*. I've found that if you try to talk about a "data warehouse" anywhere near an executive or a consultant, they will immediately launch a 3-year project to build the "Corporate Data Warehouse," and you're stuck at square one. The other possible sticking point if you use the "DW" phrase is that your initiative will be shut down with the statement "We already have a data warehouse project in progress." Data warehouses have their place, but there is no reason an agile data mart project cannot happen in tandem.

Instead of getting reports named after the system that produced them, we want information structured similarly to the way we do business. We want to be able to break down warehouse delivery by customer type, order volume by warehouse location (or vice versa), and processing backlog by customer order amount. Where the data comes from or how it's stored in each system isn't a concern for a business user; all they want to do is use the data in the ways they run their business.

OLAP cubes seem complicated, but once you understand the basics, they are pretty straightforward. The fundamental concept is similar to pivot tables—we want to aggregate relational data by the dimensions we are interested in. For example, we may have a list of purchases made in a store. While that list of purchases is good for stock checking or auditing, simply having a list of 1,000 (or more) individual purchases doesn't tell us a lot—what did people buy a lot of? Is there a time of day that's busiest? Are people buying a lot of goods at once or are most purchases in the express lane (12 items or less)?

Using a pivot table in Excel, we can group purchases by item or by checkout aisle. But grouping by time is problematic—every purchase timestamp is to the second, so unless two people are buying at the exact same instant, the rollup will simply be the same list again. We could create a calculated column to pull out the hour of purchase and aggregate by that, but doing that every day would be painful, and it quickly bogs down if we start to talk about multiple stores and hundreds of thousands of purchases.

So the idea is to get away from the reams of relational records (as shown in Figure 1-6) and give our users the ability to work with data in a format that makes sense to them (as shown in the cube browser in Figure 1-7).

ProductKey	OrderDateKey	DueDateKey	ShipDateKey	ResellerKey	CustomerKey	EmployeeKey	PromotionKey	CurrencyKey	SalesTerritory	SalesO
372	762	774	769	97	(null)	282	1	100	4	SO517
287	762	774	769	97	(null)	282	1	100	4	SO517
380	762	774	769	97	(null)	282	1	100	4	SO517
390	762	774	769	97	(null)	282	1	100	4	SO517
525	762	774	769	99	(null)	285	1	100	5	SO517
599	762	774	769	99	(null)	285	1	100	5	SO517
549	762	774	769	99	(null)	285	1	100	5	SO517
361	762	774	769	100	(null)	291	1	19	6	SO518
598	762	774	769	100	(null)	291	1	19	6	SO518
516	762	774	769	100	(null)	291	1	19	6	SO518
513	762	774	769	100	(null)	291	1	19	6	SO518
359	762	774	769	100	(null)	291	1	19	6	SO518
551	762	774	769	100	(null)	291	1	19	6	SO518
353	762	774	769	100	(null)	291	1	19	6	SO518
596	762	774	769	100	(null)	291	1	19	6	SO518
363	762	774	769	100	(null)	291	1	19	6	SO518
402	762	774	769	100	(null)	291	1	19	6	SO518
290	762	774	769	100	(null)	291	1	19	6	SO518
511	762	774	769	100	(null)	291	1	19	6	SO518
592	762	774	769	100	(null)	291	1	19	6	SO518
591	762	774	769	100	(null)	291	1	19	6	SO518
517	762	774	769	100	(null)	291	1	19	6	SO518
532	762	774	769	100	(null)	291	1	19	6	SO518

Figure 1-6. *Rows and rows and rows of relational records*

	Calendar Year ▾				
	⊞ CY 2001	⊞ CY 2002	⊞ CY 2003	⊞ CY 2004	Grand Total
Group ▾	Internet Sales Amount	Internet Sales Amount	Internet Sales Amount	Internet Sales Amount	Internet Sales Amount
⊞ Europe	$709,947.20	$1,627,759.71	$3,382,979.27	$3,209,356.08	$8,930,042.26
⊞ North America	$1,247,379.26	$2,748,298.93	$3,374,296.82	$3,997,659.37	$11,367,634.37
⊞ Pacific	$1,309,047.20	$2,154,284.88	$3,033,784.21	$2,563,884.29	$9,061,000.58
Grand Total	$3,266,373.66	$6,530,343.53	$9,791,060.30	$9,770,899.74	$29,358,677.22

Figure 1-7. *Browsing a cube in Analysis Services*

Note how the table in Figure 1-7 uses terms that a business user would be comfortable with. While the table in Figure 1-6 has ProductKey and SalesTerritory by number (meaning that we have to find the tables they map to), the table in Analysis Services has sales territory groups, fiscal years, and Internet Sales Amount columns (properly formatted). Later, we'll see that once a cube is built, creating information like this is incredibly easy.

■ **Note** We have discussed the problems with time fields in Excel, and yet Analysis Services has rolled our records by time up to the fiscal year. We can drill down by quarter, month, day, and so on. Analysis Services understands time implicitly, but this can be tricky to accomplish properly. We'll cover time dimensions in Chapter 5.

Why Do I Care?

So far, we've talked about managing business through metrics, how to best determine those metrics, how to aggregate metrics and align KPIs to corporate strategy, and the problems with trying to connect business drivers to the data we want to drive those indicators with. Just as a strategy map aligns KPIs and their data to business strategy, we need to align the products we're going to discuss with a BI solution.

This next section will give you some guidance before we start to dive into the really techie stuff behind the scenes.

The Microsoft Business Intelligence Stack

The BI solution from Microsoft is as shown in Figure 1-8.

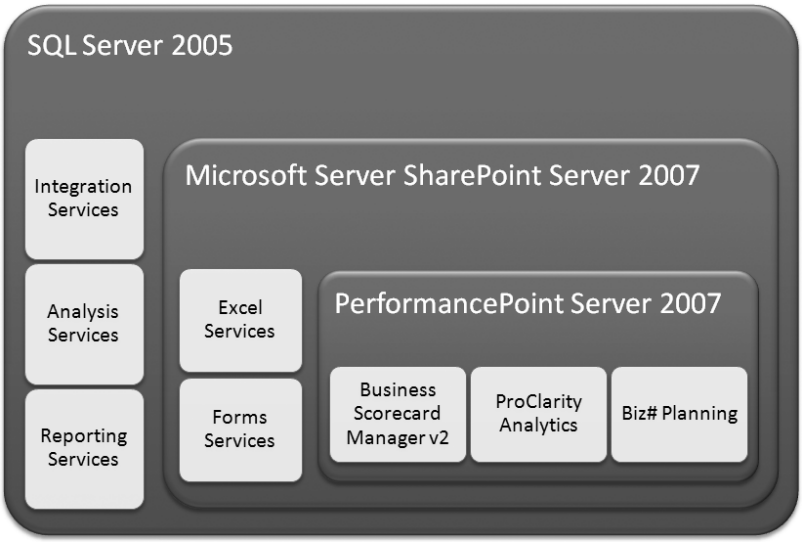


Figure 1-8. *The products in a Microsoft BI solution*

My goal is to cover these products and give some baseline understanding of how they fit into a BI solution. The most compelling part of Microsoft's BI platform is SQL Server and the capabilities that it gives you in a SQL Server license.

SQL Server 2005

The foundation of our solution is SQL Server 2005. In addition to the relational database capabilities we're all familiar with, SQL Server offers powerful BI services included in the licensing for a server. While not installed by default, it is very straightforward to install and configure these additional services.

SQL Server Integration Services (also referred to as SSIS): This is an enterprise-class ETL (extract, transform, load) tool that enables you to extract data from one location, process it, and push it to another location. While neither endpoint is required to be SQL Server (e.g., you could use Integration Services to move data from Excel spreadsheets to Oracle), in this book we'll be using Integration Services to pull data from data sources and load it into a staging database in SQL Server. Integration Services is the first step in getting our data out of the silos we have and together in some form of homogenous data store.

SQL Server Analysis Services (also referred to as SSAS): This is where we build our data marts. Analysis Services allows us to map various data sources together for use as the facts and dimensions in our cubes. In addition to building and managing OLAP cubes, Analysis Services offers the ability to have calculated measures—for example, calculating the gross profit by subtracting a cost field from a sales price field. We can also run averages, standard deviations, averages of child values, and so on.

SQL Server Reporting Services (also referred to as SSRS): This is a powerful web-based reporting server. The most important thing to understand about Reporting Services is that the data represented in a report does *not* have to be in SQL Server—you can create reports on Sybase data, for example, and simply use SQL Server as a report server.

Microsoft Office 2007

On top of SQL Server (most notably on top of Analysis Services), we see Excel 2007, Excel Services, and SharePoint. These make up the basic presentation layer for our BI. With these tools, we can create basic charts and graphs, and with Excel, of course, our users can do some basic analysis.

Excel 2007, with SQL Server 2005 Service Pack 2 (SP2), gives us a rich data mining capability. Once we have aggregated our data into a cube, we can start digging into the aggregated information and looking for various patterns. This is referred to as *data mining*, and while the engine is in SQL Server Analysis Services, I find the Excel plug-ins to be very compelling as a way for a business user to really leverage the data mining capabilities in a much easier-to-use fashion.

The next layer, however, is where we really want to focus our attention for display and visualization.

Microsoft Business Intelligence

This is where Microsoft's BI capabilities really show their mettle:

Business Scorecard Manager (BSM) 2005: This grew out of a solution accelerator designed to show off Analysis Services. It became so popular that Microsoft made it into a supported product. BSM's main purpose is to display standard red-yellow-green scorecards. The solution accelerator only displayed data from Analysis Services, and BSM works best on Analysis Services, but it is capable of using any ODBC data source for both actual values and targets.

ProClarity Analytics Server 6.3: This was acquired by Microsoft in June 2006. The ProClarity corporation carried a suite of products including a dashboard server, desktop analytics, and other servers. However, the primary product Microsoft has concentrated on, and is moving forward with, is Analytics Server. ProClarity Analytics Server provides rich, powerful visualizations on Analysis Services data. It also empowers end users to build their own analytic charts and dig into the dimensional data, and publish those charts and graphs for others to view.

Capabilities from both BSM and ProClarity Analytics Server are being rolled into PerformancePoint Server 2007. In addition to monitoring (scorecards) and analytics (detailed charts and graphs), PerformancePoint adds a robust planning and modeling system to enable end users to build dimensional models on their own, publish those models, make assignments, collect input, and roll data together. It's a powerful tool for "what-if" and forecasting scenarios. Code-named Biz# (pronounced *biz sharp*) in development, this will round out the capabilities of PerformancePoint.

By combining all the capabilities of SQL Server, SharePoint, and PerformancePoint, we can build a truly robust BI system—aggregating data, storing it, creating a strong analytical data mart based on the data gathered, and presenting that data in a myriad of ways through reporting or embedded in SharePoint sites where users can create their own dashboards (see Figure 1-8, shown previously).

A Successful Business Intelligence Engagement

Incidentally, one very powerful aspect of Microsoft's BI solution is that it's "a la carte"—you can implement it piecemeal and build up to a fully automated BI solution. Many products (and many consulting engagements) take an all-or-nothing approach—they need to gather all the requirements, define all the objectives and KPIs, identify all the data sources, and design all the dashboards, and then they will go to work on the whole thing for a number of months before finally delivering a "completed" product. Of course, unveiling a "completed" product for the first time will indicate just how incomplete it is from the customer's perspective.

PerformancePoint scorecarding gives you the capability to build a scorecard manually—so you can work with the customer to define a rough outline of their scorecard in a way that brings it to life. Once the scorecard is built and published on a web site (even if only with hand-coded numbers), then stakeholders can see the scorecard and understand it as presented, encouraging feedback and review of the metrics and objectives.

At the same time, the IT department can identify the business systems that will provide the data for the scorecard. As the scorecard survives its initial review, IT can start evaluating the actual sources of the data and identify "quick wins" for pulling data into data marts to serve the scorecard. What we want to do is build foundations in technology, knowledge, and process that we can build our BI solutions on in an iterative way.

Once we've identified business data we want to put in a cube, we can build some Integration Services ETL packages to pull the data from its source location, scrub it, and load it into a staging database. We can then build a data mart using the data from the staging database—build some cubes and perhaps some KPIs inside Analysis Services.

Finally, we wire the scorecard KPIs we've addressed to the cube, as well as provide additional analytic reports and drill-down capabilities.

This is step one. From here it's a degree of "lather, rinse, repeat," except that we want each iteration to be a growth activity—identify what worked and what didn't work so that the next round of implementation goes more smoothly. In the meantime, we can expect our business stakeholders to be evaluating the KPIs we've wired up and giving us feedback on how they need to be adjusted.

The overriding goal here is to make a slow, gradual transition from existing processes (often a pile of spreadsheets and a scorecard in PowerPoint) to a formal, fully automated BI process that simply renders scorecards and dashboards with near-real-time data. We want to do this in a measured way such that we minimize work that's going to be thrown away, and we also want to build the processes necessary to *maintain* the products that are built.

Conclusion

From here, we'll be focusing much more on technology (a sigh of relief from the geeks in the house). We'll dive into the various platforms and products I've discussed, and I'll give you a functional overview of how they work, how to get started with them, and how to best take advantage of them from a BI point of view.

These chapters are only meant to be introductory chapters—helping you to understand the capabilities and give you a running start. There is a wealth of information available to dig deeper on each technology, both online and in book stores.

With that, let's take a look at the technologies involved.