

Application Platforms and Business Strategy: Making the Connection

David Chappell

January 2008

Sponsored by Microsoft Corporation

Contents

[The Link Between Strategy and Platforms 3](#_Toc190240693)

[How Application Platforms Support Business Strategy 3](#_Toc190240694)

[A Closer Look at Application Platforms 5](#_Toc190240695)

[Supporting Business Strategy: An Example Using the Microsoft Application Platform 6](#_Toc190240696)

[An Overview of the Microsoft Application Platform 6](#_Toc190240697)

[A Rental Car Company Scenario 8](#_Toc190240698)

[Using Application Services 9](#_Toc190240699)

[Using Integration Services 10](#_Toc190240700)

[Using Data Services 11](#_Toc190240701)

[Conclusion 13](#_Toc190240702)

[About the Author 13](#_Toc190240703)

# The Link Between Strategy and Platforms

Every organization pursues business strategies that differentiate it from its competitors. Application platforms, the software that supports custom applications, play an essential role in providing this differentiation. The applications they make possible are one of the most important ways—perhaps *the* most important way—that information technology can support unique business strategies.

This paper starts with a description of how business strategies depend on application platforms. It then provides a brief overview of Microsoft’s application platform, walking through its main components. Finally, the paper gives an example of how a fictitious rental car company’s decision to embark on a new business strategy relies on the capabilities of its application platform. The goal is to make clear the role an application platform plays in implementing an organization’s business strategies.

## How Application Platforms Support Business Strategy

The essence of business strategy is being different. A firm might do different things from its competitors, for example, such as offering different products or addressing different markets. Alternatively, it might do the same things in different ways, such as providing a lower-cost service. In either case, what marks an action as strategic is its ability to differentiate an organization from its competitors, letting it provide unique value to customers.

Competitive advantage flows from successful differentiation. Yet keeping good ideas secret is hard, and so remaining different can be challenging. Figure 1 shows what frequently happens.

Figure : Successful innovations by a strategic leader eventually become obligations for other firms in the industry.

The first firm in an industry to implement a successful innovation gains a significant competitive advantage. The second firm to implement this strategy also derives some advantage from it. By the time the third firm in an industry follows suit, the new approach is usually well on its way to becoming a best practice. Anybody who doesn’t implement it is likely to be at a competitive disadvantage. What began as an innovation has become an obligation.

There’s no shortage of examples that illustrate this. The first airline to add self-service check-in kiosks, for instance, gained a significant competitive advantage. The second airline to add self-service check-in kiosks also got some advantage. Yet today, kiosks are a cost of doing business for every airline—not having them would be a competitive disadvantage. Similarly, when FedEx let its customers track their packages directly via the Web, this innovation was rightly hailed as a powerful competitive advantage. Today, we all expect this from shipping companies—the innovation is now obligatory.

Pretty much every modern business strategy depends in some way on IT. Self-service check-in kiosks and Web-based package tracking are good examples of this reality. Yet note what happens: Over time, every successful strategy becomes part of the IT infrastructure. It’s fair to say that virtually all of IT was once an innovation that conferred competitive advantage on its first adopters. Eventually, however, everything becomes a utilitarian IT function that must be supported effectively.

Given this, it’s possible to divide IT investments into two broad categories:

1. *Strategic* *IT*, spending on new capabilities that directly support new business strategies;
2. *Utility* *IT*, all other IT spending. To a large degree, the technologies in this category represent the accretion of an organization’s strategic innovations over many years.

Figure 2 illustrates how these two categories fit with the competitive advantage curve shown earlier.

Figure : Strategic IT supports innovations in business strategy that provide competitive advantage.

As the figure shows, strategic IT supports the innovations that provide competitive advantage to an organization. But what does strategic IT really consist of? The answer in most cases is simple: custom applications. While packaged software is fundamental to many businesses, getting strategic differentiation from a generic package is hard. Because of this, innovations such as airline check-in kiosks, Web-based package tracking, and many others depend on custom applications.

Custom applications are built on application platforms. This implies that the application platform an organization chooses can have a powerful impact on its ability to create competitive advantage. As Figure 2 shows, there’s a window of differentiation in which innovations can provide a competitive advantage, and so the platform must allow creating custom applications quickly. Yet successful innovations eventually move into the utility category, and so these applications must also be manageable over the long term.

To summarize, the connection between business strategy and application platforms is this:

1. Business strategy means *being different* from the competition.
2. Being different relies on *strategic IT* investments to support that differentiation.
3. Strategic IT investments are most often *custom applications.*
4. Custom applications depend on an *application platform.*

## A Closer Look at Application Platforms

There’s no one right way to think about application platforms. Different people—and different vendors—view them differently. Figure 3 shows one simple model for describing this important group of technologies.

Figure : An application platform’s services can be grouped into four areas.

The components needed to support applications can be organized into four distinct areas. Those areas are:

1. *Application services*, providing a foundation for building applications. This includes the ability to create and execute code, interact with users via the Web or a local interface, communicate directly with other software using standard protocols, and more.
2. *Integration services*, dealing with the challenges of connecting to diverse applications. As Figure 3 suggests, applications commonly need to interact with other applications built on all kinds of platforms. To allow this, integration services can provide adapters for diverse communication styles, tools for mapping between different data representations, the ability to create and run process logic, and more.
3. *Data services*, letting applications and their users work effectively with the information they depend on. The fundamental technology here is typically a relational database engine. These services can also include support for creating historical repositories of data called *data warehouses*, along with the ability to mine this data for patterns and to examine it using the techniques of online analytical processing (OLAP).
4. *Tools*, making it possible for people to work effectively with all three kinds of services. Tools are essential for things such as building and testing applications, creating integration connections, and working with data warehouses.

An application platform relies on an operating system, as Figure 3 shows, and other components. Various management tools might be used to deploy and monitor applications, for example, while other software provides security services, support for identity, and more. All of this is important, yet it’s fair to say that the application platform itself has the most direct connection with business strategy.

# Supporting Business Strategy: An Example Using the Microsoft Application Platform

Getting a clear sense of how an application platform can support business strategy requires seeing concrete examples of how the two fit together. To do this, this section first gives a brief description of the Microsoft application platform. It then walks through an example that shows how the various parts of this platform can be used to support a firm’s strategic objectives.

## An Overview of the Microsoft Application Platform

As Figure 4 shows, the Microsoft application platform includes implementations of the four components just described. Application services are provided by the .NET Framework, integration services by BizTalk Server, and data services by SQL Server. Tools for all three run in Visual Studio.

Figure : The Microsoft application platform provides application services, integration services, data services, and tools.

Some of the most important of the application services provided by the .NET Framework 3.5 today are:

1. ASP.NET, which lets developers build standards-based Web applications.
2. Windows Communication Foundation (WCF), allowing applications to communicate directly with other software via Web services (using either SOAP or a RESTful approach) and other protocols.
3. Windows Presentation Foundation (WPF), helping developers create user interfaces for Windows applications that include two- and three-dimensional graphics, video, and more.
4. Silverlight, helping developers create user interfaces for Web applications that include two-dimensional graphics, video, and other options. Both WPF and Silverlight rely on XAML, an XML-based language for describing user interfaces.

BizTalk Server 2006 R2, the product’s current release, implements a broad range of integration services. Among them are:

1. Adapters for communicating with diverse applications using various protocols.
2. The ability to run workflow logic, referred to as *orchestrations* in BizTalk Server.
3. Support for electronic data interchange (EDI), a widely used standard for business-to-business communication.
4. Support for applications that use radio frequency identifier (RFID) tags. These tags transmit identifying information to RFID readers, and they can be attached to pallets in a warehouse, horses in a stable, or other things.

For data services, the Microsoft application platform relies on SQL Server 2008. Among the technologies included with this product are:

1. SQL Server Database Engine, which allows the creation and use of relational databases.
2. SQL Server Integration Services, letting organizations create data warehouses from data in SQL Server and other database management systems.
3. SQL Server Analysis Services, which lets users create and use OLAP cubes, perform data mining, and more.

Using any of an application platform’s services—application, integration, or data—requires tools. Visual Studio 2008, the current release of Microsoft’s flagship tool, provides an integrated development environment for creating .NET Framework applications. Different versions of this tool are available that target specific functions in the software development process, such as defining an application’s architecture, writing its code, and testing. To help organizations come to grips with the broader concerns of application lifecycle management, Microsoft offers Visual Studio Team System. This set of products supports using Visual Studio with various development processes, providing a common code repository, standard reports, and other services. The goal is to make it easier for software development teams to use repeatable and manageable processes, and thus to reduce the risk of those projects.

Visual Studio 2008 also hosts tools for other platform components. For example, BizTalk Server’s Orchestration Designer runs in Visual Studio, while the Business Intelligence Development Studio, another Visual Studio-based technology, provides tools for SQL Server Analysis Services. Rather than providing a different tool environment for each platform component, Microsoft has chosen to create a common foundation for all of them.

For the most part, Visual Studio targets a technical audience. Yet in a modern software development process, less technically focused people can also play an important role. Think about the challenge of creating an effective user interface, for example—how many developers really have the right skills to do this? A better approach is to let designers and developers work together, each using an appropriate tool. To allow this, Microsoft provides Expression Blend, a tool to help designers create user interfaces for WPF and Silverlight applications. Because both of these technologies rely on XAML, developers using Visual Studio and designers using Expression Blend can work together, exchanging interface implementations using this common format.

## A Rental Car Company Scenario

The clearest way to understand the connection between business strategies and application platforms is to walk through an example. Think, for instance, about a (fictitious) rental car company that decides to reinvent itself. Rather than differentiating solely on price or service, this company’s executives have noticed that while travelers can choose from a myriad of boutique hotels and specialized airlines, all rental car companies are pretty much the same. With this in mind, they have chosen to create a boutique rental car firm, one that clearly stands out as the desirable choice for style-conscious consumers, including the underserved market of younger renters. The goal is to attract a loyal customer base willing to pay a premium for the brand.

Achieving these business goals requires changing several aspects of the organization. Rather than offering traditional rental cars, for example, the firm will need to provide a more exciting mix of choices. Implementing this business strategy also depends heavily on the application platform, as the rest of this section illustrates.

### Using Application Services

A large percentage of car rentals are booked on the Web. To appeal to younger customers and to present a more stylish image, this rental car company offers a premier user experience for Web bookings. Rather than the usual simple forms, the customer sees a beautiful and well-designed interactive interface. Among other things, she can call up dramatic video clips showing the cars on offer, giving her a better sense for what those cars look like and perhaps even enticing her into a higher-priced upgrade.

Good design can be an effective and inexpensive way to change how customers perceive a company. Yet providing this kind of user interface requires an application platform that supports it. Figure 5 shows how the Microsoft application platform makes this possible.

Figure : The .NET Framework allows creating better browser interfaces with Silverlight, exposing Web services with WCF, and more.

As the figure shows, the firm’s reservation application is built using the .NET Framework’s ASP.NET. To provide a better experience for Web customers, including the ability to display videos of available cars, the application’s browser client uses Silverlight. Relying on this technology lets the developers and designers who create the reservation application do things that aren’t possible with just standard HTML Web pages.

While the firm’s new strategy requires changing the customer experience, it also means changing how its employees interact with the applications they use. For example, this firm maintains a call center where employees use the reservation application. A browser interface might be fine for customers, but a reservation agent who uses an application all day would likely prefer a more efficient Windows client. Yet many aspects of the user interface are the same across the browser and Windows clients; employees should have access to everything the Web customers see and more. To make this easier to do, the call center clients are created using WPF. Because Silverlight and WPF both rely on XAML to describe user interfaces, significant amounts of the user interface implementation can be reused across the two clients. As mentioned earlier, this common technical underpinning helps developers and designers work together, each using a tool targeted for their role.

WCF, another part of the .NET Framework, plays an important part here as well. For example, the call center clients use WCF to communicate with the reservation application. Also, because embracing the Web 2.0 ethos of mash-ups is an important part of the image this company wishes to create, the application uses WCF to expose both SOAP-based and RESTful services over the Internet. This lets their reservations service become part of other applications, such as corporate travel systems. The firm might even create a Facebook application that uses these Web services to let its customers show their friends lists of the cool cars they’ve rented from this company.

### Using Integration Services

Using current application technologies to support new business strategies is important. Yet working well with what’s already installed is also important. However progressive this rental car company strives to be, connecting its reservation system to existing software and to its partners must be a fundamental part of its business strategy. With the Microsoft application platform, this problem is addressed by BizTalk Server 2006 R2, as Figure 6 shows.

Figure : BizTalk Server provides several kinds of integration, such as EDI connections and an SAP adapter, and it also includes support for applications that work with RFIDs.

Big firms frequently require their partners to interact with them using EDI, and so the rental car firm is obligated to communicate using this aging but widely used standard. Because it sits between a partner system, such as an airline reservation application, and this firm’s reservation application, BizTalk Server can communicate with both. A request received from the airline application, for example, will arrive as an EDI message. BizTalk Server then transforms this message into, say, an XML request, perhaps relying on WCF to communicate with the rental car reservation application. Both applications see this interaction in terms that they understand, and BizTalk Server takes care of translating between their different perspectives.

The reservation application must also interact with this firm’s SAP ERP application. To allow this, BizTalk Server includes an SAP adapter that’s capable of interacting with this popular package in its own terms. As with EDI, BizTalk Server handles the translation, shielding the reservation application from the other software it works with. And to coordinate all of these interactions, a developer might create a BizTalk orchestration that implements any necessary business logic.

A big part of this firm’s business strategy is to provide cars that are more enticing than those of its competitors. These cars are likely to be more expensive than average, and so using them as efficiently as possible is a fundamental part of making this strategy successful. To help do this, the firm has attached RFID tags to all of its vehicles. By installing RFID readers at entrances, exits, and other places, it can track the location of every vehicle within its facilities. The car tracking application it uses to do this is built on the RFID server that’s included with BizTalk Server 2006 R2.

Connecting different systems doesn’t always need an integration technology such as BizTalk Server, however. Sometimes a simpler approach is just fine. In this example, for instance, the rental car firm has an existing credit check application that it relies on to help determine which customers are allowed to rent cars. Appealing to younger drivers is a risky business—that’s why it’s an underserved market—and so checking their credit effectively is critical. (In fact, this firm might even have embarked on this strategy because it was very good at doing this.) The credit check application is written in Java, however, and it runs on a Java Enterprise Edition (EE) application server. Hooking it to the rental car reservation system might require nothing more than a direct SOAP connection, as Figure 6 shows. Because both the .NET Framework and the leading Java EE application servers support SOAP, doing this is straightforward.

### Using Data Services

Most enterprise applications consume and create data. Both kinds of activities are important, and so the data services an application platform offers are critical. In the Microsoft platform, those services are provided by SQL Server 2008.

In this example, the rental car reservation application maintains information about this firm’s customers and their rentals in a SQL Server database. The car tracking application also generates plenty of RFID-based data that’s stored in another SQL Server database. The credit check application, however, stores its data using an Oracle database. Figure 7 illustrates the situation.

Figure : SQL Server 2008 lets applications work with data, allows creating data warehouses, and provides business intelligence services.

All of these applications use their data for everyday operations, such as creating rental reservations. The accumulation of this operational data also has value, and so it’s periodically loaded into a data warehouse. As Figure 7 shows, this operation relies on SQL Server Integration Services. While it’s a standard part of SQL Server 2008, this technology can work with data stored in other database products, such as Oracle and IBM’s DB2.

Capturing long-term historical data is common, and so this firm’s competitors probably have data warehouses, too. Competitive advantage comes not from collecting this information, but rather from using it well. For example, successfully executing the strategy of going after younger customers requires keeping a close eye on trends. Knowing what car models are most appealing at different times and in different geographies depends on intelligently analyzing this historical data. Similarly, maximizing the utilization of the firm’s rental fleet depends on the RFID data that tracks the location of each car. Providing this kind of business intelligence (BI) is a fundamental responsibility of the application platform.

In the Microsoft application platform, SQL Server Analysis Services is the foundation for understanding historical data. Various clients can be used with Analysis Services for data mining and OLAP, including Excel, PerformancePoint Server, and third-party tools. And rather than restricting this analysis to a few specialists, the general nature of these tools—and their price—lets this organization deploy BI technology broadly. A rental car manager in the smallest airport location can have access to the intelligence he needs to do his job better. Given the risks inherent in a business strategy of appealing to style-conscious younger renters, using business intelligence effectively can be a big help in hitting the firm’s profit targets.

# Conclusion

The ability to support new business strategies is at the heart of business/IT alignment. IT decision makers commonly worry about operational risks such as “What if my data center goes down?” Many of them worry less about strategic risks like “What if we can’t support the next new business strategy?” Yet not being able to support a business strategy can have a much bigger long-term impact than a one-time systems failure. Missing a chance to differentiate the business because an application platform can’t quickly support the CEO’s big new idea won’t help any IT manager’s career.

Similarly, when a competitor rolls out a new strategy that lets it be first to exploit a window of differentiation, it can be important to follow suit as fast as possible. Doing this might require creating new applications, connecting existing applications, using data in new ways, and more. All of these depend on an effective application platform that can meet these goals.

Making the right choice in this area requires keeping several things in mind:

1. An application platform is more than just services for writing and running new code. It also includes integration services, since most new software must somehow connect to what already exists. Data services are an essential component as well, since creative use of an organization’s data can be fundamental to business strategy.
2. An application platform should support current technologies. Whether you’re the first innovator or a fast follower, you often need current technologies to succeed. In the rental car example, for instance, WCF’s ability to provide both SOAP-based and RESTful services allows the company to plug into both the traditional Web services world and more current Web 2.0 applications.
3. An application platform must support older technologies where appropriate, especially for integration. In the rental car scenario, for example, connecting to existing systems requires using the decades-old technology of EDI.
4. Since strategic IT projects eventually move into the utility category, an application platform should support both worlds well. A decision maker focused solely on the strategic aspects might repeatedly choose the latest hot development technology, leading to a hodge-podge of environments and high support costs. (It’s also hard to find and retain experienced developers for bleeding-edge technologies.) Similarly, an IT executive who cared only about keeping long-term support costs low would miss the business benefits that new technologies can bring. For its primary application platform, an organization should steer a middle ground between these two extremes.

Viewing an application platform as a foundation for business strategy isn’t a stretch; it’s just a statement of fact. Because of this, choosing a platform that can support an organization’s strategies well is a primary part of creating competitive advantage. Whether the choice is for a single application or for the whole company, there’s no more important IT decision.

# About the Author

David Chappell is Principal of Chappell & Associates (www.davidchappell.com) in San Francisco, California. Through his speaking, writing, and consulting, he helps information technology professionals around the world understand, use, and make better decisions about enterprise software.