The Architecture Journal

Learn the discipline, pursue the art, and contribute ideas at www.architecturejournal.net

input for better outcomes

Architecture in Turbulent Times

Microsoft

Contents

Dear Architect,

Let's start with the basics: The goal of our Architect role is to align IT with the course of business. So far, so good, eh? So, being that almost every business has been challenged as a consequence of the current economic slowdown, we—as architects—are challenged to make our architectures follow the pace of business that these days is particularly changeable and unstable.

We thought that we had a clear understanding of the rules corresponding to the latest scenario. Having made a plan to face it, and being in the middle of the execution, we're told that the organization must now implement further changes to keep business going. The results of the last quarter were known, and growth expectations were revised to lower—perhaps negative—numbers.

This issue of *The Architecture Journal* won't tell you when this meltdown will end or whether we'll have kept our jobs. It will, however, share architecture strategies, IT behaviors, and—for sure—success stories. I counted on the invaluable collaboration of Mike Walker as guest editor for this issue. Mike wrote the opening article, which details how architects can add value back to the business during climates as particular as the current one. He and the other authors are architects like you and me: They didn't live during the Great Depression and, even if they had, technology in general and IT architecture as a discipline were slightly different from what they are today. They were suddenly challenged, and they responded to those challenges. Now, they tell us how they are succeeding. I hope you'll find both inspiration and courage in these pages.

By the way, as announced in the last issue, we keep adding features to the *Journal* to take advantage of its now mainstream digital format. We've introduced a "Follow Up" section at the end of any article that has references for going further with implementation.

This issue, we're introducing the "guest columnist" concept. You'll find that an article comes with one, two, or even three guest columnists. We intend to widen coverage on a given topic by offering different perspectives and voices. We're not yet done with these new features. Also, for those of you who are interested in receiving a printed version, we're still working on a print on-demand model.

By chance, are you interested in writing an article for the *Journal*? We just finished a call for papers on "SOA at the end of the decade," and we thank all of you who participated in that call. You can learn about a future call for papers by subscribing to *The Architecture Journal* newsletter, which you can do by visiting *The Architecture Journal* Web site.

As always, you can let us know your opinions directly by e-mailing us.

Diego Dagum
Editor-in-Chief

Architecture in Turbulent Times

by Mike Walker

Summary

This article aims to show architects a way to understand key forces on the business of IT, what they can do to add value, and the key areas of focus and technologies that will help them deliver value back to the business.

Introduction

The current economic condition in which we reside (as of this writing) has given rise to particularly challenging times for information-technology (IT) professionals. Events in the financial sectors and other corresponding industry sectors have had a substantial impact on technology products and services. Although there is much debate by economists on the length, depth, and impact of the economic crisis, we can safely assume that it will not go away in the foreseeable future (see Forrester; October 15, 2008). Its duration is enough to affect business and IT priorities significantly for the next fiscal year or two.

Despite the economic slump, businesses that continue to invest and innovate will have significant competitive advantage both during and after it. Furthermore, organizations that look past the immediate challenges and toward the opportunities that the economic condition presents will have longevity and sustainability well after it. A great example of this is growth by acquisitions. Making key technology assessments and acquisitions can change the position of a company in the market, reduce risk, and increase stockholder value.

IT will be at the center of this innovation; it is a key enabler for companies. IT decision makers such as architects will find themselves in high demand, and their expertise is one that will be used to its full extent. An architect poses the technical know-how to make complex and holistic decisions that affect millions of dollars for an organization.

This article will focus on how the economy affects both architecture in the enterprise and the architect who is a technical decision maker in many companies. By exploring the economic condition along with forces that are imposed on companies, key technology focus areas will emerge. Those technology areas will lead to specific technologies that will satisfy business demands.

Shifting Demands for Architects

With this new economic condition, companies will start to realign their IT decision makers in accordance with their priorities as a business. Creating alignment with the business and architects is imperative. The architect is at the center of most major IT decisions, by either making the decision or being an advisor to the decisions in question. This realignment will naturally shift the priorities for architects. The exciting, new, strategic technology projects will fade away, and the more tactical projects will prevail.

This can be seen with the current trends and activities in the market, such as:

- **Doing more with less.** Large corporations have stated that their IT budgets will be cut by as much as 50 percent or more, but that they will continue to have the same service-level agreements (SLA) as they did in the past.
- **Trimming existing project costs.** In the same vein as "doing more with less," trimming project costs are immediate and tactical activities that will determine the course of specific IT and business directions. Specifically, big service-oriented architecture (SOA) projects will become more pragmatic and actionable, instead of ambitious and multiyear initiatives.
- **Mergers and acquisitions (M&A).** As market conditions become more climactic, industries will consolidate. IT systems have proliferated through every aspect of the business. Typically, companies have various implementations of similar process and technologies; nevertheless, they are different. More than ever, architects are needed who can understand and provide insight into technologies in the M&A decision-making process.
- **Revitalizing the skills base.** People are an asset; they represent the whole of a company. IT decision makers have an opportunity to revitalize their leadership and technical acumen in these tough times to make better decisions, grow the business, and take advantage of the economic crisis as a time of innovation.
- **New approaches to outsourcing.** While leveraging tried-and-true methods, there are new enabling technologies that will provide value to companies. These include Platform as a Service (PaaS) and Software as a Service (SaaS) vendors. As an example, Microsoft provides a wide variety of services—from its Azure cloud-services infrastructure, with hosted data storage, workflow, and application development, to SaaS solutions such as Dynamics CRM Live.

For architects, business as usual will change; they will have to leverage their diverse skill set to address the demands of the industry and their company. The skills that architects will leverage are the following:

- **Motivation and inspiration**—Sometimes, complex IT decisions that have elusive return on value require additional persuasion for the enterprise to buy in. Architects will leverage this skill to rally the enterprise for the right causes.
- **Negotiation**—There will be times at the decision-making table when an architect must negotiate to get things accomplished. Most architects are individual contributors and do not have organizational power, so an ability to negotiate is key.
- **Critical thinking**—Being able to think quickly and on one's toes often is required for architects, especially in times when making the right decision is critical for one's business.

- **Problem solving**—Architects will need to find new and innovative approaches to solving traditional problems. With various forces on their business, the problems of the past have changed and will require quantitative and qualitative ways to evaluate and solve problems.
- **Big thinking**—Even more so now, IT personnel will need to look more holistically, given pressures on cost control and return-on-investment (ROI) needs. Avoiding tunnel vision and being able to look at a problem from multiple angles to test one's own rationale represent a skill set that architects possess and will be able to demonstrate to the rest of the IT organization.
- **Business savvy**—Having a deep understanding of the business is key to making the right IT decisions; to do so, architects will use their business savvy to communicate and learn from their business leaders and subject-matter experts (SMEs). Understanding the industry in which one works is essential; it helps architects understand how the technology decisions that are made affect the business objectives and how it does within the industry.
- **Process orientation**—Thinking in terms of process is essential for an architect; this is the native language for the business. Thinking in terms of a business leader is essential, as is building repeatable and reusable processes both as artifacts from the work that they do and how they work themselves.

The skills that were mentioned—such as process orientation, business savvy/technical acumen, and critical thinking—will aid architects in better understanding how their specific company operates. Each company is different in how it interrupts the industry and correspondingly reacts to that understanding.

To understand the priorities of an organization, we must take a step back to understand what influences the direction that our companies take. There are direct and indirect influences—or forces—that determine the course of a company. These forces can change as time progresses, so that predicting them can be very difficult. Business SMEs often understand these forces very well, as they keep their eyes and ears on them to qualify their decisions. Architects must do the same.

IT has become less of a function of strategic value to companies—no longer a necessary evil or cost center, but a real differentiating factor in the business of a company. Architects who have an understanding of forces will provide inherent alignment with the goals and objectives of a company.

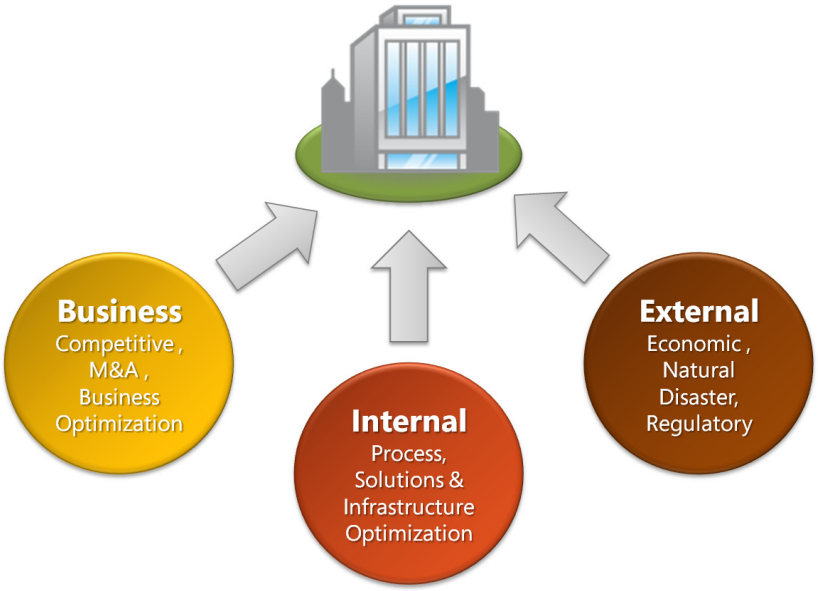
Forces can be grouped into three high-level groups. These groups include the following:

- **External**—Forces that are outside the organization forces and cannot be controlled
- **Business**—Purely business-related forces that can be derived from the inside or outside
- **Internal**—Forces that originate from the specific culture and operating model of a company

CIOs, COOs, and IT architects will take into consideration these forces, as they drive technology decisions, initiatives, projects, and purchasing. Understanding these forces will enable architects to snap to the business priorities and imperatives with more ease and less uncertainty.

Figure 1 shows more detail on the industry forces and their effect on companies.

Figure 1: Industry forces on enterprises



External forces are events or circumstances in the broader market that affect an organization. Unlike business forces, these often are uncontrollable external events that eventually will need to be addressed by the business.

The following are usually the drivers for business forces:

- **Faltering economy**—There is no doubt that the faltering economy has affected businesses dramatically. Gartner predicts that worldwide IT growth will drop significantly—from what was first forecast as 8.9 to 7.3 percent (2008 prediction) to the current forecast of 5.8 to 2.3 percent (see Gartner; December 31, 2008).
- **Regulatory compliance**—The need to fulfill compliance requirements has always been a requirement of companies. However, there is now increased pressure, fueled by the economic crisis. Governments will mandate a combination of green IT, security and privacy, auditing, and industry-specific regulations. Interestingly, 43 percent of technology leaders think that the 2008 election will affect IT decisions going further (see CDW IT Monitor; September 10, 2008).
- **Natural disasters**—Companies are increasingly dependent on IT-systems availability and information integrity. This dependency, combined with the sprawl of natural disasters, forces architects to evaluate how systems where built. There is and will continue to be a need for business-continuity planning, data-replication strategies, and disaster-recovery mechanisms.

Business forces are forces that relate directly to the business; they affect how the business operates and makes decisions. The following list shows the major business forces that will affect us in these times of uncertainty:

- **Increased competition**—Businesses are increasingly more competitive, as the market tightens up. Increased competition forces architects to focus on mission-critical solutions for the business that they support. Furthermore, Gartner furthers this by stating that enterprises do not want to disclose that they are aggressively cutting their IT budgets for competitive reasons (see Gartner; December 31, 2008).

- **Increased M&A**—In an economy such as the current one (as of this writing), it is common practice that there be significant acquisitions. We have seen this in the financial sector, but it will have a ripple effect across industries. M&A activities will spur a wealth of IT activities. Architects will play a major part in technology-portfolio evaluations, trade-off analysis of systems, architecture-gap analysis, security analysis and evaluation, and integration analysis.
- **Business-optimization activities**—With increased pressure from competition and modest projections, the theme of most companies will be to do more with less. Architects will need to look at enabling technologies to measure, track, and analyze business activities.

Internal forces are the result of both external and business forces. These are the forces on which many architects act directly; they are much closer to the business of IT:

- **Process improvement**—Given some of the business-optimization needs of companies, process improvement and streamlining will occur. Architects should look at both:
  - Efficiencies in the systems that measure, track, and analyze processes.
  - Playing a part in optimizing the process itself to adapt to changing conditions. Forrester's Gene Leganza states, "Enterprise architects need to work across development, support, and operations teams to drive decisions" (see Forrester; October 28, 2008).
- **Infrastructure optimization**—Both infrastructure and application optimization are direct results of "doing more with less." As budgets are squeezed and server budgets dwindle, architects will need to evaluate infrastructure efficiencies. More often than not, there are opportunities to consolidate, virtualize, centralize, and repurpose infrastructure investments.
- **Application portfolio management**—Not only will the infrastructure need to be evaluated, but the applications will, too. Applications often can be elusive and not completely visible to architects or senior, technical decision makers. With pressure from both cost and regulatory perspectives, it is critical that applications be well understood and managed.

How Architects Can Add Value

As discussed earlier, the role of an architect is a unique one. Architects are some of the more unbiased technology decision makers, and they have a holistic view of solutions in an enterprise. In times of uncertainty, this can be a great asset to organizations. This skill set will be used now more than ever in a faltering economy.

We find that the cutting of budgets for IT does not correlate directly to IT not being important. It does show the need to streamline and improve IT. So, in essence, it means more IT activities. For architects, it means a realignment of architectural priorities.

Table 1: Priorities of IT

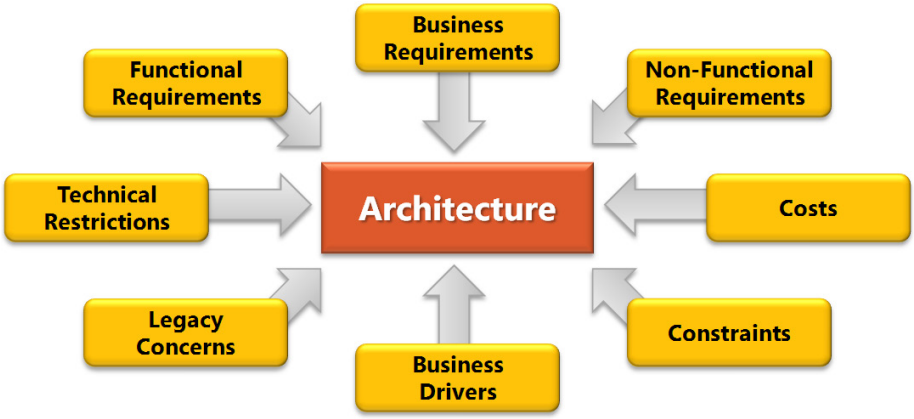
New priorities	Less-relevant priorities
Optimization of current portfolio—With market consolidation and realignment over the past couple of years, companies will try to understand their assets better and how to leverage them in a struggling economy.	**New projects**—There is a decreasing amount of new projects being initiated, because of the overwhelming amount of internal optimization that is needed.
Cost-reducing programs—Technical decision makers and architects are pressured to do more with less. Initiation of programs that look at cost reductions will be a major property for architects.	**Large-scale SOA projects**—With widespread failures of large SOA projects, architects are reconsidering their approach to SOA in their enterprise. While SOA will continue, it will be pragmatic and tactical.
Mergers and acquisitions (M&A)—For most companies, M&A is a fact of life; in recent years, however, there has been an accelerated number of acquisitions. A number of pre- and post-M&A projects will occur during this time.	**Web 2.0 projects**—Activity will happen here for niche areas; however, experiments on whether this technology paradigm is viable will be deemed non–mission-critical.
Compliance—Around the world, new regulations will emerge that will control all aspects of how IT operates. Examples of regulations that are likely to emerge include: • Privacy. • Fraud. • Payments. • Green manufacturing and IT. • Process management. • IT architecture and management.	**Server replacements**—Toward reducing costs, infrastructure architects will review depreciation cycles of their servers and desktops. Likely to happen is the consolidation of servers, and the lifetimes of servers and desktops will be extended—thus reducing the amount of projects for new hardware appropriations.
Value-added customer-facing projects—These further the reach of the company past traditional methods to add value to both the customer and the company. A key area of growth is the mobile-phone space. Mobile applications will continue to grow and integrate services to the mobile phone.	

Shifting of Priorities
Architectural priorities have changed, and the activities that an architect once made a high priority have now changed their course. What was once strategic to the organization is now labeled as risky or nice to have. Prudent low-risk initiatives are now in order. In the coming years, architects will engage in mission-critical, high–return-on-investment (ROI), low–total-cost-of-ownership (TCO) activities.

Table 1 shows examples of the IT priorities that are expected to change in the coming years.

From Table 1, it is clear that priorities have shifted from large-scale, sometimes risky IT investments to tactical activities and projects that have a near-term ROI. Dynamic, dexterous architects will have few or no problems adapting to this change.

Figure 2: Inputs to architecture decision making



Methods for Maximizing Architecture Decisions

In the previous sections, we discussed that we must change our mindsets and make our solutions much pragmatic. To do so, we must use pragmatic methods.

Pragmatic methods should both quantify and qualify our architecture decisions. The challenge is that there are not many methods to do this, as such. Traditional challenges were that architects quantified decisions in their own minds. The challenge here is that this is subject to bias and interpretation of the challenges; it does not provide a repeatable method that produces consistent results.

There is a method that can be used to produce this level of analysis in a repeatable fashion. The method is called the Architecture Qualification Method (AQM).

The Architecture Qualification Method (AQM)
The AQM is a pragmatic way of aligning with business architecture and, ultimately, fully qualifying our architecture decisions. This is performed by architects and can plug in to existing architecture frameworks (for example, TOGAF, FEAF, EACOE, and IASA), architecture standards, and proven practices. The AQM provides a set of tools that aid architects to achieve the following:

- Use of a method for capturing quality attributes for the enterprise
- Mechanisms for capturing definitions of quality attributes that *can* be tailored to your organization
- Intelligence that allows for the classification and variability of quality attributes
- Provision of a framework to rationalize attributes to the solution
- A way to govern how architectures are classified and rationalized throughout the organization

The AQM directly solves issues by using complexity and the understanding of architectures. Figure 2 shows an example of only a small subset of inputs to the architecture decision-making process. As you can see, there is input from all sides.

The AQM solves the following challenges:

- Provide a way to capture architecture analysis in a repeatable way.
- Qualify the quality attributes of the architecture.
- Reduce complexity and confusion in the build-out of solution architectures.
- Create a classification framework that allows for the orchestration of architecture standards and design patterns.
- Ease governance through the qualification of usage of standards and design patterns.
- Make architecture reuse proactively via reduction of complexity.

Architectures have multiple cross-cutting concerns that can be difficult to understand and quantify.

How Does the AQM Do This, and How Is It Different?
The AQM takes a structured approach to solving architectural problems. This is different from other solutions, because it provides a way for enterprises to define their own measurements, based on the business that they are in. See Figure 3, for example.

Figure 3 shows how the priorities of these two industries are different. This is furthered by the fact that the companies in these industries have slightly different ways of running their businesses, which can have a dramatic impact on how we create architectures.

Figure 3: Differences in industry priorities

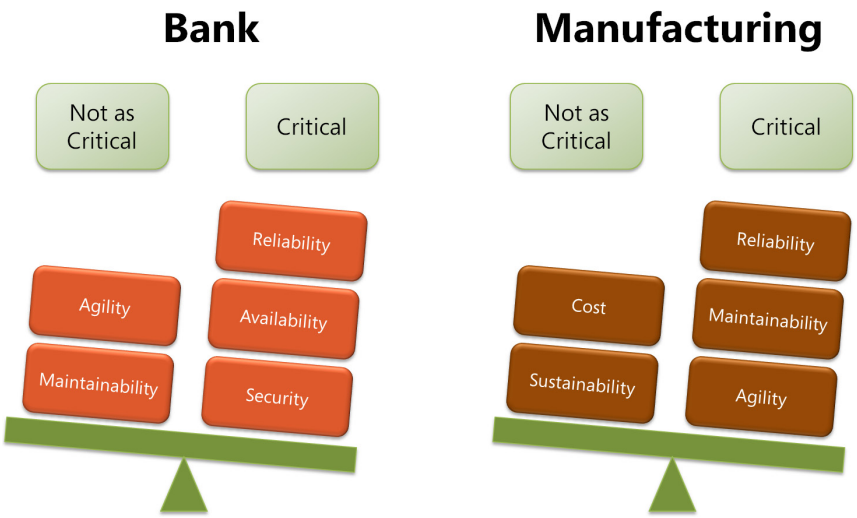
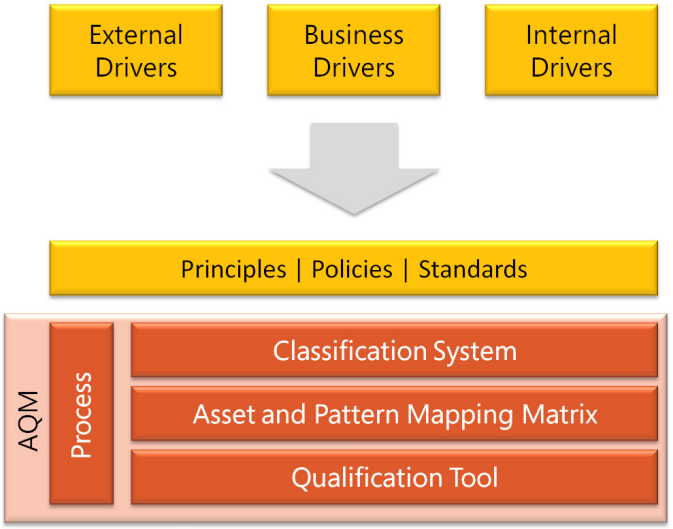


Figure 4: The AQM and business priorities



This is illustrated in Jeanne Ross's book *Enterprise Architecture as Strategy: Creating a Foundation for Business Execution* (Boston, MA: Harvard Business School Press, 2006), in which she compares the differences between two companies in the same industry (such as FedEx and UPS) and shows vast differences in the IT systems that support their businesses.

What we get when we look at all of this is what is shown in Figure 4.

Providing a way to have full traceability is a core tenet of the AQM. It is absolutely critical to be able not only to justify our decisions, but also to quantify and qualify them. The only way to do that is to align our decisions with business imperatives.

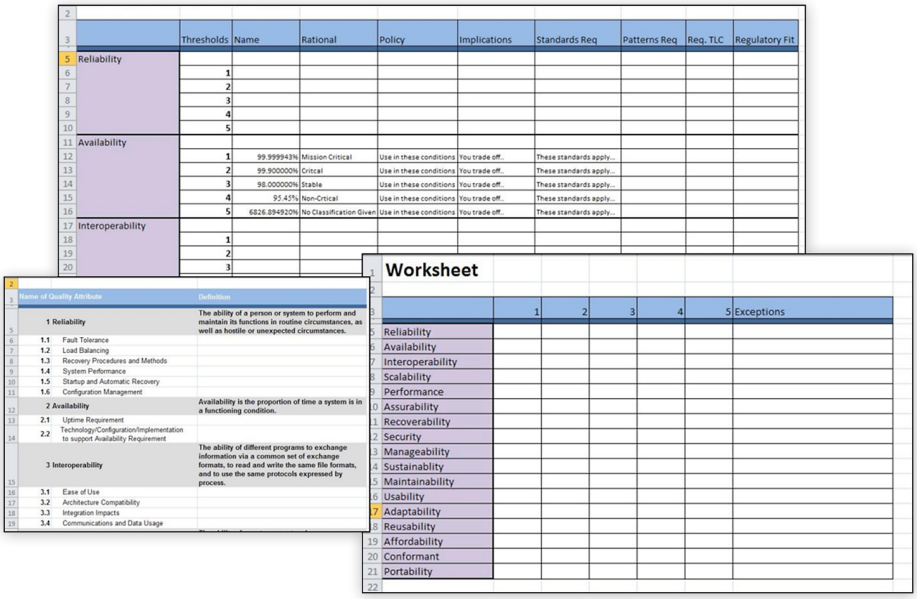
The AQM is used with the following components, as illustrated in Figure 4.

- **Process**—Specific to the AQM, this component provides the integration into existing processes and its own unique process for executing the AQM.
- **Classification system**—The classification system covers two aspects:
  - Definition of quality attributes into the terminology and the specific taxonomies that are well understood by your enterprise. (The unfortunate reality is that we all have various definitions of architecture terms.)
  - Creation of classifications, properties, and thresholds on quality attributes. This is one of the more important aspects of the AQM. This mechanism provides a way to fully qualify quality attributes through proper definition of what they mean to your enterprise. Through these classifications, properties, and thresholds, the AQM is able to both quantify and qualify architectures.
- **Asset and pattern mapping matrix**—As soon as a classification system has been created through the AQM, you can start to correlate it with real things—"real things" meaning your design patterns, standards, policies, and existing IT assets (system or application). By doing, so you can have a systematic way of prescribing what patterns to use for solutions—subsequently automating decision making, making it traceable, and easing governance.

  An example of this is a common scenario that I run into. Say that you organization has standards on a set of technologies for communications. Say also that those standards are FTP, SFTP, Web Services, CIFS/SMB, Connect Direct, and Microsoft SQL Server Integration Services (SSIS). Because all of these are in the standards list, developers or architects might not choose the optimal one for a variety of reasons that might include cost factors, complexity, or lack of understanding of the technology or personal biases.

  In the scenario, then, the architect chooses FTP because it is cheap and faster to implement. The problem is that the drivers for the solution call for the transactions to be highly secure and have a high level of resiliency. FTP is not the choice here. So, with the AQM, we can put qualifiers on these standards to show in what circumstances you should use each standard. This is applied to all of the architecture quality attributes (architecture "–ilities," such as scalability, security, reliability, interoperability, and so on).
- **Qualification tool**—The qualification tool comes into play when the architect wants to build out a solution. As mentioned in the preceding scenario, the tool provides a way to put all of the pieces together in an easy way. All of the complexity is removed from the architect, so that the architect can focus on creating a solution, instead of thumbing though thousands of standards pages to find (ultimately) something that might or might not be useful.

Figure 5: Aspects of the AQM



So, How Do I Use the AQM?

It is one thing to define a methodology, but it is an entirely different thing to make it actionable. The following list shows how the AQM is rationalized and immediately actionable and pragmatic:

- Office Excel–based templates that can be used by nearly every architect from Day 1
- Integration into an architecture meta-data repository
- Standard process definitions in Office SharePoint

Figure 5 shows screen captures of various aspects of the AQM. Data has not been populated in these, for very specific purposes. However, in the sample, data is populated in the classification matrix to show sample values. This is very simplistic and shows just what kind of data would be represented. In a real-world implementation, the repository would handle a great deal of this by joining multiple relationships with information and classifiers of a quality attribute. The spreadsheet is a viewing mechanism.

We took a very high-level look at the AQM and showed the purpose, methods, and tools that are associated with it. Through this method, you can expect a greater level of accuracy and quality of architectural decision making.

Four Key Architectural Imperatives

Figure 6 shows the imperatives that architects should keep in mind in these times of uncertainty.

These imperatives comprise four overarching imperatives:

- **Align**—Find direct links to business imperatives.
- **Optimize**—Do more with what you have.
- **Externalize**—Move IT assets outside of the IT operating environment, if they do not add value.
- **Consolidate**—Reduce unnecessary redundancies.

Align

Pressure from many forces on businesses will force IT alignment with business objectives. Now more than ever, companies are striving for this alignment. Architects will need to invest in better qualification of architecture decisions to ensure that value is added to the business. This qualification often is elusive and difficult to ascertain.

Understanding what areas to look at is critical. In the following list, you will find the major areas in which you can start to quantify architecture decisions. This is not a comprehensive list, but a starting point for architects to gather these key information points:

- **Key metrics**—All architects struggle with ways of quantifying their efforts, especially higher-level architects (such as enterprise architects). Instead of defining IT-specific metrics, architecture organizations will need to operate more like a business. By doing so, they will need not only to demonstrate their effectiveness, but also to quantify it.
- **Assessments**—Repeatable and consistent ways of evaluating solutions for their business fit is a must. These assessments should drive how decisions are made, and they illustrate how architectural trade-offs occurred. Assessments include architecture-viability assessments, architecture trade-off analysis, architecture decisions documents, and standards RFP assessments.
- **Requirements management**—Why should architects care about requirements? Simply put, it drives not only the functionality, but also the architecture. Capturing functional and nonfunctional requirements in reusable ways will help align architectures to the business. This is the fastest, easiest way to get such alignment.
- **Architecture management**—The next area of concern is to look at what has already been built and how it fits into the existing and new imperatives of the business. Architecture management links into standard processes, such as application portfolio management and PMO processes.

Figure 6: Key architectural imperatives

Optimize

As IT budgets shrink and the "big bang" projects dissolve, eyes will turn to optimization of existing solutions. Driving to get more value (from what the enterprise already has) will be a key imperative. Architects will be need to evaluate current solutions by determining how they are used, whether they are running optimally, whether they are redundant, and whether they fit the use that the business intended.

These questions are very difficult to answer. To answer them, a great deal of information is required that might or might not be captured. There is a strong dependency on process. If the company has lax processes around software development and architectural processes, the quality or the information itself might be absent.

Key activities that will aid architects in obtaining this information are:

- **Portfolio management.** Reviewing the IT portfolio of applications will allow architects to either inventory existing systems or review the systems that have been cataloged. When choosing to optimize the business, portfolio management is very important; it surfaces all of the key aspects of a solution and tells us information such as:
  - How the solution links back to the business and (sometimes) a business process or capability.
  - Scorecarding of the solution across all other solutions.
  - Overall cost of the solution.
  - Links to how the solution is supported.
- **Application life-cycle management (ALM).** Whereas portfolio management looked holistically across the enterprise, ALM is much more focused. ALM improvements can include process optimizations to streamline efforts, new tools to automate and accelerate application development, and key information-gathering points to support architecture efforts and quantify business value.
- **Revisiting architecture and development tooling.** Optimizing the tooling that is used in the actual development and architectural planning will be essential for companies. In times of slowed project work, it is optimal to retool.
- **Optimizing solutions.** Not only will there be process improvements, but those solutions that we evaluated and classified in portfolio management will need to be optimized in some way. This will open up opportunities for architects to be creative with solutions. With a wave of new innovations in social computing, context-aware architectures, cloud-based architectures, and SaaS

Figure 7: Evolution of externalized services

Outsourcing	ASP	MSP	Cloud/SaaS
Theme: Manage IT Services	**Theme:** Externalize Applications	**Theme:** Solutions & Process	**Theme:** Security, Connectivity, Openness
Barrier to Entry: High	**Barrier to Entry:** Good	**Barrier to Entry:** Better	**Barrier to Entry:** Better
Complexity: High	**Complexity:** Good	**Complexity:** Good	**Complexity:** Good
Integration: Proprietary	**Integration:** Open/Binary	**Integration:** Open/Binary	**Integration:** Open
Connectivity: Limited/Closed	**Connectivity:** Closed/Internet	**Connectivity:** Closed/Internet	**Connectivity:** Internet
Processing: Batch	**Processing:** Real-Time	**Processing:** Real-Time	**Processing:** Real-Time
Time to Mkt: Extended	**Time to Mkt:** Better	**Time to Mkt:** Better	**Time to Mkt:** Better
LOB Integration: Poor	**LOB Integration:** Poor	**LOB Integration:** Good	**LOB Integration:** Good
Market Relevancy: Relevant	**Market Relevancy:** Obsolete	**Market Relevancy:** Diminishing	**Market Relevancy:** Emerging
Primary Issue: High Costs, Lack of Agility	**Primary Issue:** Process Disconnect, Security, SLAs, Scale	**Primary Issue:** Process Disconnect, Security, SLAs, Scale	**Primary Issue:** New Programming Mode

(to name a few), there are ways to introduce lower risk, cost, and support for your company.

Externalize

As companies classify solutions through methods such as portfolio management and ALM, they will be able to determine which applications add strategic value. This will drive business decisions for those solutions. Several options are available to these companies; it will be up to architects to help IT and business leaders determine the best course of action.

A trend since the very beginning of IT is the notion of externalizing IT assets. We saw this with traditional outsourcing, application service providers (ASPs), and managed services providers (MSPs).

As Figure 7 shows, the concept of externalizing services, applications, or entire business processes is not entirely new but, instead, evolved.

For each solution, it is clear that there are distinct benefits and drawbacks. Some, such as ASP models, are just rendered obsolete, while outsourcing still strives in key problem areas such as the mainframe. What is most evident is that these technologies and methods are evolving.

There is no better time to start to think about externalizing IT services. Independent solution vendors (ISVs) and platform providers such as Microsoft (with its Live Services and Azure Cloud Services) solve traditional technology problems.

As with all technology solutions, if they are done correctly, SaaS and cloud-based solutions will provide companies with significant benefits:

- **Accelerated technology adoption**—The barrier to entry on access to new technologies is as easy as a subscription—a relatively small investment in prototype solutions on innovative technologies that in the past would have software licensing, procurement, deployment issues, and support staff training (among others) associated with them.
- **Complexity**—SaaS specifically enables companies to reduce the complexity in the solutions in which they would traditionally build or buy. Combining the power of strong Internet, security, and XML standards with SaaS-vendor solution expertise reduces many of the technical complexities that once loomed.
- **Lower total cost of ownership (TCO)**—By reducing the number of staff, acquisition of physical servers, software licenses, and overall operational costs, the TCO of solutions or entire IT services can be lowered.
- **Agility**—Allowing companies to speed up the delivery of solutions on modern technologies and reducing the complexities of line-of-business (LOB) application integration will make companies much more agile than ever before.

Consolidate

Pressure from all sides will force architects to think about doing more with less. This will include reducing the complexity and redundancy in the enterprise. As architects optimize their enterprise through portfolio management, ALM, and tool rejuvenation, they will look at ways to optimize and consolidate applications.

Key areas in which consolidation will occur are:

- **IT infrastructure.** The hardware backbone of an enterprise often is the first to be consolidated—sometimes, the easiest of all consolidation efforts with mainstream and ever-evolving virtualization solutions. While it is easy to virtualize, there is careful planning needed, as it could lead to the same problem that you have with server sprawl.
- **IT services.** Collaboration, VoIP, e-mail, business intelligence, portals, system monitoring, and project-management systems (to name a few) are all IT services that can be streamlined by creation of standards, consolidation of multiple vendors, or moving them outside of the firewall to a service provider.
- **Solution architectures.** Companies often find redundancies in solutions across LOBs or functional areas. Consolidating solutions will be key to lowering costs and complexity within the organization.
- **Process.** Process management often is overlooked, but it can be a valuable exercise to consolidate disconnected and redundant processes. This streamlines your architectural efforts by providing repeatable and predictable measurements.

Through these four architectural imperatives, weathering the turbulent economic storm will be tolerable. Benefits for architects are

huge. Not only will they be tightly aligned to the business, but they also will be seen as true business partners in IT. Architects will be able to add significant value in reducing TCO.

Conclusion

In the fragile economic state in which we find ourselves currently (as of this writing), enterprises can differentiate themselves with technology innovation and optimization. Those that capitalize on market conditions as an opportunity to streamline, consolidate, and acquire IT assets and/or entire businesses will have longevity well past the economic crisis.

This article has aimed to show architects a way to understand key forces on the business of IT, what they can do to add value, and the key areas of focus and technologies that will help them deliver value back to the business.

About the Author

Mike Walker (Mike@MikeTheArchitect.com) is a principal architect for Microsoft who is responsible for building the strategy for managing, delivering, and communicating the Microsoft position on enterprise architecture. He is responsible for driving Microsoft's worldwide Enterprise 2.0 and Enterprise Architecture strategies in key industry segments.

Mike joined Microsoft in early 2006. His background is as a financial-services enterprise architect and strategist, specializing in business transformation around technology, strategic infrastructure planning, portfolio management of technology projects, and solution architecture. As a thought leader, Mike combines this experience with a strong focus on strategic execution.

Visit Mike Walker's blog.

Follow up on this topic

- Application Platform Optimization
- Application Lifecycle Management
- Externalizing to the Cloud

IT Infrastructure and Application Optimization

by Ramnish Singh

The IT Infrastructure and Application-Optimization Model from Microsoft helps customers understand and, subsequently, improve the current state of their IT infrastructure and application and what that means in terms of cost, security risk, and operational agility. Dramatic cost savings can be realized by moving from an unmanaged environment towards a dynamic environment. Security improves from highly vulnerable in a basic infrastructure to dynamically proactive in a more mature infrastructure. IT infrastructure and application management changes from highly manual and reactive to highly automated and proactive.

Microsoft provides the technologies, processes, and procedures to help customers move up through the optimization journey. Processes move from fragmented or nonexistent to optimized and repeatable. The ability of customers to use technology to improve their business agility and deliver business value will increase as they move from the "basic" state up the continuum toward a "dynamic" state—

Figure 1: Optimization continuum, from "basic" to "dynamic"

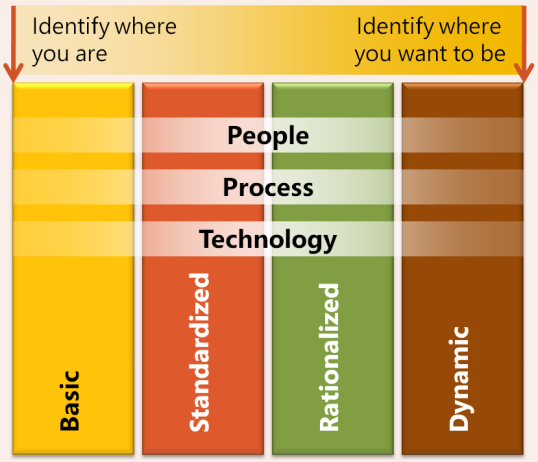


Figure 2: Microsoft IT Infrastructure and Application-Optimization Model

empowering information workers and managers, and supporting new business opportunities.

By working with Microsoft and using this model as a framework, an enterprise can quickly understand the strategic value and business benefits to the organization in moving from a "basic" level of maturity (in which the IT infrastructure is generally considered a cost center) toward a more "dynamic" one (in which the business value of the IT infrastructure is clearly understood and the IT infrastructure is viewed as a strategic business asset and business enabler).

The model focuses on business value and bridges the gap between IT and business organizations. Clear mapping is shown between business drivers and priorities to technology solutions that are integrated platform capabilities—enabling dynamic IT and delivering business agility. Sample architectures are discussed for the following industry or horizontal solution: Business Insight, Customer and Performance Insight, Documents and Records Management, Enabling Mobile Workforce, Enterprise Resource Planning, Healthcare, Improving Customer Service, Manufacturing Operations, Optimizing Business Operations, Optimizing Finance Operations, Performance Management, Public Sector, Sales, Store Systems, Supply Chain Management, and Web Platform.

Ramnish Singh (ramnishs@microsoft.com) is an MS Certified Architect in Bangalore, India.

Reuse, Buy, or Build: Components and Services

by Simon Thurman and Phil Rowland

Summary

The purpose of this article is to highlight and discuss the considerations that are required to make the appropriate choice among reusing, buying, or building components and services.

Introduction

Let us begin with a simple statement: All things being equal, the choice should be to reuse, buy, and then build components and services in order to deliver the required solution to the business. But not all things are equal, and that is the purpose of this article: to highlight and discuss the considerations that are required to make the appropriate choice.

Of course, this is something that all architects have been doing forever—albeit, probably with a view on fine-grained software components. What we want to do is clarify that choice for solution components and test whether it is applicable to the world of services.

In order to explore this subject in more detail, we have identified three areas that we think are worthy of discussion: an Evaluation Framework, Forces, and a Balancing Act. Because it is something with which we are all familiar, we will look at these in the context of components first. Then, we will move on to test whether the same thinking can be applied to services.

At the outset, we will make the assertion that ultimately it is a business decision—not one of technology.

Definitions

Here are the *Oxford English Dictionary* definitions of "reuse," "buy," and "build":

Reuse—"The action of using something again." In architectural terms, *reuse* can mean different things to different architects. For example, it could mean reusing an entire existing IT solution by creating another instance of it to satisfy some other business requirements. Alternatively, it could mean that a number of solutions use the same functionality that is contained within another solution to satisfy parts or all of their requirements.

Buy—"Obtain in exchange for payment." In architectural terms, the buying of packages has been commonplace for many years. For the purposes of this article, *buy* refers to the procurement of a software component or service.

Build—"Construct by putting parts or materials together." For the purposes of this article, *build* within the architecture community could be misleading to some. Therefore, *build* in this article relates to the development of a bespoke solution by using software-development tools and methodologies.

Assessment Structure

We think that the considerations and decision to reuse, buy, or build fall into the following categories: strategic, operational, business, and technical (see Figure 1).

Strategic considerations, which deal with the long-term vision and direction of an organization. How things should be without the external influence of other factors. These include:

- Market share and differentiation.
- Growth.
- Strategic change management.
- Managed portfolio.

Of course, there are other factors that determine the path that a decision can take. These fall into the operational category and include:

- Governance.
- Change management.
- Skills.
- Support and maintenance.

Alongside the strategic and operational considerations, there are those that fall into the business and technical categories. The business considerations are the ones that support the short- to medium-term objectives of the business. These include:

- Delivery and running costs.
- Contract management.
- Depreciating of assets.
- Internal or external provision.

Figure 1: Consideration categories

Table 1: Evaluation framework

	Positive	Negative	Considerations
Reuse	Delivery time Complexity Cost Maturity	Requirements compromise Maintenance Support	Governance Consolidation Culture
Buy	Delivery time Support Maturity	Requirements compromise Maintenance Complexity Cost	Integration Customization
Build	Requirements match Maintenance Support	Delivery time Complexity Cost Maturity	Skill Core

Most organizations have a technical strategy—one that states what technologies can be used and for what purposes. Considerations in this category include:

- Technology choice.
- Interoperability/Integration.
- Repository of assets.

While it is important to understand the higher-level considerations that affect the direction in which the business goes, it is probably more valuable for our purposes to explore a little deeper and identify the lower-level considerations. In order to do so, we have created an Evaluation Framework.

Evaluation Framework

The Assessment Structure offers a classification for the types of considerations within the context of the business. Now, we want to move on and examine the elementary considerations:

The Evaluation Framework identifies seven attributes that influence the sourcing decision. These are delivery time, complexity, cost, maturity, requirements compromise/match, maintenance, and support. Our hope is that the choice of words provides sufficient explanation of what we mean; therefore, we will not discuss further.

However, we do believe that it is valuable to discuss the entries in the considerations column. The idea here is to identify the high-order factors that should be considered when making the decision. It is also worth noting that the specific consideration might not be unique to the category with which it is aligned. So, for example, we recognize that governance is also a consideration when buying or developing an asset, but we believe that it is of higher-order importance when thinking about reuse.

In the reuse category, governance, consolidation, and culture are the identified priority considerations. Governance covers such areas as ownership of a specific business capability and data within the business. The owner can be thought of as the custodian of the asset who handles such things as change requests to both functionality and data. Identifying the owner should not be underestimated; this might prove to be a very difficult and challenging task and fraught with politics. The policy that describes the motivation and execution of the reuse strategy would also come under governance.

Consolidation describes the desire to minimize the number of assets that deliver similar functionality. This might be to achieve a variety of outcomes that range from reduction of overall cost to delivering data sources that present a single version of the truth. In order to be executed successfully, consolidation has strong links with governance.

Finally, there has to be the correct culture within the organization to achieve success when looking to reuse assets. This might result in the ability to compromise in order to meet the greater needs of the business, instead of completely fulfilling the needs of a specific project.

In the reuse category, it is also worth stating that some development effort might be required in order to achieve reuse of an asset; that is, it is not simply a matter of reusing an asset and, therefore, completely ignoring the other categories.

Integration and customization are the high-order considerations within the buy category. Integration is probably self-explanatory, when selecting a component to reuse it has to integrate with what is already in place and what is planned for the future. Particular attention should be paid to how interoperable the asset is. It is worth remembering that interoperability makes integration easier. Of course, integration is not only a technical consideration; it refers also to other areas, such as people, processes, and tools.

Beware of customization. A high degree of customization can eliminate some of the benefits that are allocated to the buy category and can introduce new disadvantages.

Again, as with the reuse category, it is highly likely that some development effort will be required when buying an asset. The amount might well depend on how interoperable the asset is.

Finally, the build category has skill and core as its two high-order considerations. *Skill* refers to the expertise that is required to develop an asset. This might be a simple binary decision: Do you have the skill or not? Of course, the answer to this question might not necessarily eliminate development as an option; there are alternative approaches, such as off-shoring. However, that is probably a different discussion. So, for the purposes of this subject, the decision might be somewhat easier if there is no skill.

If the business capability that is to be implemented is core to the business—that is, a differentiator—it is highly likely that the build category is the favored or only approach. It is worth noting that core does not automatically mean business-critical or vice versa. A capability could be considered business-critical, but not core to the business; a human-resources function might well illustrate this point.

We did consider including a fourth category, which was a hybrid of two or more of the categories. This is something that happens quite regularly within specific industries. For example, it is quite common in the oil and gas industry for two or more competing companies to develop something together and then reuse it among them. There is probably a lot to learn from these sorts of activities, although it falls out of the scope of this paper for us to discuss here.

In all of these cases, it is important to recognize the impact that these considerations will have on the ultimate decision, and that classifying them in this way provides a mechanism to ensure that an objective approach is applied to the selection process and that opportunities are not hidden from view.

Forces

Of course, there are other forces that influence decisions. We refer to these as the pushing and pulling forces, but some might consider

Cost-Effective Architectures

by Mark Bloodworth

The key concern for architecture is to deliver agility through stability—that is, to be able to accommodate the inevitable changes with the minimum effort and disruption. Traditional techniques such as loose coupling and high cohesion will certainly help, but they are not enough. Uncertainty is best dealt with by deciding at the last responsible moment; embracing uncertainty in this way will naturally lead to an architecture that handles change. Theories that have been designed to deal with uncertainty in other industries, such as Real Options, can also be usefully applied to architectural decision making.

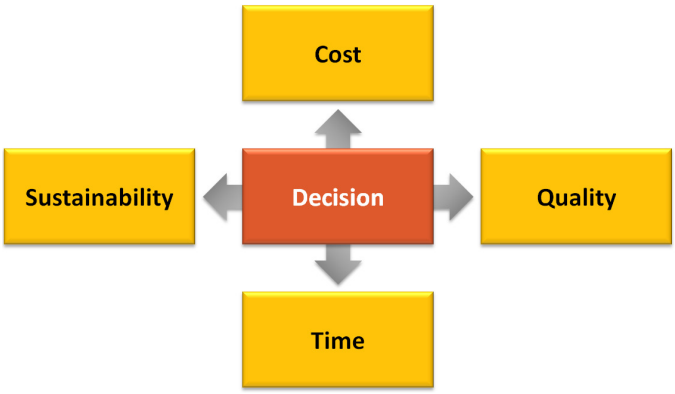
Coming back to the present, the most pressing questions are where to make investments and how to make them. Utility theory can be applied to decide where to invest: Having a group of people assign value and cost to the candidates is a simple application of this theory. The value and cost should be done in units; it is not important what the units mean, as long as they are consistently applied. It is also important to separate cost from value during the process. This technique provides a means of comparison, along with an explanation and record of how the decision was reached. It is also a technique that can easily be opened to all stakeholders (some might not be able to make value assessments, and others might not be able to make cost assessments, but the overall contribution is what matters).

When this decision has been made, we have to decide whether to extend an existing system or service or introduce a new system or service (with the accompanying build-versus-buy decision). It is worth noting that mapping function to business area tends to lead to greater conceptual integrity, which in turn makes the architecture more readily understood and durable. Implementing the fewest features possible is a good principle to follow—one that fits well with taking decisions at the last responsible moment.

The past should not be ignored when facing architectural decisions. By retrospectively analyzing architectural decisions, trends and issues can be identified. Systems and services that are hard to change can be noted, and strategies can be devised for how to replace them—and thus reduce overhead. Having kept a record of how previous decisions were made, it is possible to analyze the efficacy of these decisions and improve the decision-making process.

Mark Bloodworth (marblood@microsoft.com) is an Architect Evangelist in the Development and Platform Evangelism Team at Microsoft in the U.K.

Figure 2: Pushing/pulling forces, or drivers



them to be drivers (see Figure 2). These forces pull and/or push solution decisions in particular directions; while some might see it as less than ideal, this is unavoidable.

Each set of drivers can be considered unique to an organization; typical example drivers are as follows:

- **Cost**—The drive to remain cost-neutral or, more likely, more profitable—for example, affordability, return on investment (payback), and finally the cost implications of not doing it.
- **Quality**—The drive to achieve an outcome that meets or exceeds the quality expectations of a customer—for example, 99.999 percent availability.
- **Time**—The drive to be first to market to help differentiate from competitors.
- **Sustainability**—The drive to reduce energy consumption and improve the carbon footprint of an organization.

The intention of these and any other Forces that an organization identifies is to ensure that they are Simple (that is, make sense to an organization), Measurable (that is, an organization can recognize when it has been met), Achievable and Realistic (that is, it is within an organization's gift to make it happen), and, finally, Timely (that is, it can be delivered within an agreed timescale). A good rule of thumb is to ensure that each chosen Force is SMART.

Balancing Act

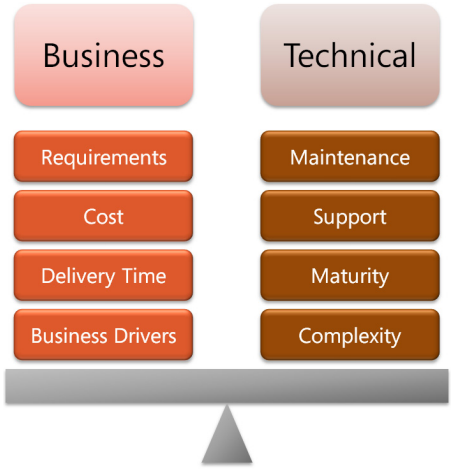
Ultimately, any solution decision that is made by an architect is often a balancing act (see Figure 3) among the seven attributes, the high-order considerations, and the forces.

It will come as no surprise that we do not believe that there is a silver bullet whereby an architect can feed in all of these inputs, turn the handle, and out pops the answer; neither do we believe that any of these three has a higher priority than another.

So, where does all of this leave us? As with any decision-making process, we believe that it is important to ensure that solution decisions are transparent and clearly communicable and understood by those who need to know and must buy into them. Our approach has led us to the conclusion that use of the Evaluation Framework, in conjunction with an appreciation of the organization's Forces, provides a simple evaluation tool for architects to use to provide a balanced proposal.

In no way is this approach considered to be a substitute for the necessary rigor that is required for architecting solutions and does not detract from the need for a richer, deeper understanding of architecture and use of methodologies that exist and that architects use today and will use in the future.

Figure 3: Depiction of balancing act



For example, measurement of each SMART Force can be of particular importance and relevance in the evaluation of solution options. A Spider Diagram, such as the one that is shown in the following figure, can help communicate the relevant fit that a solution has to an identified set of business drivers.

In the first example (Figure 4), the Spider Diagram would help an organization in which quality is important—perhaps an engineering company—identify that a solution component is not of the quality that it can accept and, therefore, cannot be used.

In the second example (Figure 5), the same engineering organization can be reassured that the solution component is of high-enough quality for its intended use and it would be prepared to accept it. The production of Spider Diagrams such as these for each solution option can help visually highlight which one provides the best fit.

Services

Ever since there has been an architect role, architects have been answering the questions that we have posed here. In recent years, we believe that the question has been more relevant to software components, and not services, which has been the basis of our paper so far. However, the emergence of services and such things as cloud computing has and will continue to make the reuse, buy, or build dilemma more relevant than ever before.

Arguably, the starting point with services is somewhat different from that of components—specifically, in the reuse category. If a company already uses a service, the chances of it reusing that service somewhere else is incredibly high. Take, for example, any common data set that numerous solutions use—perhaps a credit-checking facility in the banking industry or a list of dealerships in the automotive industry. Design and implementation of these as services means that they can be reused by other systems within an organization. It is also not inconceivable to suggest that it might also emerge as a new product in an organization's product portfolio for other organizations to buy—for example, an address-lookup service for resolution of postal addresses. This opens up a further discussion (for another day) on a possible additional dimension to the Evaluation Framework that addresses the external and internal consumption or provision of services.

All of this raises the question: Should the capability be delivered as a software component or service? All things being equal, if the motivation is to reuse, the answer might be quite simple—although this, of course, can be influenced by a number of factors. While the decision of component or service is an interesting and important one, what we are concerned with here is whether to build your own service or buy one. Also, can the experience and rules that we have used in the past be applied?

Conclusion

In this article, we have discussed three main areas: Evaluation Framework, Forces, and Balance. Looking at buy or build, let us address each area in turn and its applicability to services:

Let us begin with the Evaluation Framework. It would seem that the seven attributes apply to services in the same way that they do to components; the considerations, however, might not.

For *buy*, integration and customization were the identified high-order considerations. This remains the case for integration; however, customization should not be a priority concern. If it has to be considered, perhaps there is an issue with the choice of service; for example, the compromise on requirements is too much to be acceptable.

The situation is different for the *build* category. The high-priority considerations here were skill and core. Both of these remain important factors for services. As with components, the ability is required to build a service; and, if that service is core to the business, it is highly likely that the build option is the preferred one.

Services probably warrant an additional dimension to the Evaluation Framework in order to apply further considerations.

Service-level agreements are always a hot topic with the subject of services. Does the service provide the required availability, reliability, and scalability that the business demands? If we consider the previous credit-checking example, if this service is part of a wider business process (which is likely to be the case) and the end-to-end process must complete in a fixed time, service issues such as reliability are a vital consideration.

Figure 4: Forces diagram—solution option 1

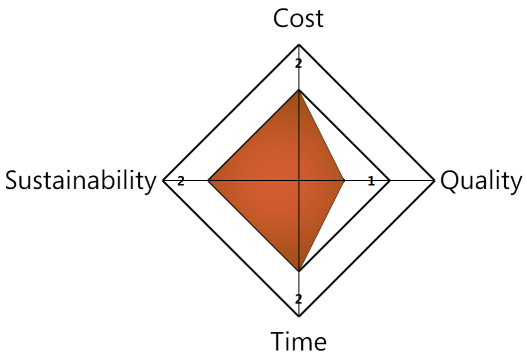
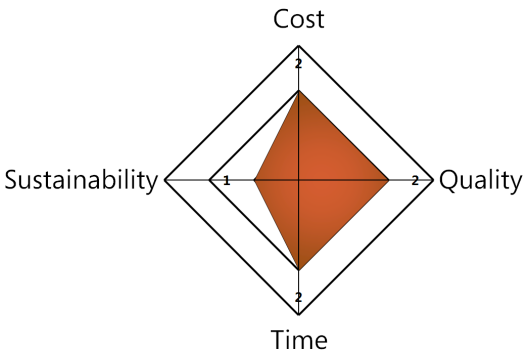


Figure 5: Forces diagram—solution option 2



We have noticed also that companies that already partner with other organizations in the course of their business are comfortable with a services model. They can clearly see the advantages, understand the approach and risks, and have a willingness and culture to succeed. One example of such a business is the airline industry. Today, that industry works with numerous partners in order to deliver the end-to-end service to their customers—for example, check-in at the start, the flight itself, and finally baggage reclaim at their destination.

Reflecting upon what we have discussed, we believe that both the Forces and Balancing Act are as useful when considering services as they are for components. Architects, then, should be confident in taking their knowledge of and past experiences in designing software components forward into the design of solutions that encompass services.

About the Authors

Simon Thurman, (simont@microsoft.com) Architect Evangelist—Microsoft, is an Architect in the Developer Group. Over the past 20 years, he has architected and developed software on mainframes, client-server architectures, and the Internet. Such memorable software includes ensuring that you have the correct address, routing algorithms, and streamlining the house-moving process. Today, he helps customers do innovative things with new technology to deliver business value. The rest of his time is spent with his wife and daughters, and mountain-biking.

Phil Rowland (philip.rowland@capgemini.com) is an Enterprise Architect at Capgemini UK plc. For the past 20 years, he has successfully undertaken a number of roles, including presales, technical strategy, technical consultancy, business analysis, solution design, software development, and implementation management. Phil's breadth of knowledge and experience has enabled him to design a number of highly available, scalable, and robust solutions to clients in the public and private sectors, as well as providing advice and consultancy for Enterprise and Solution Architecture.

Architects and the Economic Drag

by Nirmala Desai

Summary: In these times of global budget restraints, here are a few tips for architects to help them meet their regular challenges, as well as "do more with even less." This article focuses on investing wisely in strategies and new technologies that will modernize the business, followed by a short note on areas that help reduce total cost of ownership (TCO).

Going up the value chain—The primary way to go up the value chain is to use technology to generate revenue for the business, whether it is by providing an additional channel for marketing on social networking sites or by promoting content on mobiles. The secondary way is to reduce maintenance costs by automating labor-intensive processes, and then optimizing and promoting reuse of technical components.

Addressing the challenge of keeping that IT spine supple—You might already be looking at Business Process Management and Service-Oriented Architecture. While they have not proven a pill for all ills, they have provided process-automation flexibility and interoperability features that have proven a definite advantage. Enterprise architecture is a result of evolution. To tame this animal, you could create domain models and use them to provide a direction to the overall architecture—thus saving a lot of redundancy—or you could build an asset library that promotes reuse in the organization.

Addressing legacy-modernization challenges—Along with budgetary concerns, lack of know-how is also a challenge. While there are tools that help with Model-Driven Development, code parsers that can draw up a model from legacy code require intelligence that is not yet available. Product architects can seek to innovate on tools and technologies for transforming legacy applications to more modern platforms. Enterprise architects can review their modernization needs and strategize to build long-term partnerships with relevant product and service vendors.

Addressing collaboration needs—Compared to e-mail, collaboration technologies provide even faster ways of approving documents and making decisions by removing the multiple to and fro that is required among different people. More recently, Web conferencing is meeting these needs. However, there is a need for being able to conference with people who are not necessarily using the same infrastructure.

Implementing unified communication—While the advantages of collaboration tools are well known, unified communication has yet to reach a stage in which it is well understood by infrastructure and implementation teams. This might be a good time to review the tools that are available and prepare a business case for their implementation.

A scenario in which this can be beneficial is one in which a project manager needs—for a short time, and on an urgent basis—a resource who has a certain skill set. The PM could quickly locate the necessary person by accessing an interactive directory (not on the same domain that the PM is on), converse on IM, and then proceed to a voice call or even a video call—all from the desk of the PM, and in minutes. This resource might be located in another part of geography and be made available for the task either virtually or physically, and on a contract basis.

Preparing for the upturn—This is something that not too many people are considering. The economy will begin to prosper again when a new financial system is in place that is more transparent and, perhaps, better averse to risk. People are withholding investments not because they believe that the system is contaminated, but because they are afraid. At this time, it seems to be a wise perspective for businesses to anticipate the need for transparency and build clear risk-assessment and risk-mitigation strategies, and for architects to prepare their systems for the same—not only in the financial sector, but in all industries.

Reducing TCO—Areas that help reduce TCO include virtualization, which makes physical availability of systems redundant, and cloud computing, in which businesses can offload maintenance of their hardware and software systems and focus on what is their core business. You could also leverage your existing ERP systems to provide more value. "Cutting the flab" might require you to consolidate your IT landscape, consolidate your master data-storage locations, review your integration mechanisms, and make them all less complex.

Nirmala Desai (Nirmala.desai@patni.com) provides project management and technical consultancy to customers worldwide. She is currently involved in developing innovative methods to modernize legacy applications to the Microsoft Platform.

"Live Long and Prosper": Customer and Employee Retention with an Agile Process

by Rodney Guzman

Summary

This article profiles the evolution of a professional-services company this is focused on software-application development.

Introduction

Building software is an art. But like a building, a well-built piece of software will stand the test of time without constant upgrades and maintenance. Unlike construction, however, each building that you create is unique. If software developers cannot cookie-cutter their software applications, much like tract homes, how can they become more efficient? Being more efficient allows us to create higher-quality software in less time. And less time means less money. From building libraries of reusable controls to executing an agile-development process to growing employees with the help of mentors, every aspect of software development can be improved to lower the cost of software creation.

This article is a profile on the evolution of a professional-services company that is focused on software-application development. Over the past 10 years, this company has had to become more efficient to remain competitive and unique in a market that has commoditized many aspects of its offerings. Many hard lessons were learned, but what has emerged has allowed the company to continue to thrive even in today's economy. At the core is a hybrid agile methodology that is used to deliver everything from enterprise-size applications to one-week demo applications. This is powered by like-minded software and system engineers, as well as project and engagement managers.

Discover the process that this company uses today to build its software consistently on time and on budget.

Make the Customer Your Ally

Your customers can be extremely diverse, having very different backgrounds and expectations on how custom software development is performed. Or you might be lucky and have only internal customers who are intimate with your software-development process. Regardless, your work will be remembered by your ability to hold to your word of when what you say will be delivered is delivered.

What must be acknowledged is that no matter the quality of the technical solution that you produce, success will be measured by how you meet the expectations of your customers. The process that you use to create custom solutions should actively engage the decision makers who will be passing judgment on whether or not the outcome is successful. Your customers should never be surprised or embarrassed. They can be surprised because the feature that they thought was going to be developed was not developed, or a deliverable was late and they did not know about it ahead of time. They can be embarrassed because promises that they made were broken, due to a feature not being delivered when it was supposed to be delivered.

Custom software development can take on a life of its own after the project has begun. It is well known that it is quite impossible to detail every requirement of each feature before any code is cut. Gone are the days in which weeks (or even months) are spent on requirements documentation that quickly becomes obsolete during the project. The software industry has been alive for a long time now, and your customers are much wiser. The concept that the features mature as the project progresses is not foreign, and your customer will expect you to be able to react to change during the project. Keeping the customer engaged and part of the decision-making process for these changes is what must be done.

Everyone knows that change will occur during the building of a custom solution, new features will be born, and priorities will change. Embracing this fluidity and immersing your customers in it allows everyone to be on the same page. When your customers want a new feature, it will be placed on a feature backlog that is to be built. A priority will be assigned to it and an acknowledgement made by everyone that without a change in scope or timeline, other features will not be developed in the final deliverable. By allowing your customers to change priorities during the project and helping them understand the ramifications of performing that action, they become part of the solution and will become your ally.

Your customers will feel enabled by being able to affect the project. They should be kept engaged with status meetings as frequently as you can (for example, twice a week) in order for them have a constant pulse on the project, and not by weekly status reports that no one ever reads. If they are part of the creation process, and not just there to receive the deliverable at the end of a sprint, your customers will already know what will be delivered to them before they see it and they can actively set expectations on their end. The risk for performing rework to appease a customer is significantly reduced, you complete the project when you said that you would, and your project team can be released on schedule to work on another project. Your customers will ask you back for repeat business and will be a reference for new customers.

It can be perceived that what is being endorsed here is a recipe for a never-ending project, which could not be further from the truth. The goal of the process is to guide the airplane that is your project to a scheduled landing in one piece, entirely intact, and as close to as what the customer wanted. The fact that some details might be missing from the final product will have been a decision to which your customer has already agreed.

The Actors

People make the process run. Certainly, there are such software tools as Microsoft Team Foundation Server that enable the process, but one cannot expect to become efficient and organized by just installing and using a product. There are key actors in this dance who allow an agile process to function: project managers, developers, and—for lack of an industry-accepted term—engagement managers.

The Job that Few Want and Even Fewer Are Capable Of

Many developers view project management as a necessary evil. It is apparent that the expectations of a project manager (PM) vary tremendously from company to company. Here, they help bring order to chaos. They are the unsung heroes of delivery and are leaned on by the developers to be the bad guys. They ensure that no detail is left undone and hold both the development team and customer accountable.

After the project has begun, the PM is the main point of contact with the customer. The PM brokers all conversations with the customer (including being CC'd on all e-mail messages) and is the final decision maker for anything that can affect the schedule and scope. The developers are shielded by the PM from the political minutiae that often occurs during the project. The PM initiates and facilitates conversations with the customer when the developers want to push back on how a feature is interpreted.

Constant communication is the mantra of the PM; protection against misinterpretation of these communications is paramount. Each non-written communication can result in key decisions being made and the scope changing, as well as actionable items upon which the team is relying in order to meet the interim deadlines. These interactions must be documented and echoed back via e-mail to all of the participants in order for everyone, including the customer, to understand how the team interpreted the conversation. The first time that you must lean on one of these e-mail messages to show why something that the customer did not expect occurred during the project will be the last the time that you need any evidence that this is a worthwhile activity. Not having this to lean on is much worse and could cause you to make up, on your dime, the time that is necessary.

For the team, the PM will monitor the progress of each interim deliverable. Progress should be measured on a daily basis, as interim deliverables (to keep the customer engaged at the appropriate level) are delivered every one to two weeks. Your developers should report their hours at the end of each day, in a way that will show how much energy was put into their tasks and how much more time they estimate it will take to complete it. Each morning, your PM can gut-check the cadence of the project and the momentum that is being made toward the delivery of interim milestones. Minor course corrections—such as providing additional help to developers, so that they are not spinning their wheels—can be made on a daily basis. Keeping everyone moving toward the interim deliverable is the job of the PM.

Developers, Developers, Developers, Developers

For an organization that creates custom software, nothing is as expensive as replacing a developer. The process is designed to protect the developers from outrageous commitments, miscommunications between the customer and the team, and consistently working long hours. Fostering a climate of constant feedback, participation, and ownership will quell feelings of hopelessness and simply not caring about the outcome of the project.

Many developers wield superhuman powers when it comes to how fast they believe they can accomplish tasks. Too often, they do not account for the time that it takes to discuss details, have meetings, write documentation, unit-test, and deploy code. Just as you should include your customer as an active member of your project, so should you include your developers. Providing the entire team with a sense of ownership of the deliverables is what you should strive toward. For developers, ownership can be obtained by allowing them to perform estimates for their tasks. At first, their estimates will be way off, and they will pay the consequences of working longer hours to make up for it. With just a little experience, they will learn to provide estimates that are much more accurate.

The end result is a sense of ownership of the deliverables, and this fact cannot be overstated enough. This is software development, after all, and not everything always goes according to plan. Sometimes, additional effort is required to get the train back on track. The process of constant communication between the team is designed to catch the train before it fully derails. However, team members will sometimes need to work extra hours to get back on track. If they own it, they will be much more likely to want to fix it.

On each project, there is one developer who has a special role: This is the technical peer to the PM. This person leads the team of developers and is responsible for the design, architecture, and development of the more difficult portions of the project. Code standards are enforced by the lead, who mentors the development team and provides code reviews as needed. The technical lead reports to the PM and, without consensus with the PM, cannot stray from the tasks that have been agreed-upon. Your customer will often form a special relationship with the technical lead—sometimes, even trying to circumvent the PM by going directly to the lead. The lead is responsible for ensuring from the team estimates that are as accurate as possible.

The People who Buy Lunch

The role of an engagement manager is to form a relationship with your customer that spans all past, current, and potential future projects. More than a salesperson, these managers understand the customer business and the technical details of what is being delivered. They receive honorable mention here, because an engagement manager is the one person who the PM and technical lead can leverage to persuade the customer to align expectations with the team. The fact is that not every detail of your project can be documented. Many conversations, agreements, and understandings that never make it to paper are formulated when the project is first envisioned.

The engagement manager is the constant reminder to the customer of all of the past discussions and agreements, and will keep the spirit of the deliverable alive that was agreed upon when the contract was signed. This is acted upon during customer meetings, which the engagement manager always attends. The technical lead and PM work closely with the engagement manager, so that preemptive strikes can be made to manage the expectations of the customer. This is an active process that is an insurance policy on keeping the deliverables on time and on budget. The triad of the engagement manager, PM, and technical lead is the team that constantly manages the expectations of the customer and actively works together to keep the deliverables to the timeline and to the budget.

Becoming More Efficient

You should expect your process and your people to become more efficient. Process refinement is not something about which you will meet every quarter to decide what to change; it should be done at

the end of each project—sometimes, facilitated by a lessons-learned meeting. For example, you might decide that you should categorize your tasks in a more standardize manner from project to project, so that people can think about how they will the organize projects more consistently. Or you might begin to see patterns of blown estimates in your projects, which should be acknowledged in your lessons-learned meeting.

For example, drag-and-drop touch-based functionality is hard in Windows Presentation Foundation (WPF). By putting metadata on your tasks to classify the type of feature that it is, such as drag and drop, you can report on these more effectively across multiple projects. By having a greater understanding of your risk areas before you start a project, you would be able to make strategic decisions on whether or not to accept those features, or provide a proper estimate to accommodate for it. These are just some simple examples of how to make your process more efficient. But the real improvements that you will see are by making your people more efficient.

All individuals bring their own unique strengths to the project. Knowing what those strengths are and how to maximize the potential of each person comes only from working with the sample people over a period of time. The investment of bringing new people into the fold is not quantifiable, but it is felt by everyone, so that retaining the quality people that you have is extremely important. When you are running a project, putting people in roles and giving them tasks at which you know they will not fail is critical; no one is demoralized more by being set up to fail. This village knowledge of what everyone brings to the table happens over time. Your efficiencies will come by being certain that everyone is aware of your strengths, the strengths of your team, and the weaknesses that people want to fortify. Asking someone to become better at something in which they have zero interest is a recipe for frustration.

Technical efficiencies also should be gained over time. Developers will get better at their craft when they see more and more projects that reuse the same technologies and tools. They will see architectural and design patterns in their projects that they can leverage from one to the next. Best practices will become second nature. Developers will get better at asking the right technical questions ahead of time to reduce risks down the road. However, the technology suite of today is so vast that no one person could be the master of all. Your developers will need to be focused in one area, such as WPF, in order to become masters of their craft. Asking a developer to dabble across all technologies can be done, but gaining efficiencies is much more challenging, because patterns that are evident will no longer be so.

Employee retention is key to becoming more efficient. Developers can be fickle beasts, and they must be constantly challenged and inspired. Consider the scenario of running a large 12-month project in which your best developers are engaged. This team is stuck with the same customer, solving the same business problems and having very similar conversations for a year. The technology could be cool, but everyone gets into a groove after a few weeks, and each week feels like the last. To keep these people inspired, consider reserving a handful of hours each week to allow the entire team across all projects to collaborate together on projects of their choosing. This type of freedom will liberate them from feeling lost in long-term projects. This time will become the most precious thing that you can give to your developers. It will boost morale and help employee retention.

Distributed Extreme Programming

by Ridi Ferdiana

Nowadays, agility and collocation have become daily values in software development. Agile methods such as Extreme Programming (XP) get so much attention that it shifts the conventional paradigm into a dynamic, incremental, and "working software" principle. Collocation in other sides gives a new way for a company to build good software from talents that are separated geographically. This well-known technique is called Global Software Development (GSD).

Distributed Extreme Programming (DXP) is an alternative development approach that integrates the XP method with GSD method. Integration of these two methods (which have contradictive elements) is linked by a communication pattern and assistive tools in the DXP term and discipline. DXP is divided into two main parts: DXP process framework and DXP supporting tools.

DXP process framework consists of four components: requirements engineering, architecture design, project planning, and product development. In the requirements-engineering stage, DXP emphasizes distributive compilation of user stories. This step generally starts with a draft user story that is created by the member or team that has direct communication within the client or has the knowledge on that domain and can initialize it. The result then will be discussed through direct communication lines, such as video conferencing or instant messaging (if bandwidth is limited).

The layout of the user stories then will be arranged in an iteration plan document. The iteration plan document is the artifact in the project-planning process. This document will be discussed with the client in the form of a "Planning Game" through Internet conferencing, and the result will be legalized by e-mail. Those two artifacts will be the input in the architecture development through a task card and class-responsibility-collaborator (CRC).

In the product-development stage, the concept of pair programming can be done through a technology such as shared view; however, when an Internet connection is not enabled, the concept of "to-do implementation" can be done. "To-do implementation" is an indirect pair-programming development concept. Generally, this technique is performed by a pair that takes the roles of thinker and executor. The thinker will make the code path framework, based on task card. This person also makes the unit-test without implementation. The framework will be stored in a custom metadata. The metadata will be read by the executor, who will implement it. The process is both iterative and incremental.

Determination of DXP tools becomes a conditional agenda for every team. A team that has an adequate budget can acquire a cool IDE, such as Microsoft Visual Studio Team System, with other communication tools, such as the Unified Communication platform. A team that has limited resources can use a technique such as metadata insertion in e-mail or Microsoft Windows Live services, or perform a custom development. The combination of tools and processes can be challenging and has a unique value on its implementation.

For further discussion about this, you can visit me at Ridi's blog.

Ridi Ferdiana (ridi@mvps.org), Microsoft Innovation Center, Electrical Engineering Gadjah Mada University.

Task Estimation

There is no book, class, or training that will make you good at estimating. Every piece of work that you will estimate has unique nuisances that cannot be cookie-cut from project to project. Certainly, the approach that one takes to estimate can be standardized and book-learned, but knowing when a project will take longer (or example, because of a demanding customer) comes from experience. The challenge is to ensure that enough time is being allotted to the project to accommodate the fluidity that naturally occurs and that you want to promote.

When the project is first discussed with the customer, many features must be called out with assumptions and risks identified. Before a contract is signed, consider documenting this information formally and providing it back to the customer to ensure that there is a common understanding of what is going to be built. This will provide additional touch points with your customer and allow further conversations to be had and trust to be earned. The person who is authoring this document will most likely become the engagement manager for this project. The details of this document can make their way into a contract that you would sign with your customer.

As soon as everyone is on the same page, but before a contract is signed, an estimate must be created. Have as many eyes involved as possible at this point. Ideally, you would include the PM and technical lead who would be running the project in this part of the process. A sense of ownership of the deliverables for the agreed-upon estimates will be derived from their involvement in this part of the process. In contrast, if an estimate is performed in a vacuum without proper participation from the team, something can be easily missed, and the project turns into a death march. The engagement manager will become extremely unpopular by pressuring the project teams with wrong estimates.

The milestones and schedule of the customer are factored in. An attempt might be made to run the development team as small as possible to avoid the inefficiencies that larger teams can have, but sometimes inefficiencies cannot be avoided. The schedule might be more demanding and therefore warrant a larger team to complete the work more quickly. The number of developers on the team, combined with the read of the engagement manager on how difficult the customer is to work with, factors into how much project-management time will be required. There might be additional resources, such as an artist or quality-assurance engineer, that must be factored into the estimate. Your own weighting factors for these additional types of resources must be developed. For example, for every 100 developer hours, you might 10 ten artist hours.

When development hours are derived from the identified features, it is difficult to accommodate everything that occurs around software development. This includes frequent meetings to discuss design and architecture, unit testing, and code reviews. To formulate a weighting system that is appropriate for your organization, use your past projects for the forensics on how long it took you to develop a feature versus how long you estimated that it would take. When you think about how long it takes to develop a feature, think of it in raw developer hours—without the meetings and so on. When the raw developer hours have been captured, factor in a weighting for the additional items that make your process work.

The type of project will factor into the weighting that you utilize. A production quality application is different from a proof-of-concept or demo application. A proof of concept does not need the same rigor as a production application. For example, corners can be cut in the architecture and design for a proof-of-concept application, and you probably will not need the same level of unit testing and documentation.

An estimate is only a guesstimate that is founded on the experience of many. Be prepared to make mistakes on the road to getting better at estimation, but set yourself up with the ability to learn from your mistakes. Review your estimates with as many people as possible before you commit to them. Use the process to hold the expectations of everyone to those estimates. Sometimes, mistakes will be made, and only longer hours will accommodate it. The goal is to drive these incidents down to the point at which they are anomalies and not the status quo.

Project Execution

Agile process is about keeping things simple, communicating every detail, and getting work done as quickly as possible. There is a simple formula that you can follow to streamline execution of your project. Every step of the process is intended to be action-oriented and produce tangible results. You are not expected to waste time in long meetings or writing novels of technical documentation; your focus is on building software quickly, in such a way that everyone (including the customer) has deep awareness of the direction of the project.

Kickoff

By the time that developers start learning details of the project in an official kickoff, many ground-setting actions have already occurred. The engagement manager, PM, and technical lead have already formed an action plan. The triad has received the customer-signed contract and might already have had numerous impromptu meetings to discuss the details of the project. Expectations might already have been set with the customer on what high-level features must be delivered, and when. The triad now must deliver the details to the development team. This will be most likely one of the longer meetings that you will have.

Different features are discussed by the technical lead. From the available resources, your technical lead should have asked for people who would be best suited for the technology that is being used on this project. High-level feature areas—such as different groupings of screens, Web services, and data sources—are called out in the meeting. People are assigned to the different feature areas. During the meeting, if it is appropriate, you can let your developers talk amongst themselves and decide. When developers choose the work on which they will work, a sense of ownership of those features is derived. As with the estimates, strive to promote a sense of ownership throughout the entire team, at all levels.

When feature areas have been assigned, you are ready to switch gears into creating a feature backlog and sprint-planning discussion.

Planning

The PM, technical lead, and (potentially) development team will meet to create as comprehensive a feature backlog as possible. A feature backlog is a list of features that is shared with everyone and that will be built in all of the development sprints. Based on everyone's understanding of customer priorities, a priority level is assigned to each feature in the backlog. Additionally, features that are dependent on one another are called out. Riskier features are tagged as high-priority items that should be completed near the beginning of the project to avoid dependent complications down the road.

When the feature backlog has been completed, it is reviewed with the customer for completeness and accuracy. The customer can choose to change the priorities at this point and ask for clarification

on how the features were documented. There is a tipping point here, when a customer sees the feature list and has a deeper understanding of all that they are going to get for their money. A comfort level is maintained with the due diligence that your team has already performed during the contract process, and is affirmed with a feature backlog list.

This honeymoon stage is the perfect time to discuss how low-priority features will be handled; this where most of the fluidity will occur during the project. There are low-priority features that simply will not make it into the final deliverable. The team will do its best to do this, but it simply will not happen because of its being unplanned during the project. Low-priority features will be scheduled with the related higher-priority features in a sprint, because it would make sense to complete them while working in related areas. If a low-priority feature does not get completed in a sprint, it is rescheduled to another sprint. In order for this to work, your customer must be engaged within the sprint to see why a low-priority feature did not make it into the sprint.

When the feature backlog has been agreed upon, the features for the first sprint are identified. The technical lead works with the development team to break into tasks each feature for the next sprint. Each developer who is assigned to a feature area, in conjunction with the technical lead, develops a task list. The technical lead attempts to standardize the tasks that are created in terms of how they are worded and their duration. Strive to make your tasks no longer than a single workday, as you will want to be able to measure progress of a project at a daily level.

There is a lot of back and forth on the estimate between the developer and technical lead. Your technical lead was already involved in providing a higher-level estimate and has a good pulse of how each feature should be developed. When the technical lead and developer are in agreement with the estimate, a sense of ownership of that estimate is derived from the developer, because the developer had a strong hand in developing it. These interactions can occur in a group style, so that each developer is aware of the details about the different functional areas and will be able to better meet the expectations of the technical lead when it is the lead's turn to help provide a task list and estimates.

During this planning phase, multiple things have occurred to better help you deliver your project. You have further endeared yourself to your customer, who has now been exposed to the details of how the software will be delivered. You have also endeared yourself to your development team by having them participate in the planning process. Both will help when dealing with the unplanned. You will be able to lean on your relationship with the customer when things get rough, and developers will go that extra mile without asking to get things right. This will only assist you in delivering your project on time and on budget.

Sprints

A sprint is a focused effort by the team and customer to produce interim deliverables that can be reviewed. A sprint could be two days long or two weeks long, depending on the features that have been identified. At the end of a sprint, a deliverable is handed to the customer. Sometimes, there is very little to show from a user-experience perspective; but handing over a deliverable to your customer is still paramount to keeping their trust.

At the beginning of a sprint, developers know what is expected of them to complete their features for that sprint. When the sprint kicks off, developers communicate the order in which they will be completing the features. Any dependencies between developers are called out, and the technical lead can choose to reorder the features within the sprint.

Each day of a sprint looks and feels very similar. The team comes together for a brief meeting to discuss what was accomplished the previous day, what will be accomplished that day, and any risk area that has been identified. If further discussion is required on a particular item, a separate meeting spinoff occurs with affected team members, so as not to waste everyone else's time.

At the end of the day, developers are required to report what hours were spent on what tasks and how many remaining hours they have on those tasks. The PM examines these hours and discusses them during the brief meeting. This gives the PM a sense of the cadence of the project to ensure that the deliverables for the sprint are on track.

If it becomes known that a low-priority feature will not make it into a sprint, a discussion with the customer occurs in one of the status meetings as soon as possible, in order for everyone to participate in that discussion. This is not necessarily a common occurrence, and it typically should be avoided in the first sprint, just so that the customer does not come to believe that this is normal.

Conclusion

Building great software requires not only a development process, but also a process to manage your customers effectively. The process is powered by people who are bought in and who believe that following it produces results, reduces chaos, and allows them to succeed. Managing the expectations of the customer is as important to the success of the project as the technology work itself. Having the right people in the right positions who love what they do makes the process work. Focusing developers in a technology area allows them to breathe the same issues, solve the same problems, and see the same patterns.

Ownership from everyone is key to promoting a well-running project. Your customers will be willing to make sacrifices during the project, because you have empowered a relationship with them through your process. When you need them to, your developers will go that extra mile when the time comes. No software-development process or plan is perfect, because no project is the same as another; each is a custom piece of artwork. Your process should embrace this uniqueness of each project and the changes that naturally occur during it.

About the Author

Rodney Guzman (rodneyg@interknowlogy.com) is the CTO and cofounder of InterKnowlogy.

Follow up on this topic

- Tools for Agility (Kent Beck)
- MSF for Agile Development
- SCRUM Process for Visual Studio Team System
- Agile Processes in Offshore Software Development

Surviving Turbulent Times: Prioritizing IT Initiatives Using Business Architecture

by Martin Sykes and Brad Clayton

Summary

This article describes a technique that Microsoft architects around the world use every day to ensure that IT projects have both a clear alignment with the strategic goals of the organization and a strong demonstrable impact.

Introduction

In turbulent times, budget pressures increase and IT spending is scrutinized. We will describe a technique that Microsoft architects around the world use every day to ensure that IT projects have both a clear alignment with the strategic goals of the organization and a strong demonstrable impact. This is used to prioritize the project portfolio to maximize return on investment (ROI) and determine the projects that should receive budget and resource allocation.

Methods include Microsoft Services Business Architecture (MSBA) to identify **what** to change and the Benefits Dependency Network (BDN) to understand fully **why** the changes must be made. This leads to a clear alignment with **how** the projects will make those changes. By using these methods, we identify the people, processes, and technology that are to be changed and provide clear alignment with organizational goals so as to achieve both financial and nonfinancial benefits. Many organizations have subsequently incorporated these methods into their project management and portfolio-prioritization processes.

We will use a fictitious retail bank, Contoso Bank, where we are working with the Chief Information Officer (CIO) and the enterprise architecture (EA) team to address two needs:

- Provide a holistic way to review the IT portfolio and ensure that resources are focused effectively. We will create a common model that describes the business capabilities and associates the supporting people, processes, and technology. This will allow us to see areas in which duplication exists, as well as opportunities for reduction in cost and complexity.
- Produce a method for reporting on IT activities, so that stakeholders understand both how IT investments are structured and that we are realizing the predicted benefits.

Figure 1 provides an overview of the process. The two key methods work closely together to translate three inputs into a clear set of recommendations for a rationalized and optimized portfolio:

As an input, the EA team provides a consolidated list of all projects that are underway, as well as details of the benefits that are documented in the project proposals. We commonly find that most of the benefits are described in terms that the IT staff will understand or that have loose ties to the strategic direction of the business. Budgets are constantly undergoing scrutiny, and stakeholders need to know that the budget is used appropriately. This process will connect the IT project portfolio to business capabilities and, in turn, business strategy. By clearly defining measurable benefits that can be aligned with an impact on the business, we will have a solid foundation for later measuring the realized value across the entire IT portfolio.

The Benefits Dependency Network

The Benefits Dependency Network (BDN) is a model—usually created on one page—that links IT projects to the business activities that are being changed and the reasons behind those changes. It is one part of the benefits-management approach that was developed by the Information Systems Research Centre at Cranfield University School of Management in the U.K. Over the last 10 years, we have extended the original model to include a time line.

Figure 2 illustrates the key components of the BDN. In this model, we start with an understanding of **why** the business must change via the drivers, objectives, and benefits. We can then connect the benefits to the business processes and capabilities to

Figure 1: Overview of the process

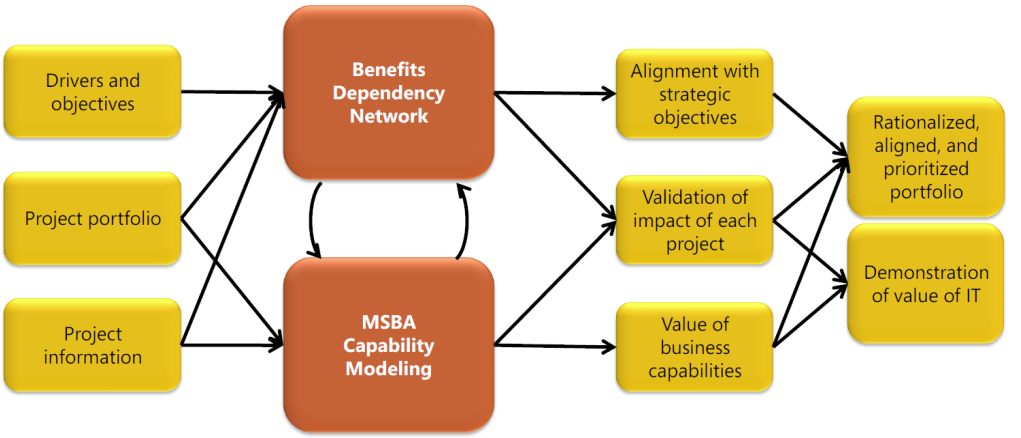
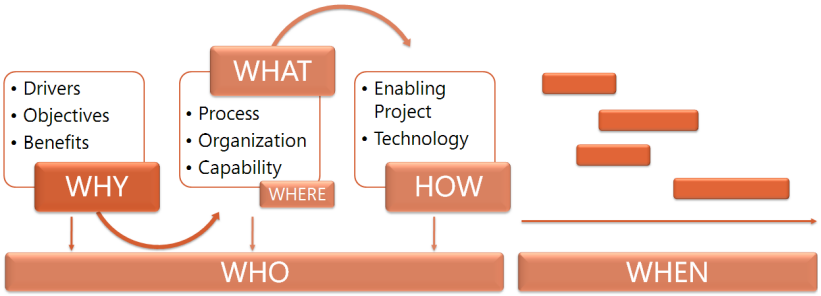


Figure 2: Components of a BDN



understand **what** will have to change to realize the benefits. We also gain a better understanding of **where** the changes will happen in the organization units and specific locations. Finally, we can determine **how** to make the changes via projects that will change business activities and the technologies that will support them. In all of this, we can also identify **who** are the stakeholders. Who will be responsible for demonstrating the benefits that are achieved; who owns the business activities that are to be changed; and who will run the projects?

By prioritizing the objectives and benefits, and then incorporating requirements in the order in which to change business activities, resource constraints on projects, and any technology dependencies, we can sequence the projects and show **when** they will be implemented on a time line.

Firstly, the value in a BDN is in having a clear definition of the why, what, and how. Secondly, the values are in the connections that are drawn across the diagram to show all of the relationships.

Typically, a BDN is started in a workshop and completed over a number of weeks via interviews and iteration. Often, we use the BDN at the start of major programs to define the projects, but we use it also to define the components of large projects and map the portfolio of projects in an IT organization.

Creating the Enterprise Benefit Map

Like many customers, Contoso Bank has hundreds of IT projects. Starting with the projects that have the largest business or IT contribution, we identify the business functions that will be affected, along with the benefits that are expected. By using the Contoso Bank business and IT strategies, we capture the top-level organizational drivers and objectives.

This information is laid out as Figure 3 shows—with projects on the left, business functions and benefits in the middle, and organizational drivers and objectives on the right. By working with the project stakeholders and subject-matter experts from both IT and the business, we connect the items in each column: aligning projects with the business function, the business function with the benefits, and so on.

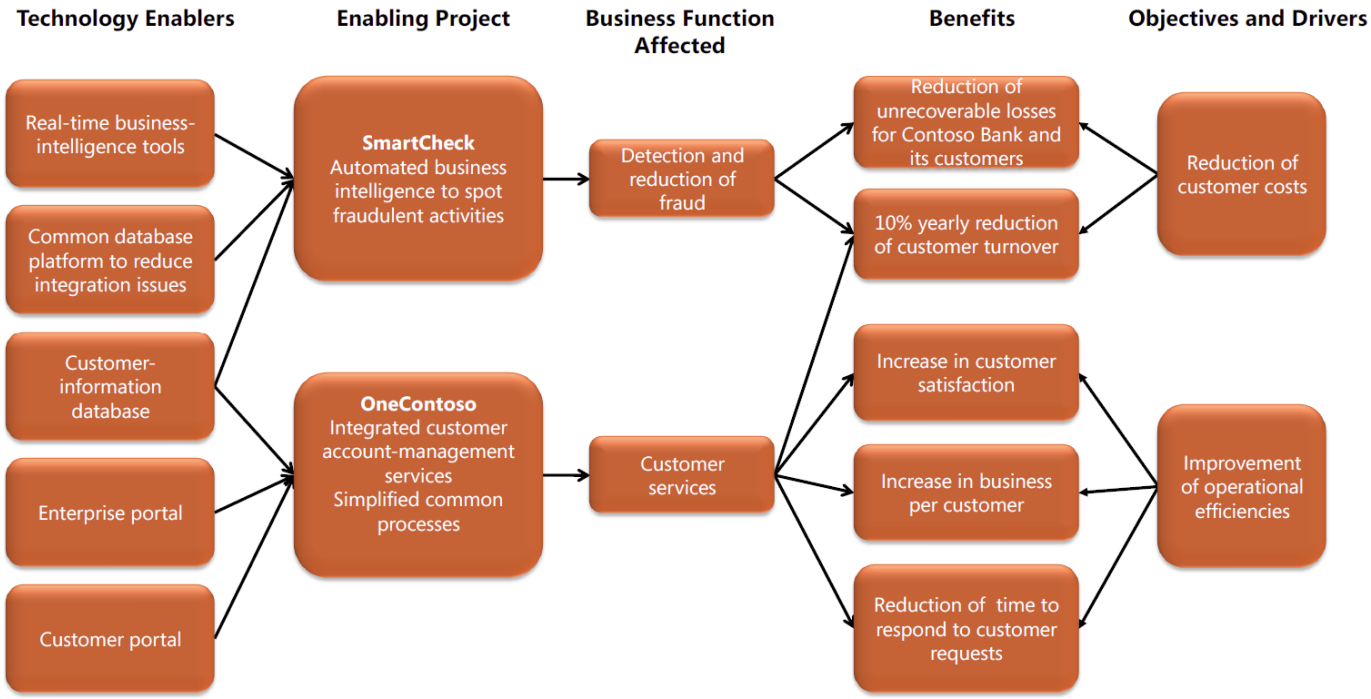
This simple view provides us with the ability to quickly validate the portfolio and identify if there are projects that do not align with any of the objectives. For these projects, either we need more information or we have identified the first round of projects for possible rationalization.

We repeat the process again for the next tier of projects. Although the model grows in complexity, the process gathers pace as we iterate through the project portfolio.

Creating the Enterprise Capability Model

During creation of the BDN, we described the business function that was affected by the proposed change in terms that are used by the project definition. We must use a common framework for the business functions in order to compare projects and identify possible duplication or opportunities for optimization across projects. This framework is defined by MSBA.

Figure 3: Extract from the BDN of Contoso Bank



Cost-Cutting and Innovation: Not Mutually Exclusive

by Ric Merrifield

It is more accurate to call it a false dichotomy, but when people talk about cost-cutting, they often talk about it almost as if it were the opposite of innovation, that they were mutually exclusive. This is not so.

It is yesterday's news (literally) that you can use technology, even cloud computing, to create an innovative business model. Yesterday, Microsoft announces that it would stop making the Encarta digital encyclopedia product, and many people speculate that the success of Wikipedia, a cloud-based encyclopedia, was a big part of the demise of Encarta.

We have seen this many times before: where Netflix used the cloud to help create an innovative way to rent videos, Amazon used the cloud to sell books, and so on. Although people understand that these innovations cut costs out of the business model, they seem to see it almost as a footnote to these innovations.

Recently, someone asked what the biggest mistake is that people are making in thinking about cost-cutting in these turbulent times. My answer was that people are underestimating how much cost and waste have crept into their operating models in the past 20-odd years of growth. I have evidence that companies should be able to cut 20 percent to 40 percent from their operating costs just from eliminating duplication and waste. But I will go a step farther and say that one of the best ways to achieve this level of cost-cutting is innovation; and, while cloud-computing opportunities absolutely open a world of new possibilities for innovation in cost-cutting, they are not the only way.

Look at Tata Motors and its new $2,000 Nano car. The focus of my work is outcomes—"what" you are doing—and "what" Tata Motors did was come to market with a car that is 90 percent less expensive than other car in its class, such as the Smart Car from Daimler-Chrysler. "How" they did that is their business, but the point is that they did it, which sends a powerful message to the rest of the world—not just the automotive world—about the magnitude of cost-cutting opportunities.

One piece of advice in terms of "how" to see where to cut costs innovatively: Instead of asking, "Can we do this differently?", I suggest that you ask, "Why can't we do X?" Whether X means outsourcing work to your customer (as banks did with the ATM), using a cloud service (such as airport check-in), getting rid of an entire department (ING DIRECT eliminated paper checks, so that, with only online checks, people cannot bounce checks, so that there is no need for a department that handles bounced checks), or something else, my experience is that you will find all kinds of innovative ways to cut costs out of "what" you do if you just ask, "Why can't we?"

When you rethink your cost-cutting approach and shed your doubts, I think that you will find innovation and cost-cutting to be an even better combination than most of us have found peanut butter and chocolate to be.

Ric Merrifield (ricm@microsoft.com) is a Principal Architect at Microsoft. He has just published *Rethink: A Business Manifesto for Cutting Costs and Boosting Innovation*.

We always begin development of capability maps from a predefined template. For this scenario, we begin with the retail-banking capability template. By using this model as a starting point, we validate the top two levels and adjust the naming of capabilities to reflect the terminology of the bank and create the framework into which to map projects.

With the EAs and the project, leads we work through the portfolio of projects. For each project, we validate the benefits and objectives that are identified in the workshop, understand the technical solutions that are required for the project, and identify the business capabilities that will be affected. During this process, we take the business functions and organizational groups that are identified in the workshop and perform a mapping to the capabilities model.

What Is a Capability?

A *capability* is an abstract view of a business function. A capability model provides a stable view of the organization, even if the implementation changes. Capabilities are hierarchical; each might contain child capabilities that provide greater detail and specificity. Capabilities are measurable and have properties that might include current performance, business value, consumers, owners, maturity, and key performance indicators. The properties will also detail the people, processes, and technology that are used to instantiate the capability.

Capabilities also can have connectors that are used to record relationships. The connectors include details about the type of relationship and service-level expectations.

While processes are modified, technology changes, and people reorganize, a business capability remains constant.

Microsoft Services Business Architecture (MSBA)

MSBA is a blueprint of a business that spans the organization-mapping information at a high level down to a detailed granular level and expresses each discreet function or business capability. MSBA defines business architecture as what the business does in the form of business capability. This is in contrast to how the business implements the capability in the form of people, processes, and technology.

MSBA is a view of the operating model of a business or organization, based on its business capabilities, and is not limited to the four walls of the legal entity of the organization. Use of capabilities provides an abstract view of the organization that remains stable, even as processes, technologies, and people change. Even the changes that are associated with outsourcing do not change the capability map.

The capability map acts as a common language that can be used between both the business and IT. It also provides a solid foundation for discovering and describing services, when defining a service-oriented architecture (SOA). Having studied businesses across many industries, Microsoft found that businesses within most industries exhibit five core operational capabilities, as shown in the center of Figure 4.

These MSBA capabilities are the root capabilities that are built on, when developing the business architecture for the operating model of an organization. The capabilities that appear outside of the diagram are known as the environmental constituents. These represent entities that are outside of the direct control of the organization and generally are provisioned by other organizations. In most cases, these entities are not detailed and include only the capabilities that are needed. It is common to see regulatory requirements called out in this area, with relationships to capabilities that are affected by the requirement.

Note: Public and third-sector models also exist.

Figure 4: MSBA capabilities

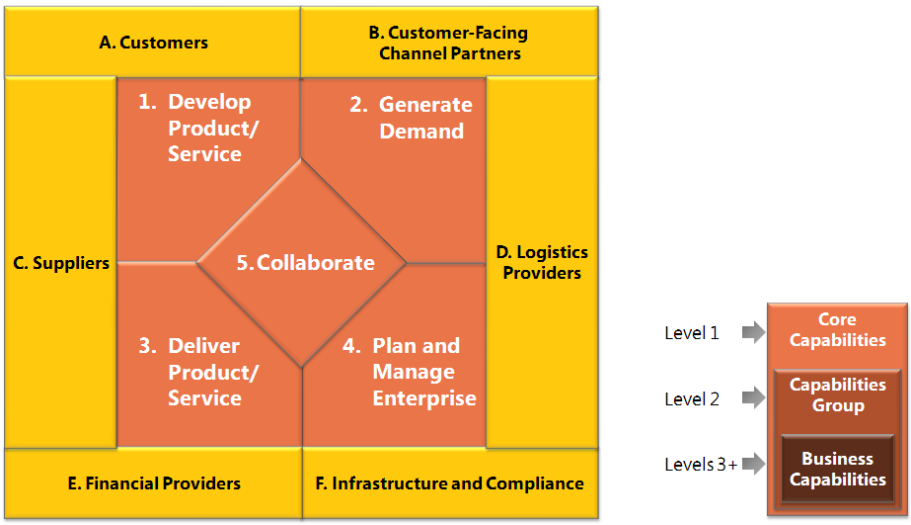


Figure 4 identifies the core capabilities. Below this, we have a capabilities group at Level 2, and then business capabilities in Level 3 and below. This is an important distinction. While Levels 1 and 2 provide a framework to understand the business, it is only when we get to Levels 3 and below that we see a level in which specific project changes can be discussed and scoped sensibly.

The purpose of our capability model is to help understand the value of the IT portfolio. This means keeping the capability modeling at a high enough level that we do not get lost in the details and can retain the organizational overview.

In most cases, it will be necessary only to map the capabilities of a project to Level 3. In some cases, however, it might be necessary to map capabilities to Level 4. This is done when further clarity is required or to highlight specifics of one project when contrasted with another.

Bringing Together the BDN and Capability Model

It is time to bring together the two models. The original BDN contains the business functions from the initial project analysis. The capability model contains the capability definitions that are aligned with the wording that is used within the organization. We use this to update the BDN, so that it incorporates the new capability definitions that are replacing the business functions.

The example in Figure 2 from the initial workshop identified the SmartCheck project as affecting the "detect and reduce fraud" business activity. As we drilled into the capability model, we changed this to relate directly to a Level 4 business capability: "Fraud Detection." In another example, the "OneContoso" project has the goal of providing an integrated customer account–management service. In this project, we identified many different capabilities at Level 4 that would be affected. These capabilities roll up into the Level 3 "Customer Services" capability. The Level 4 capabilities are important to the "OneContoso" project; for our purposes, however, we can focus on the higher Level 3 "Customer Services" capability.

Measuring the Capabilities

It is important to understand why we are going to measure the capabilities that we have identified. In most cases, we measure a set of properties for capabilities and plot these on the capability model to create a graphical representation that is called a heat map, in which the fill and border colors for capabilities represent the value of a property.

There are a few properties that are often included for project-scoping work:

- **Business value:** The strategic value of a capability or the degree to which it affects organizational performance. This is the measure to which a capability is a market differentiator or a competitive requirement.
- **Maturity:** The level of consistency and predictability with which the capability is performed. Is there a well-defined process in place that is monitored and measured?
- **Performance:** The capability that meets service-level expectations.

There might be additional properties that are important to each project; however, our goal by capturing the same information for each capability is to contrast the properties and identify where to focus. When reviewing a portfolio of projects, "level of IT investment" is a primary property. We use heat maps to provide comparisons quickly and enable us to focus on areas of interest.

The Contoso Bank Heat Map

A *heat map* is a graphical view of the entity map in which property values are used to define entity shape characteristics. For example, business value will be represented by the border color, and performance will be represented by the fill color of the shape.

Figure 5 show the Contoso capability map in which the fill color of the shape is determined by business value and the border is determined by the level of IT investment.

The two properties that are used to understand the portfolio are the business value that is to be achieved by improving the capability and the level of IT investment in the capability. A five-point scale is used to help quantify value and spend. By working with IT and business stakeholders, we capture their understanding of the value to be realized in each capability as a result of all of the IT work that is being done in the capability. A capability might be affected by more than one project; so, here we are interested in the cumulative impact. By working with the IT stakeholders, we assess the scale of IT investment in a capability. Just as there might be more than one project affecting a capability, we might also find many capabilities being affected by one project; so, we are asking for an assessment of how the IT investment is distributed.

One of the more important pieces in the assessment is the relative impact of each project on a capability, when more than one project is mapped to it; so, we identify projects that have overlaps in scope and could be rationalized to reduce costs.

Figure 5 shows a simplified heat map—down to Level 2—that was created for our Contoso Bank example. In reality, this would drill down to Level 3 and show only the capabilities that are being affected by the project portfolio.

Mapping Project Technologies to Projects in the BDN

We now return to the BDN and the portfolio of IT projects. Every IT department typically has many projects for the implementation of

shared infrastructure, technology upgrades, or projects. Some of the technology projects will be uniquely associated with a project that enables a business change and, as such, will have been identified as we evaluated each of these projects earlier. We now bring all of them into the enterprise-level BDN and show the linkage between the projects that are for implementing technology and the projects that are for effecting change in the business.

We leave the technology projects until after the business projects, because our focus is on the value of IT to the business, and the business view is where we must start to get that structure right.

At this point, we find technology upgrades and deployments that do not obviously connect to business-change projects. Sometimes, these projects should just be stopped. In other cases, we find that the technology is required just to run the business. For example, it could be a network upgrade that is necessary to accommodate normal traffic growth.

We use capability modeling to replace the processes and organization names that initially were used for the business changes in the BDN, because capabilities persist, while their implementation in people, process, and technology will change over time by the actions of the projects in our portfolio. If it aids in understanding, we can also create alternative heat maps that use annotations to show which capabilities will be affected by specific enabling technologies.

Using the Capability Heat Map and Benefits Model to Prioritize the Portfolio

With a completed high-level BDN and heat map to demonstrate the potential business value that is being delivered by IT, we now can start a review of the portfolio of IT projects. For a project to have been in the initial portfolio, it would have to have cleared the basic ROI hurdles of the bank. With the information that we have now, we will:

- Recommend cancelation of projects that have no clear alignment with business objectives and benefits.
- Restructure projects that have overlapping impact on the same capabilities, so as to reduce duplication of effort.
- Rationalize IT projects to leverage common technologies across the business projects.

Finally, we can prioritize the remaining project portfolio, based on the highest priority objectives, strongest business benefits, and highest business-value capabilities. The Contoso Bank team has to determine which of the objectives, benefits, and capabilities it should use as the drivers of prioritization. However, the analysis that we have done makes this information clear. The information and approach that we have applied allows us also to compare projects that have strong financial and nonfinancial benefits.

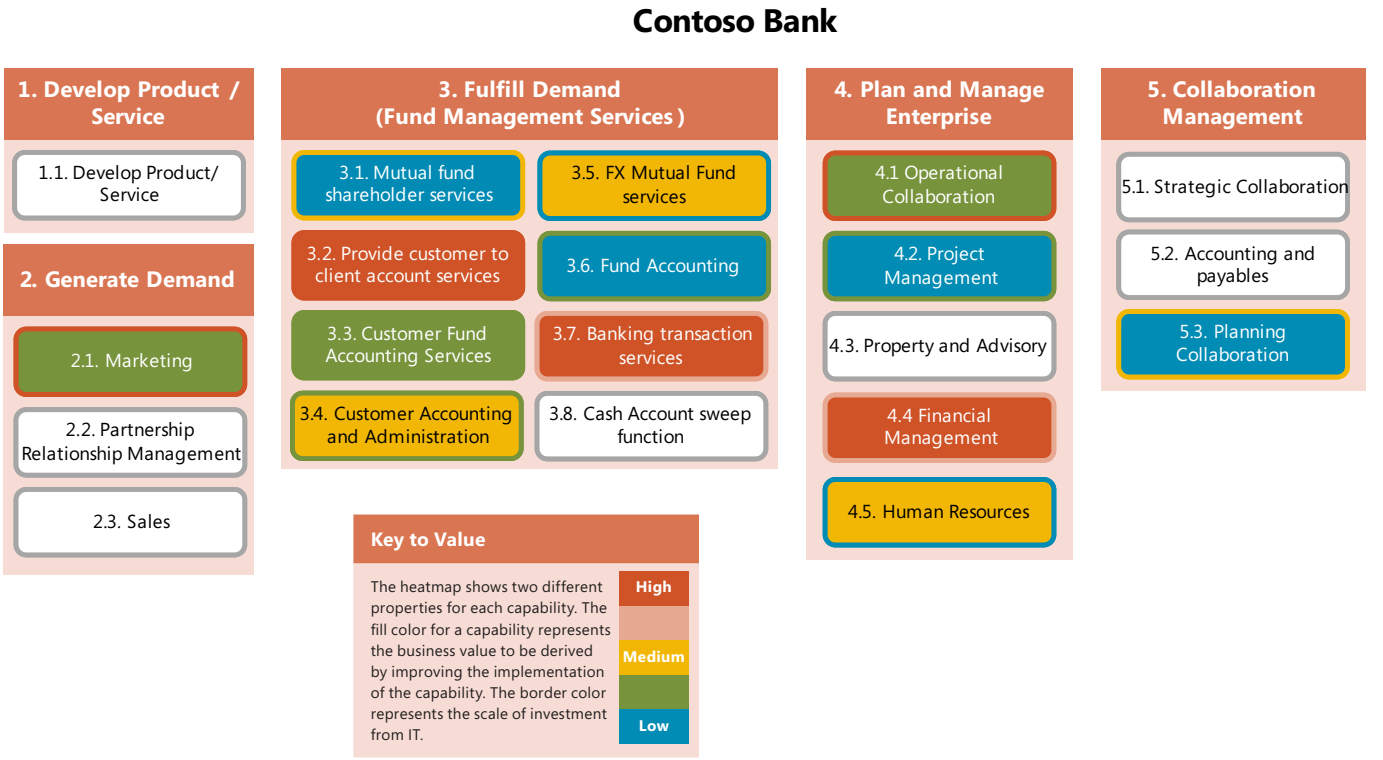
The result of this prioritization is an enterprise-wide reduction in total cost of ownership (TCO) for IT and earlier realization of high-value business benefits that will lead to an improved ROI from the early realization of benefits. To achieve this, we also might have to review the resourcing and re-plan the implementation of the portfolio.

Resourcing the Newly Sorted Portfolio and Putting On a Time Line

With the projects listed in order of priority and allocated resources, we now produce a standard IT program plan and road map. Different views of this road map now can be created. One view shows the benefits that each project delivers, and the other the capabilities that are affected.

Just as a heat map can show the current value of properties, so can it be used also to demonstrate the value that is to be expected at

Figure 5: The simplified Contoso Bank capability heat map



some point in the future. Contoso Bank operates a quarterly reporting process to review the performance of every department. By using the new project time line, we work with the architects to create a heat map for each of the next eight quarters that shows the anticipated cumulative benefits to each capability in each quarter. The capability model itself does not change between quarters, as the capability abstractions do not change; it is their implementation that changes. So, the CIO now has a series of views to show a time sequence for how IT will deliver value and be used to track delivery against plan.

The BDN and capability map are resources that architects can update as projects progress and strategies evolve or are replaced. If the information is updated quarterly, it can be compared against the projected information, and adjustments can be made to future projects to deal with the gaps, as required.

Conclusion

A powerful and positive side effect of the use of a BDN and an MSBA capability model is the ability to tell a very clear and effective story about the IT portfolio. By using the capability heat map, the CIO has a powerful one-page model to focus attention on the impact of IT investments and to be included with updated information on the progress of projects—delivering the benefits against the plan. The information and analysis serves as the foundation for identification and prioritization of future projects.

The BDN at an enterprise-level provides an overview of the specific benefits from IT investments and a route into the BDNs for each specific project. By aligning each project with an enterprise road map, the realization of specific benefits can be noted also on the road map and tracked over time. For the first time, Contoso Bank not only can associate every IT project with the clear delivery of value, but also it can create a scorecard to demonstrate the realization of that benefit.

References

Peppard, Professor Joe, Professor John Ward, and Professor Elizabeth Daniel. "Managing the Realization of Business Benefits from IT Investments." *MIS Quarterly Executive*. Vol. 6, No. 1, 2007, pp. 1–11.

Merrifield, Ric. *Rethink: A Business Manifesto for Cutting Costs and Boosting Innovation*. Upper Saddle River, NJ: FT Press, 2009.

Merrifield, Ric, Jack Calhoun, and Dennis Stevens. "The Next Revolution in Productivity." *Harvard Business Review*. June 2008.

Ward, John, and Joe Peppard. *Strategic Planning for Information Systems*. Third ed. Chichester: Wiley, 2008.

About the Authors

Martin Sykes (msykes@microsoft.com) is a Senior Program Manager in the Microsoft Consulting Service IT Architecture and Planning Service Line. His focus is on developing the methods and offerings for the Microsoft approach to IT Architecture & Planning.

Brad Clayton (bradcla@microsoft.com) is a Senior Program Manager in the Microsoft Consulting Service IT Architecture and Planning Service Line. His focus is on the evolution of the Advisory Services Program.

Follow up on this topic

- Microsoft Services

MSBA and SIP: A Combined Approach

by Paulo Rocha and Roger Sessions

Microsoft Services Business Architecture (MSBA) is a powerful methodology for uncovering initiatives that can improve the ability of an organization to conduct business. As powerful as it is, however, MSBA leaves unanswered an important question: How do we know that the hierarchical MSBA-generated capability map is optimal?

At every decision point in the creation of the MSBA capability hierarchy, there are numerous outcomes. Different analysts will make different decisions, depending on the skill, experience, biases, and expectations of that person. By the time that the analysis is complete, there can be many possible capability maps.

Out of these many hierarchical maps, only one is optimal—meaning that it will deliver the greatest return on the least IT investment. Every other map will either cost more than the optimal map, deliver less value, or both. In turbulent times, every dollar counts.

So, how does one know that one particular MSBA solution is optimal? Even better, how one can drive MSBA capability mapping, so that only the optimal solution is delivered in the first place? Like most architectural methodologies, MSBA offers no solution to this problem.

Fortunately, another methodology does have a solution. That methodology is called Simple Iterative Partitions (SIP). SIP is highly complementary with MSBA. Whereas MSBA drives capability maps, SIP analyzes capability maps for correctness. Whereas MSBA identifies decision points in a process, SIP identifies which of those many decisions is the best possible decision.

SIP is about simplicity. It is based on logic, set theory, and the mathematics of equivalence relations. Every decision point is pragmatically tested to determine the decision that generates the simplest overall outcome.

MSBA and SIP are an ideal partnership. MSBA tells you what to change and—combined with Benefits Dependency Networks—why those changes must be made, and SIP guides you to the simplest possible way to make those changes. In turbulent times, it is not enough to identify areas for change. We must make those changes as cost-effectively as possible.

For more information on SIP and its relationship to MSBA, visit SIP central.

Paulo Rocha (paulo.rocha@kiwiocas.net) is an Architect with Microsoft New Zealand.

Roger Sessions (roger@objectwatch.com) is CTO of ObjectWatch.

Practical Architecture in Impractical Times

by Matt Valentine

Summary

This article focuses on pragmatic approaches for effectively navigating this new environment and helping architects to become more proficient, capable, and skilled.

Navigating the Storm

Smooth seas do not make skillful sailors.
ENGLISH PROVERB

For architects, the current economic condition offers an unparalleled opportunity for growth and innovation. The use of sound architectural practices and a strategy focused on efficiency gains and delivering near-term revenue are effective ways to ensure that the contributions of architects are indispensable.

In today's turbulent economic climate, the immediate reaction might be to cut costs across the board—especially in IT. However, in reality, IT today is so interconnected with operations that such reductions would be counterproductive. In fact, according to McKinsey and Company, a dollar that is invested in IT can have 10 times the impact of traditional cost-reduction efforts.[1]

Your business customers, partners, suppliers, and employees are going to continue to demand new capabilities—with likely far fewer resources. It is exactly these constraints that offer architects an unparalleled opportunity for growth and innovation.

The fact is that in any economy, innovation is the foundation for success. While there is an overall slowdown in spending, organizations will still need to make smart investments in innovation to help them grow the business. The following are two examples:

- RCA investments in technology during the Great Depression established the company as the leader in TV technology for the next 25 years.
- During the Great Depression, Sears shifted its business model from catalogs to retail stores and became the largest retailer in the United States for the next 50 years.

Architects are generally accustomed to driving alignment between IT and business requirements. While much architecture work is strategic—focused on achieving a specific goal or business advantage—in today's economic climate, now is not the time to embark on risky and expensive business-reengineering efforts. In fact, the primary finding from the Gartner Business Executive Survey, 2009, is that "most leadership teams will reduce their focus on longer-term issues."[2] As a result, the key to surviving the economic downturn is to intensify opportunities to produce near-term revenue and efficiency gains.

Effective architecture seeks to align business and technical requirements by shared understanding of system design by balancing **business**, **technology (or system)**, and **user** goals.[3] Before detailing some practical projects, let us quickly review some valuable architectural techniques and practices that are particularly suited to current economic conditions.

Business Goals

Maintain Focus on the Business

If you have a job without aggravation, you don't have a job.
MALCOLM FORBES

In the current economic environment, the ability of an architect to drive alignment between IT and business has never been more critical. While much has been written on this topic, here are a few things to remember:

- There is no better way to look forward than to stay focused on customers, both internal and external.
- The unfortunate reality is that few business leaders understand the value that IT delivers; therefore, it critical to focus on projects that contribute meaningfully to the success of the enterprise.
- The value of an IT department is directly proportional to the usefulness of the information that it produces.[4]
- Using the Pareto principle as a guide, identify and address the 20 percent of the causes that result in 80 percent of the effects: revenue, bugs, downtime, business processes, and so on.
- Keep in mind the advice of Dale Carnegie: "When dealing with people, remember that you are not dealing with creatures of logic, but of emotion."
- The enemy of good is perfect—also known as Cosby's Law: "I don't know the key to success, but the key to failure is to try to please everyone."

Take an Agile Approach

Big jobs usually go to the men who prove their ability to outgrow small ones.
RALPH WALDO EMERSON

By applying an agile development methodology to all projects—in which work is performed in small, rapid increments with minimal planning, and is aligned closely with customer needs and company goals—change can be managed more easily. The result will be finite, manageable projects that have a clear connection to the business.

"The agile community taught us the idea of test-driven development, where you define the test, write enough code

for the test to run green, and then simplify the code through refactoring—making sure that the tests continue to run green. This thinking can very much apply to IT in the large, where you adopt a Lean mindset and focus on quickly delivering projects that increase flow—value for the customer, then simplifying and reducing waste through consolidation."

SAM GUCKENHEIMER, GROUP PRODUCT PLANNER, MICROSOFT VISUAL STUDIO TEAM SYSTEM AND AUTHOR OF SOFTWARE ENGINEERING WITH VISUAL STUDIO TEAM SYSTEM

Use Well-Known Patterns

The aim of science is always to reduce complexity to simplicity.

WILLIAM JAMES

The ability to identify recurring patterns quickly and take action—whether in sports, business or architecture—has long been a hallmark of effective leaders. From an architectural perspective, patterns provide a reusable solution to commonly occurring problems—whether business or technology:

- Industry patterns, including customer service in retail banking, mass-assembly manufacturing systems, and emergency response services in government
- Technology patterns, including data warehousing, service-oriented architecture, and e-commerce Web sites
- Collaboration patterns, including running effective meetings, brainstorming, and architectural-design sessions

Now is not the time to build a better mousetrap; therefore, leverage resources such as Patterns & Practices[5], Microsoft Operations Framework[6], Solution Accelerators[7], and the MSDN Architecture Center[8] to accelerate solution delivery, drive cross-team alignment, increase predictability, mitigate risk, and increase productivity that results in better alignment with business.

Technology Goals

Any sufficiently advanced technology is indistinguishable from magic.

ARTHUR C. CLARKE

Businesses are complex systems of people, processes, and technology, and this complexity is not without cost. In fact, according to a recent survey by Bain & Company of executives at 960 companies from around the world, nearly 70 percent reported that complexity was driving up costs and hindering growth.[9]

Systems thinking[10] and Dr. Goldratt's Theory of Constraints[11] have taught us that system performance depends on how parts interact, and not on how they work separately. Therefore, taking a holistic approach and minimizing the friction between processes can be more effective than addressing each subsystem individually.

"We find that it is simplest to focus on just a few vendors and, ideally, one if they can provide the best holistic integrated environment. ***Our studies show that this reduces the time for service implementation by 50 percent and [provides] a reliability/productivity boost overall.****"*

STEPHEN IBARAKI, FCIPS, I.S.P., MVP, DF/NPA, CNP

As a result, organizations are converging what were once disparate, standalone applications and systems into integrated, unified application platforms in order to better to support their business applications. While no strict definition of "application platform" exists, these software systems essentially consist of an integrated and interoperable set of IT infrastructure, business applications, and tools for application development.

Infrastructure

Core infrastructure provides fundamental, foundational capabilities for supporting IT operations—for example, identity and access management, desktop-device and server management, security and networking, and data protection and recovery.

In general, these "utility" capabilities do not provide a competitive advantage to the business. Therefore, the focus should be on driving new efficiencies to reduce costs.

Applications

Business applications are generally how users experience IT. Applications provide people with access to data and processes that they need to do their jobs—making decisions, understanding their customers, managing operations, and so on.

Whether business applications are packaged, custom-developed, run on premises, or hosted in external data centers, they can have a profound impact on the business. As a result, strategic investments in business application can help the business differentiate effectively and result in competitive advantages.

Tools

Use of application-development tools and frameworks to build highly specialized business applications can improve the ability to outperform competitors. A homegrown data-analysis engine that identifies high-value segments and audiences, or an automated business process that enables an organization to deliver products and services more quickly, is an example of how custom application development can provide a competitive advantage.

Organizations need powerful, flexible tools and skills that support a wide range of application types and architectural styles. The greater the flexibility of the development platform, the greater the benefit to the business.

While optimizing disparate applications and subsystems—such as tuning a database, profiling the performance of an application, or increasing load on a Web site—can result in local efficiencies, the biggest benefits come from taking a holistic approach to improving how infrastructure, applications, and tools work together to support critical business applications.

User Goals

Users ultimately decide the success and failure of any IT project. Therefore, a pragmatic approach must stay focused on users—both internal and external.

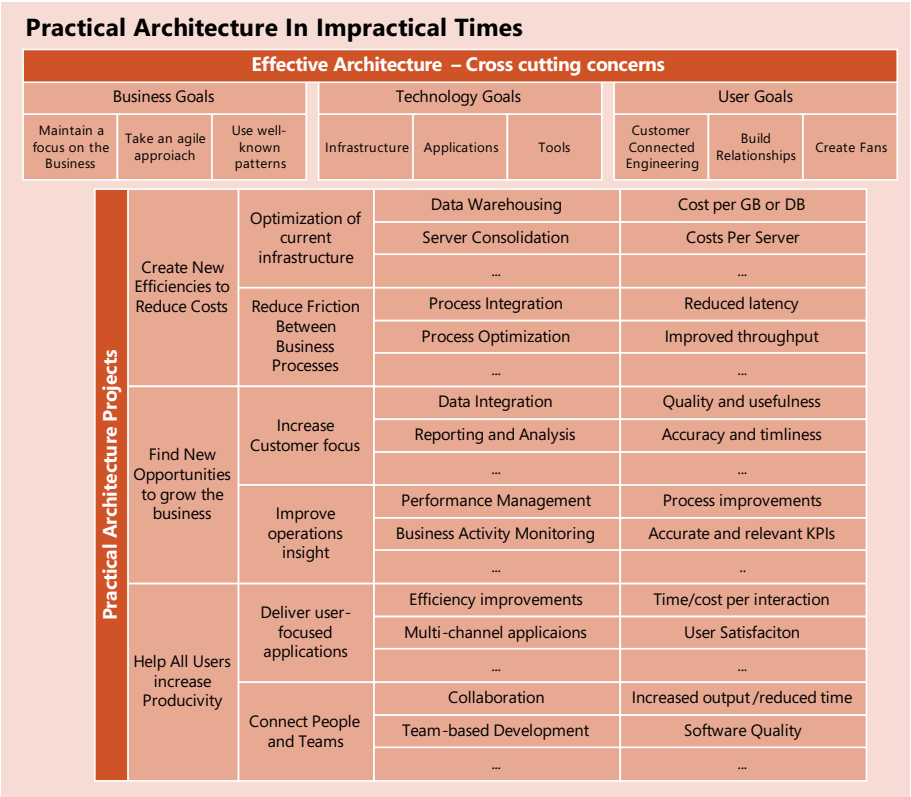
Customer-Connected Engineering

Many projects fail to deliver on promises and meet the requirements of users. According to a well-publicized report from The Standish Group, only one out of three projects is a success, and 66 percent of projects fail or take longer than expected.[12]

IT and custom application-development projects can be complex and, thus, risky endeavors. To mitigate this risk, it is important for projects to leverage customer-connected engineering, scenario-driven development, and agile development methodologies that rely on frequent check-ins with the business to ensure that IT projects do not end in spectacular failures.

"I've talked to many companies that have ***implemented stunningly beautiful SOA infrastructures*** *that support managed communications using virtualized proxies and*

Figure 1: Pragmatic opportunities to apply crosscutting techniques and practices



dynamic bindings. They've deployed the best technology the industry has to offer, including registries, repositories, SOA management, XML gateways, and even the occasional ESB. Many have set up knowledge bases, best practices, guidance frameworks, and governance processes. And yet these SOA initiatives invariably stall out. ***The techies just can't sell SOA to the business. They have yet to demonstrate how all this infrastructure yields any business value.***"

ANNE THOMAS MAINES, VICE PRESIDENT AND RESEARCH DIRECTOR, BURTON GROUP[13]

Build Relationships

The secret of business is to know something that nobody else knows.

ARISTOTLE ONASSIS

User scenarios and use cases help architects understand the necessary requirements, but nothing beats building deep relationships and understanding of your customers—both internal and external. In addition to technical experience, highly effective architects have a deep understanding of the business and industry issues and, given their ability to relate to both groups, are actively sought out for their expertise.

Create Fans

If we don't take care of our customers, someone else will.

ANONYMOUS

It is not enough to deliver on time, on budget, and to specification. Users increasingly want more capabilities in their business applications. They want more flexibility, easier access, collaboration, and better productivity from their applications. By delivering business applications that exceed expectations, you not only create fans for your work, but also improve the ability of people to advance the business.

Practical Architecture

In the previous section, we reviewed valuable crosscutting techniques and practices that can help architects navigate the current economic conditions. Now, let us look at some pragmatic opportunities to apply these best practices, including:

- Creating new efficiencies to reduce costs.
- Finding new opportunities to grow the business.
- Helping all users increase productivity.

Creating New Efficiencies to Reduce Costs

Infrastructure Optimization

Driving lower expenses through continuous IT consolidation and optimization is more important than ever. Capability maturity models and enterprise frameworks such as the Zachman Framework[14], The Open Group Architecture Framework (TOGAF)[15], or Microsoft's own Core Infrastructure Optimization[16] model can be effective architectural approaches for driving down IT costs.

Two major opportunities for driving down costs are data and server management:

- With automated server virtualization, deployment, and security processes of data-management servers, IT organizations can lower costs and increase per-server FTE efficiency. A Microsoft-sponsored study shows that organizations that operated at the highest level of performance incurred 63 percent lower per-server operations costs and managed approximately three times more servers per administrator than those that operated at basic levels.[17]
- Server virtualization enables organizations to reduce their hardware footprint, save on the cost of utilities, and (where systems management is used) reduce IT labor costs. These savings are reflected in an annual $2,300 per-server difference in IT labor costs in organizations that adopt the practice, compared to those that do not.[17]

In addition to reducing costs, an optimized IT infrastructure can enable quicker response to changing business needs:

"Compared to multiple, single-instance deployments of Microsoft Dynamics CRM, using the multitenant capability of Microsoft Dynamics CRM offers ***lower startup costs****. Previous deployment times ranged from weeks to months, depending on internal requirements. Having a proven production environment eliminated this time, and now we're able to* ***spin-up instances in as little as five minutes****."*

MARK BACIAK, PRINCIPAL ARCHITECT, MICROSOFT CORPORATION[18]

With the emergence of cloud technologies that promise capacity on demand, the opportunity exists to shift away from expensive hardware capital expenditures (CAPEX) to operating expenses (OPEX) that are based on elastic capacity needs.

Reducing Friction Between Business Processes

> *Nothing is particularly hard if you divide it into small jobs.*
>
> HENRY FORD

Effective architecture can help drive down operational costs through improved business-process flow, automation, management, and optimization.

Service-oriented architecture (SOA) is a popular strategy for designing solutions that might require integration across operating systems, applications, and databases. A service-oriented approach can unlock the value of existing systems and improve cost efficiency and overall business agility. The key to success with SOA is to focus on practical applications of service-orientation instead of expensive, mega-scale business-process reengineering efforts.

To support customer support requests for the growing Xbox, Xbox360, and Zune products, architects at the Microsoft Entertainment and Devices (E&D) division have focused on practical approaches for reusing existing systems to minimize the need to create expensive new applications. By taking an agile, service-oriented approach, E&D has been able to increase scale and data consistency quickly and improve process transparency—resulting in improved process flow, customer satisfaction, and customer service–agent productivity.

> *"The economic condition has intensified our existing methodologies for driving business value. Using a practical approach to architecture and a focus on continuous improvement, we're using value-stream mapping—a Lean approach—to improve flow, eliminate waste, and reduce latency.*

> *"By having a service-oriented integration layer, we were able to streamline our outsourced repair logistics processes, which reduced costs by as much as $3M a month; and, by creating new services based on BizTalk rules that capture entitlements and eligibility information, we're seeing a ton of agility and adaptability value—**about a half-million a month in savings**."*
>
> MIKE HATCH, PRINCIPAL SOLUTION ARCHITECT, MICROSOFT E&D DIVISION

In late 2004, EDS, an HP Company, began a project to help airlines extend the life and value of existing IT systems, take more advantage of outsourcing options, and improve competitive advantage. Using a service-oriented approach that is based on Microsoft technologies, EDS Airline SOA is a flexible and efficient transaction-processing platform that provides middleware, messaging, and security capabilities for more than 20 applications that provide key airline services and functionality—including reservations, flight planning, and flight operations.

By using a service-oriented approach, EDS extends the life of existing IT infrastructure while proactively retiring outdated systems. EDS also pulls functionality from TPF and other legacy systems and migrates those applications to a Microsoft platform that is easier and less expensive to maintain. The results of this project include savings of between 10 and 25 percent on new services, the ability to execute a legacy modernization strategy more rapidly, and a nearly 30 percent lower total cost of ownership versus mainframe.[19]

> *"To help reduce costs and streamline airline operations, we took a pragmatic approach to SOA. We focused on systematically decoupling and then shifting application workloads from expensive, inflexible legacy systems to more agile and cost-efficient systems. Today, the system in production provides high reliability and performance—over 1.2 billion transactions per year—at lower cost. We expect to be able to make changes to the reservation system in a few weeks, where before it took several months."*
>
> MIKE MELTON, TRANSPORTATION GROUP CHIEF TECHNOLOGIST, EDS, AN HP COMPANY

From a data perspective, by reducing the number of required steps from 19 to 9, automating 70 percent of its business-intelligence requests, and providing Microsoft Office Excel as the user interface, Lloyds TSB—a leading financial-services group—was able to reduce the time for critical business reports from months to minutes, which provided significant efficiency gains for the business.[20]

Finding New Opportunities to Grow the Business

Increase Customer Focus

> *You can close more business in two months by becoming interested in other people than you can in two years by trying to get people interested in you.*
>
> DALE CARNEGIE

Information is at the core of a business's ability to make effective decisions. Architecture initiatives that increase the quality, timeliness, and usefulness of information have a direct correlation to increased revenue. A practical approach is to focus on anticipating the needs of your customers—both internal and external—because they are the key to growth. Typical customer queries include the following:

- Which customers are the most profitable? Least?
- Which customers have the highest potential for growth? Least?
- Which customers are at risk for leaving for the competition?
- What are the primary customer-satisfaction issues?
- Which types of customers do we wish we had more of?

While disconnected data can provide usable customer insights, the combining of data from different systems and business units is likely to produce a more comprehensive and valuable customer view.

Lloyds TSB is a great example. By providing their business analysts with integrated access to disparate data stores, including their 40 TB Teradata Warehouse, Lloyds TSB increased their understanding of the spending behaviors of customers in order to maximize the value of the services that they offer.

> *"We wanted to create a single version of the truth by consolidating all of our BI applications and islands of information into a new set of technologies that would make it easy for analysts and end users to access the data that they needed to gain actionable insight that they could share with their colleagues."*
>
> MATT SHERRIFF, DATA WAREHOUSE DEVELOPMENT AND BI MANAGER, CUSTOMER VALUE MANAGEMENT, LLOYDS TSB[20]

A practical approach to information architectures starts with flexible and versatile Extract, Transform, and Load (ETL) and data-integration capabilities that can quickly combine data from a variety of operational data stores, data marts, and data warehouses. The next step is to provide the business with self-service capabilities for information access and analysis, which include:

The Hidden Architects

by Derek White

Achieving near-term revenue and efficiency gains will drive successful businesses during these challenging times, and perhaps, as an architect it is tempting just to focus on the systems and processes. However, it is also worth extending this to **consider the people with whom we work and how their roles and services might be provided to best meet this challenge.**

Is this straying outside the definition of our role? Possibly, when based on the ISO/IEC 42010 standard, "an architect is the person, team, or organization responsible for the system's architecture." However, to use the classic analogy with the building trade, architects are frequently involved in the appointment of both builders and specialists. Perhaps, then, this is an area in which we might have more to offer.

As architects, we bring a mix of skills that bridge the gulf between the technical possibilities and the business needs. Part of this can be an understanding of how the skill sets within our teams might achieve near-term gains. In order for people to be effective, they often need to be recognized for their skills and abilities, and then designated with appropriate roles and responsibilities. **The architect can be a key enabler in the process of recognition of skills and abilities** and, through this, add significant value to the business.

As an example, let us consider how we might designate the role of Technical Architect on a new project, given the constraints of the current climate of uncertainty, in which headcount is either frozen or being cut, and yet the work still has to be done.

Within this scenario, it might well be possible to meet this role within the wider team, especially where contractors are used. They are usually brought into a business to provide specific technical skills; however, through the breadth of their previous experience (from working with a number of other organizations), it might be that they have more to offer. Therefore, a contractor could perform the role of Technical Architect alongside an Enterprise Architect, so that the technical expertise of the contractor might be leveraged without undermining the long-term strategic direction.

To return to the challenge of achieving near-term revenue and efficiency gains, we should ask: Are we key enablers in the process of recognition of skills and abilities? Are we able to spot the people who are hidden architects?

Derek White (derek.white@pits.co.uk) is an independent consultant.

- Decision-support systems that help everyone in the organization to make better customer decisions.
- Data-mining capabilities that uncover new data relationships and help predict future customer trends, such as revenue, satisfaction, and churn.
- Data-visualization capabilities that increase the usability of customer information through graphical and visual means.
- Flexible reporting capabilities that include pivoting, drill-down, and linking, which enable people to create and manage their own real-time reports.

By providing the business with a flexible data-integration architecture and self-service data-analysis capabilities, it is possible not only to create deep customer insights, but also reduce the backlog on IT and increase the timely flow of information through the organization.

Improve Operational Insight

> *Beware of little expenses: A small leak will sink a ship.*
>
> BENJAMIN FRANKLIN

With a flexible information architecture, businesses can improve performance management, reduce risk exposure, identify sources of revenue leakage, eliminate inefficiencies, and recognize and realize new growth opportunities.

Performance-management solutions include:

- Scorecards and dashboards that monitor and measure process effectiveness and chart progress towards strategic business objectives.
- Key Performance Reporting that displays metrics to track personal and team performance and drive accountability in the organization.
- Business-initiative monitoring that collects and distributes information that illustrates the effectiveness of business initiatives.

In Europe, the Middle East, and Africa (EMEA), Microsoft relied on local subsidiaries to feed sales data into a proprietary application that lacked the functionality to produce sales forecasts. Microsoft implemented a performance-management solution that enabled employees to create and share sales forecast graphs and tables quickly, in real time, across portal sites. As a result, sales forecast accuracy and revenues for the group have increased 31 percent.[21]

An effective Business Activity Monitoring (BAM) architecture, or the real-time monitoring of business activities, gives organizations the ability to track business processes from a variety of tools, including Web services, Web portals, Office Excel, other desktop clients, and mobile devices. This provides an increased level of visibility into critical processes that drive revenue or—when they malfunction—that do not.

ASB Group Investments—a default provider for the New Zealand government's retirement savings program, KiwiSaver—uses BAM to reconcile the financial totals of all contributions against a batch total that is received daily.

> *"[By using BAM], we can define and track key performance indicators throughout the process of receiving and managing KiwiSaver contributions. A critical aspect of being an institutional participant in this program is visibility into the process.*

"If something goes wrong, we can use the BAM functionality to drill down and find out what happened. This is extremely helpful. Now, we have the tools to locate and fix problems quickly."

ROBERT MAHER, TEAM LEADER OF THE MIDDLEWARE STREAM, STRATEGY AND ARCHITECTURE, ASB TECHNOLOGY SOLUTIONS

Helping All Users Increase Productivity

Deliver User-Focused Applications

It is no use saying, "We are doing our best." You have to succeed in doing what is necessary.

WINSTON CHURCHILL

People are at the core of business. It is their creativity, innovation, collaboration, and passion that drive a business to succeed. Helping people communicate, collaborate, be more productive, and make better decisions should top the list of practical architecture projects.

To help engineers improve productivity, Honeywell Transportation Systems designed and developed a system in only four months to integrate SAP information into familiar Microsoft Office tools.

"SAP is critical to our business, especially to our supply-chain and manufacturing organizations. But not all of our engineers are SAP power users. SAP is a very transactional system, and engineers are not used to it, because they don't use it continuously as our supply-chain workers do. Every time they accessed an SAP part number, it required multiple clicks, which amounted to several hours per week.

"By using smart tags, we were able to reduce two to three minutes' worth of work that only a limited number of people knew how to do to a single mouse-click that everyone can do. It represents a tremendous efficiency gain across the company, lets our engineers take full advantage of the investment we've made in SAP and MatrixOne, and broadens access to information."

JERRY IBRAHIM, DIRECTOR OF IT EMERGING TECHNOLOGIES, HONEYWELL TRANSPORTATION SYSTEMS[22]

Historically, users have worked on their PCs (or terminals) to access monolithic business applications. Today, users are operating a wide range of devices. Inside their organizations, they are using a desktop PC; on the road, they carry a laptop; they are connecting to the Internet through a variety of devices; and they often do work at home by using their PCs.

To deliver applications that are accessible from a wide range of presentation technologies—desktop, Web, RIA, portal, and devices—architects must design multichannel architectures by using loosely coupled, independently evolvable pieces that are facilitated by flexible development frameworks and tools that support a wide range of application types and architectural styles, as well as connected and disconnected scenarios.

"The more relevant we are to our customers, the more they will appreciate it, and the more likely it is that they will stay with us. We are driven by what our customers expect from us and know that if we offer more dynamic and more efficient channels for interaction, our relationship with our customers will be strengthened."

PATRICK CHEW, HEAD OF DELIVERY, CONSUMER FINANCIAL SERVICES, OCBC BANK[23]

The Architectural Journal #10[24] focuses on composite applications, and the recent patterns & practices release, "Composite Application Guidance for WPF and Silverlight," is designed to help you more easily build enterprise-level applications that span multiple delivery channels and presentation technologies.

Connecting People and Teams

None of us is as smart as all of us.

KEN BLANCHARD

In today's highly interconnected world, it is not uncommon for people and teams across separate organizations, companies, and even countries to come together to deliver business results.

For architects, this means creating effective collaboration-oriented architectures[25] that help people create, distribute, analyze, and share information by using a combination of on-premise and off-premise software and services. This de-parameterization—characterized by systems that span organizations, systems, networks, and security domains[26]—presents a variety of new and interesting considerations.

A practical approach focuses on increasing the productivity of distributed people and teams by simplifying the access to shared information. Key focus areas include:

- Integrated communications that unify e-mail, instant messaging, voice mail, telephones, mobile devices, and Web conferencing into a single experience.
- Collaborative workspaces that allow people to create and manage collaboration spaces and access them from multiple tools: e-mail, phone, Web conferencing, portals, and productivity applications.
- Locating information and people across multiple information sources and organizations.
- People-driven processes that enable unstructured and ad-hoc business processes to connect with automated business processes.

Capgemini, one of the leading consulting firms in the world, wanted to bring its people and knowledge together globally to form highly efficient teams that could access information quickly to win new business and complete projects productively. By improving access to critical information, experts, and colleagues, thus reducing delays and increasing productivity, Capgemini teams can respond to new opportunities faster and have increased their new business close rate. The ability to win business and work smarter and faster is expected to result in a payback of eight months and a net present value per user of more than $2,700 USD.[27]

Team-Based Application Development

It's easy to get good players. Getting them to play together, that's the hard part.

CASEY STENGEL

Custom business applications that result in the ability to outperform competitors can result in a competitive advantage. As a result, increasing the predictability, repeatability, and measurability of team-based application-development processes is as critical as optimizing such traditional business operations as manufacturing, inventory, and production.

By reducing the friction between development roles—architect, analyst, designer, developer, and tester—businesses can expect to increase productivity, as well as deliver more stable, secure, and scalable business applications. A practical approach to team-based software development uses a Lean approach that focuses on:

- Improvement of the flow of value to the organization by reducing risk, increasing business visibility into the development process, and improving the ability to meet the needs of the business.
- Reduction of waste by eliminating unnecessary code, optimizing processes, shortening development cycles, and avoiding unnecessary complexity.

> *"A Lean approach to software engineering thinks end-to-end about the optimal flow of value and the continual reduction of waste. Let me give you an example: One of the most difficult problems has always been bugs that can't be reproduced: the 'no-repro' bug. By eliminating this waste through better tooling and processes, we can build a better bridge between development and test."*
>
> SAM GUCKENHEIMER, GROUP PRODUCT PLANNER, MICROSOFT VISUAL STUDIO TEAM SYSTEM AND AUTHOR OF SOFTWARE ENGINEERING WITH VISUAL STUDIO TEAM SYSTEM

EDS, a global business- and technology-services company, realized a 286 percent ROI in four months by using Microsoft Visual Team System to support a more efficient global development strategy and realign its internal software-development assets.[28]

Over six months, Dell—a leading manufacturer of PCs and technology-related products—realized a 225 percent ROI by deploying a centralized global source-code management platform to provide developers with source code on a just-in-time basis, regardless of their location. Improved source-code management has also enabled Dell to consolidate its source code onto fewer servers, redeploy 100 system administrators, and improve the productivity of its developers.[29]

Conclusion

> *You may delay, but time will not.*
>
> BENJAMIN FRANKLIN

Today's turbulent business climate creates interesting challenges for architects who are responsible for reducing costs and delivering new capabilities. A practical approach to architecture focuses on:

- Balancing business, technology, and user goals with the goal of delivering near-term revenue and efficiency gains.
- Taking a holistic approach that reduces the complexity of how infrastructure, applications, and tools work together to support critical business applications.

The Microsoft Application Platform, which provides all of the capabilities that this article covers, is an integrated and interoperable set of mission-critical server infrastructure, applications, and tools that helps organizations to reduce operations costs, improve business flexibility, and increase productivity for all—developers, users, and administrators. More information, visit the Microsoft Application Platform Web site.

References

1. http://www.mckinseyquarterly.com/Business_Technology/BT_Strategy/Managing_IT_in_a_downturn_Beyond_cost_cutting_2196
2. http://www.gartner.com/DisplayDocument?ref=g_search&id=920217 (subscription/purchase required)
3. http://apparchguide.codeplex.com/Wiki/View.aspx
4. http://www.burtongroup.com/Research/PublicDocument.aspx?cid=1572
5. http://msdn.microsoft.com/en-us/practices/bb190351.aspx
6. http://technet.microsoft.com/en-us/solutionaccelerators/dd320379.aspx
7. http://technet.microsoft.com/en-us/solutionaccelerators/default.aspx
8. http://msdn.microsoft.com/en-us/architecture/default.aspx
9. http://harvardbusinessonline.hbsp.harvard.edu/b02/en/common/item_detail.jhtml;jsessionid=300XUPXY2GWQIAKRGWDSELQBKE0YIISW?id=U0808A&referral=2341
10. http://en.wikipedia.org/wiki/Systems_thinking
11. http://en.wikipedia.org/wiki/Theory_of_constraints
12. http://www.projectsmart.co.uk/docs/chaos-report.pdf
13. http://apsblog.burtongroup.com/2008/03/looking-for-soa.html
14. http://en.wikipedia.org/wiki/Zachman_framework
15. http://en.wikipedia.org/wiki/TOGAF
16. http://www.microsoft.com/infrastructure/about/overview.mspx
17. http://www.microsoft.com/downloads/details.aspx?FamilyID=F190007A-B46A-48CA-9703-F5110B8BA0FB&displaylang=en
18. http://www.microsoft.com/casestudies/casestudy.aspx?casestudyid=4000002874
19. http://download.microsoft.com/download/a/7/0/a70394d4-164b-47e3-906d-71275b6b6ae3/EDS%20and%20Microsoft%20WP_SOA%20in%20the%20Airline%20Industry.pdf
20. http://download.microsoft.com/download/3/8/E/38E37AF9-A6AC-429B-8AE7-FF83C600A75B/LloydsTSB_BI_Final.doc
21. http://www.microsoft.com/casestudies/casestudy.aspx?casestudyid=4000000079
22. http://www.microsoft.com/casestudies/casestudy.aspx?casestudyid=4000001333
23. http://www.microsoft.com/asia/partnersolutionmarketplace/philippines/CaseStudyDetail.aspx?casestudyid=201248
24. http://msdn.microsoft.com/en-us/architecture/bb266330.aspx
25. http://www.opengroup.org/jericho/COA_v1.0.pdf
26. http://en.wikipedia.org/wiki/De-perimeterisation
27. http://download.microsoft.com/download/A/D/1/AD1BB687-C475-41B0-925C-084E394C3299/collaboration_vision_white_paper.doc
28. http://www.microsoft.com/casestudies/casestudy.aspx?casestudyid=201310
29. http://www.microsoft.com/casestudies/casestudy.aspx?casestudyid=201353

Acknowledgements

Special thanks to Sam Guckenheimer, J.D. Meier, Mike Hatch, Lynn Langit from Microsoft, and Clyde Keeling at EDS for their help and subject-matter expertise.

About the Author

Matt Valentine (mattva@microsoft.com) joined Microsoft in 2003 to focus on helping customers address business challenges by using world-class distributed-computing technologies. He has over 15 years of experience in architecting and delivering application, database, Web, and transaction systems solutions with major corporations, including AT&T, IBM, ATG, and Microsoft. Key distributed-application projects have included telecommunications OSS projects, distributed insurance solutions, and several of the world's largest travel e-commerce sites. Matt holds a BS in Computer Science from Bucknell University and works on the Application Platform Marketing team at the Microsoft world headquarters in Redmond, WA.

Reducing Infrastructure Costs Through Virtualization

by Leandro Sgallari

Summary

Nowadays, when we think about projects, it is necessary that we have a precise budget and defined cost beforehand. This article identifies the virtualization model—at all possible levels—as the best fit today.

Introduction

Nowadays, we live in uncertain times all around the world. When it comes to architecture and design, we must think a lot more in costs—unlike in other times, when we used to think first of the solution and then in costs. If we had a streamlined and defined return on investment (ROI), only then did the project get the green light. Today, if we think about projects, it is necessary to have a precise budget and defined cost first before we can start to think about the project.

Upon brief reflection of what IT architecture is, one finds that the model that the whole world favors (because of costs and the evolution of technologies) is the virtualization model. Ten years ago, it was all about decentralization—both in data centers and servers, and in communication and desktops. When we thought of an application, we always tried to have the layers of such an application as close to the client as possible; data centers, databases, and e-mail servers were distributed all over.

This situation was the result of being unable to face the huge cost of having hardware equipment of multiple large capabilities (such as an eight-processor server and lots and lots of RAM gigabytes) or having redundant point-to-point communication links with good bandwidth—the cost of which only large companies could consider including in their architecture. All of this encouraged a decentralized IT administration model that required specialists in the different platforms of each site.

The Original Model

Although this model worked for years, many things were not considered that today have rendered this model not as efficient as was initially thought.

Taking as an example the analysis of a distributed application that was devised 10 years ago, there was an architecture in which it was important to have the data near the client, which led to the following scheme:

> *A database in the central office in which the information from the different sites, the database from each site, the application server from each site, and the local applications that were installed on each desktop were all consolidated.*

A replication scheme among the databases was used for the distribution of information. This drove us to have database administration for each site, besides having on each site an infrastructure administrator who had thorough knowledge.

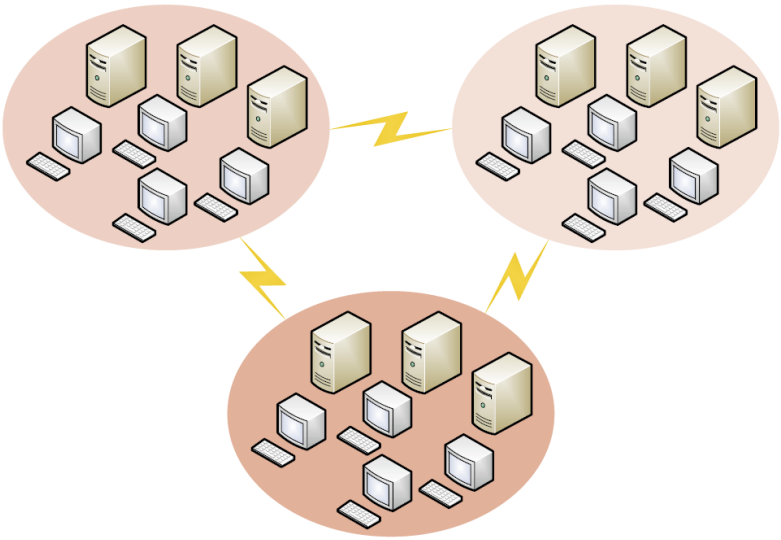
Initially, this had (as variables within the equation) high communication-link costs, large servers that represented a very high cost, and operative systems that were neither very solid nor rigid with regard to changes and also had little functionality—that is, they offered very few functions or roles within the operative system.

For many of the needs of the company, it was necessary to add software that could comply with that functionality. In addition, in order to carry out the tiniest of changes, it was necessary to set the server offline and have IT personnel who represented an average cost. The variable that was not really considered was the updating and maintenance of the whole structure, which at the time—due to the fact that technology did not evolve in the way that it does today—was not such an important aspect.

If we consider basic accounting principles (which I have learned during recent years), one should always see the IT personnel as an asset to the company, with both amortization time (which is the time that it takes to shape the person, according to the culture and needs of the company) and an updating cost (which is what must be invested to have a person trained in the different technologies as they evolve).

Over time, all of this changed; the variables in this equation also changed, and the updating and maintenance variable (which in many cases had not been taken into account) started to gain more and more importance.

Figure 1: Typical scheme



This is the equation that we face nowadays:

> *Average to low communication-link costs (taking into account the virtual private network), large servers with many RAM gigabytes at average to low cost, operative systems that had hundreds of embedded and flexible options and lots of functionality (many things already come solved and embedded in the operative system, so that in general it is not necessary to set the server offline to make these changes), average to high personnel costs, and average to high updating and maintenance costs.*

Within the scheme that is encouraged today, many things must be considered; it is necessary, therefore, to have the whole scheme in mind—not just a part of it—to avoid making the same mistakes that we incurred in the past.

Nowadays, when uncertainty (crises, corporate mergers, acquisitions, and constant changes) is all around, it is vital to work toward an environment that would basically support constant dynamic changes. More than ever, it is necessary to think about platform and application updating, growth, and corporate and budget contractions. This, of course, will highly influence the model that is to be chosen, and that model (taking into account the aforementioned equation) should be based mainly on the updating and maintenance variable.

When we consider all of the preceding, we will see that the model that best fits is the virtualization model applied to all of the possible levels, where all of the equation variables are considered in order to determine feasibility and total cost of ownership (TCO).

There will be infinite virtualization scenarios—from choosing cloud computing in specific services and virtualizing (or outsourcing) the whole or part of the IT department to using virtualization for servers, applications, or desktops.

Server Virtualization

Today, there are many important players and technologies that have been widely tested, such as Microsoft Hyper-V and VMWare. The hardware costs have gone considerably down: If we were to compare four- to eight-processor equipment of the past to one today, it would result in an important cost margin that would be an improvement, and it would be necessary to add the progress that has been made in technologies and redundancy within equipment, board, hot-plug memories, and so on.

Generally speaking, almost all the components of the server can be changed without having to take the server offline. The same thing applies to operative systems. This means that we can do away with the theory that was used in the past, according to which we used to divide into different hardware pieces the different business applications. In addition to this, the advantages of tolerance to failure in the virtualization schemes that are used today make it possible to take a physical server offline without affecting the virtual server that is running in that physical server. This, of course, means that from a simple technical point of view, there is already a huge advantage in the use of server virtualization.

Application Virtualization Considerations

by Darryl D'Costa

Initially, the motivation to virtualize an application will come from the need to isolate it and thus resolve a conflict between the application and some other installed component. Subsequently, the need to virtualize applications will be driven by other reasons, such as making application updates and upgrades easier, keeping applications disconnected from the operating system, and reducing overall desktop-management costs.

Regardless of the reason, virtualizing some applications will simply not be possible for several reasons. For example, if you use Microsoft Application Virtualization (App-V), you will not be able to virtualize applications that use a system-level driver, applications for which shell extension support is required (for example, "right-click the desktop and choose **Add to WinZip...**"), or applications that use a boot-time service or COM+. These candidates will be easy to identify.

Additionally, there will be a subset of applications that you will not be able to virtualize although they do not meet these criteria—that is, they appear OK to virtualize, but will not. This will include applications that have poorly designed installers that cause issues that cannot be resolved, or applications that virtualize but whose deployment becomes too confusing or complex for users (for example, multiple plug-ins that are virtualized separately with Microsoft Internet Explorer and published with separate icons). Having gone through this process, you will have an initial inventory of applications that you can and cannot virtualize.

When you have an inventory of your applications, it is essential next to understand what those applications do and on what components they depend. With traditional application installation, you just install an application, and the application's installer (or you) installs the dependencies, such as the required version of Microsoft .NET Framework or Java. When you virtualize applications, however, you must take this a step further by coming to terms with whether the application's dependencies are required only by the application or whether the dependency is a key component that is used by other (virtual and nonvirtual) applications.

If the case is the latter, you have another decision to make: Should the dependency be installed physically on the workstation—thereby available to both virtual and nonvirtual applications, or does it get virtualized separately—thereby available to multiple virtual applications? With App-V, you can virtualize components and applications individually, then dynamically connect them later without having to resequence the application. This is known as Dynamic Suite Composition.

The decision to virtualize applications is not just a matter of whether they can be virtualized and the challenges that might be involved in sequencing the applications successfully. While there are many advantages to virtualizing applications, some risks remain. Unpatched virtual applications can be just as vulnerable as unpatched locally installed applications.

Additionally, most of today's Microsoft Windows applications were not developed to run in a virtual environment. Another risk is that the vendor of the application might deny you support, because the vendor has not tested or certified the application in a virtual environment.

Finally, as previously noted, Windows applications tend to integrate with the shell—which can be problematic and require extra work and time to virtualize the application, if at all.

Darryl D'Costa (ddcosta@microsoft.com) is an architect at MS Consulting Services Australia.

From an architectural point of view, this allows us to respond to organizational changes quickly. Above all (and this is something to consider nowadays), this also enables us to achieve substantial savings at the time of shaping a data center. There will also be savings in the electrical bill, refrigeration costs, physical space, and hardware. Moreover, deployment and disaster recovery will be much simpler. All of this will result in lower maintenance costs—whether we have our own IT department in 100 percent administration of this platform or a virtualized environment of the IT department (later, we will develop the IT department cost, as well as the options and the reasons behind virtualizing it). This all leads to there not being almost any scenario in which virtualization is not applied and which will not result in a much lower TCO.

Desktop Virtualization

In this regard, it is also possible to find very well-developed and well-tested technologies, such as Microsoft Terminal Server, Citrix, and so on. This kind of virtualization was previously thought about for remote points or links that had a relatively small bandwidth. Nowadays, it is used as a method to reduce desktop-administration expenses, because (thanks to this technology) it is possible technically to have tolerance to failures, add it to the server-virtualization scheme, and create a pool of servers. There is substantial reduction of the desktop-maintenance cost and the cost of desktops themselves, because with equipment that has smaller hardware, it is possible to run any kind of application and still have centralized control and deployment of applications and security policies.

Virtualization of the IT Department

In the past, there was a relatively low or not-so-significant IT department cost, compared to the one nowadays. Today, there is a high IT department cost, and it is necessary to consider the following variables: training in new technologies, training in the company environment, the cost of personnel search, and the time during which the search takes place. All of this leads to the IT department not being able to respond with the speed that the company needs. In addition, we currently experience a high labor turnover, which means that many times this process has to start again—which, of course, drives the cost upwards.

If we consider all of these factors, especially the costs and the time that the search involves, we will see that having a virtualized IT department results in a lower TCO and in every possible advantage. With virtualization, all of these IT-department problems are moved to an external company that is exclusively devoted to IT, particularly as it refers to specialists in technology or in specific technologies. This would mean that there is no point in having a specialist as part of the internal IT department.

Cloud Computing

Currently, there an infinite number of services are available on the Web, from e-mail services (as has been the case for a very long time) to CRM, ERP, Document Managers, and other services. This solution naturally offers a world of advantages: It is unnecessary to have a specialist in this technology within our IT department, it is equally unnecessary to maintain that technology from either the hardware or the software point of view, and keeping security copies of the information is no longer required. Depending on the kind of hired service and service-level agreement (SLA), it will be possible to have a redundant and always-online service. In some cases, the cost for this kind of service can be high—depending on both the number of users within our organization who require this service and the characteristics of the service—and is worth considering.

Four Steps for Virtualization

by David Ziembicki

Instead of selecting a one-size-fits-all solution, virtualization provides architects with a new, more flexible set of choices that can be combined to optimize the cost and user experience of the desktop infrastructure. The following four steps lead to an optimized solution.

Define User Types

Analyze your user base, and define between three and five categories—such as Mobile Workers, Information Workers, Task Workers, and so on—and the percent distribution of users. The requirements of these user types will be utilized to select the appropriate mix of technologies.

Define Desktop-Architecture Patterns

Each architecture pattern should consist of a device type (thin client, PC, and so on) and choice of one of the following:

- OS execution (Local or Desktop Virtualization)
- Application execution (Local or Application Virtualization)
- Display (Local or Presentation Virtualization)

For each pattern, determine to which user types it can be applied. For example, with mobile or potentially disconnected users, Presentation Virtualization alone would not be applicable, because it requires a network connection.

Determine TCO for Each Architecture Pattern

Use a recognized total-cost-of-ownership (TCO) model to determine the TCO for each pattern. Minor adjustments to these models can be made to account for specific technology differences, but most include TCO values for PCs, PCs with virtualized applications, virtual desktop infrastructure (VDI), and Terminal Services/Citrix thin-client scenarios.

Model Desktop-Optimization Scenarios

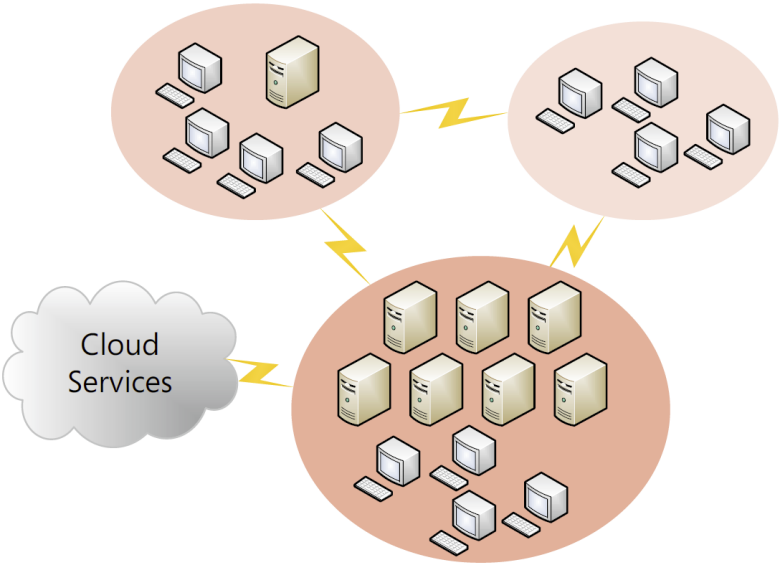
By using the preceding data, appropriate architecture patterns can be selected for each user type by choosing the lowest TCO architecture pattern that still meets user requirements. By varying the user distribution and selected architecture patterns, an optimized mix can be determined.

A one-size-fits-all approach would result in either a lot of PCs (if virtualization is not used), a lot of servers (if everything is virtualized), or failure to meet power user needs (if only thin clients are used). An optimized solution is one that utilizes the right mix of technologies to provide the required functionality for each user type at the lowest average TCO. Combined with a unified management system, substantial cost savings can be realized.

For more information, visit my blog.

David Ziembicki (davidzi@microsoft.com) is an architect with Microsoft Consulting Service's Federal practice.

Figure 2: Current architecture



Example of Architecture
(Example based on a company with 500 employees.)

As an example, we will use a virtualization architecture that uses such Microsoft technologies as Hyper-V and Terminal Server.
Number of servers: 15.

Typical structure of IT department: One manager, two IT administrators, one database administrator (DBA), and five Help Desk employees.

Based on everything that was explained previously, we will take the best of each virtualization technology to carry out a cost reduction.

Server Virtualization
It will be possible to reduce approximately 15 noncritical servers into 4 physical servers that will be able to support these 15 virtual servers. It will be necessary to carry out a load analysis and distribute the servers and business applications correctly. Nowadays, given the kind of roles of a typical company server, there are not many servers that have a high processing consumption; thus, it will be necessary to isolate these kinds of servers, so that a resource competition conflict is not generated. It will also be necessary to include (if we do not already have it) an external storage in which our virtualization scheme will be stored (so that it is possible to work on it in a cluster) and which will have tolerance to failure of all of the virtual equipment. All of this will be possible by using Microsoft Windows Server 2008 64 bits and Hyper-V System Center Virtual Machine Manager to carry out the P2V conversions.

Cost Reduction
Within this example, it will be possible to reduce approximately 70 percent of the energy consumption, as a result of less consumption on the part of the servers. In addition, there will also be a reduction of approximately 70 percent in the refrigeration consumption, as a result of the use of storage.

The licensing cost will also decrease (when we use Microsoft licensing) very substantially. The Microsoft licensing scheme is based on Table 1. In the table, we can see that by using Windows Server 2008 data-center server licensing, it will be possible to obtain a reduction in licensing from 15 servers (which will be able to use different versions of Windows Server 2008, depending on the processor and RAM needs) to only 4 (with data-center or enterprise licenses). Depending on the versions of Microsoft Windows that are used, in the least favorable scenario, we will achieve a reduction in cost of 50 percent.

Desktop Virtualization
Depending on the memory consumption of the applications, it will be possible to implement approximately five virtual servers for Terminal Server—typically, in five physical servers to cover 500 work positions. The main advantage of having virtualized servers is that this will automatically commute to any other, in case of a failure in any physical equipment. In this way, we will be able to have a desktop with fewer resources, and it will be possible to update the applications more rapidly, as with deployment, management of printers, and any other desktop problem. In turn, this will also enable us to make the desktop of the user available to remote or external users.

Cost Reduction
If we consider, on the one hand, the cost of updating 500 desktops as a result of the installation of some business application and, on the other hand, the purchase of five 32-GB RAM servers and two Quad Core processors each, we will obtain a cost reduction of approximately 90 percent.

Virtualization of the IT Department
It is first necessary to analyze the critical and noncritical applications; it is important also to analyze the IT labor market in the country in which it is applied. Generally speaking, the advice that is given is to virtualize whatever is difficult to get in the market and to have partial virtualization of the IT department. For this example—and considering the current work market—we will opt to virtualize (for example) only one IT administrator and one DBA; the Help Desk, one IT Administrator, and the IT manager will continue to be physical. By having an SLA with external suppliers and a framework work contract, it will be possible to increase rapidly the IT department or change swiftly the scheme without a great increase in initial costs. It will also be possible to decrease training costs, hiring costs, and so on.

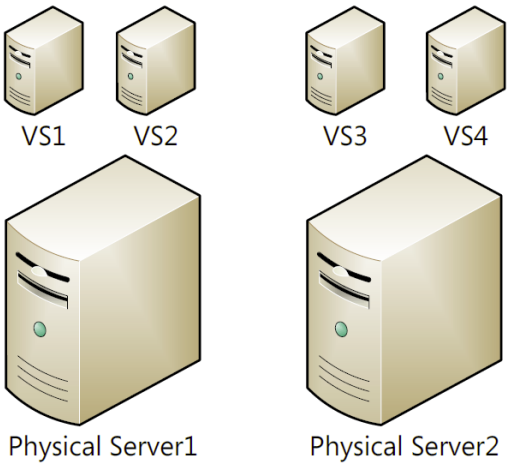
Cost Reduction
If we consider the TCO, hiring costs, training costs, and salaries, we will obtain a cost reduction of approximately 55 percent.

Cloud Computing
Let us take an application that will not be worth having internally, because of the size of the company. For this example, we will use

Table 1: Microsoft Licensing Scheme

Version of Windows Server 2008 host	Covered virtual servers
Standard	1
Enterprise	4
Data center	Unlimited

Figure 3: Virtual-server scenario



a CRM. Ten CRM licenses will be hired online. In this way, no costs will be associated to the initial licensing, administration training, CRM server deployment, disaster-recovery policies, or anything that pertains to administration of the CRM.

Cost Reduction
Based on online services, there will be a cost reduction of approximately 80 percent—based on 10 licenses, and considering the initial cost of having a server, trained personnel, backup policies, and so on.

Conclusion

The virtualization scenario makes it possible to make structural changes in the IT department with the speed that the market actually needs.

We can have strong cost reduction, because with a physical structure, we often do not use all of the resources—hardware, software, employees, and so on—at 100 percent. On the other hand, with virtualization we have the opposite case: We use and push the resource utilization as far as possible, and then we add more resources to virtualize.

In our example, we can see the individual cost reduction; if we look at it globally, however, the cost reduction is more significant.

About the author

Leandro Sgallari (LeandroS@tisistemas.com.ar) is owner of and consultant at TI Sistemas in Argentina and has over 12 years of experience in IT infrastructure. He is project leader and infrastructure architect for IT projects. Leandro is a Microsoft Certified Trainer and teaches Microsoft Windows 2000/2003 official courses in Microsoft Certified Partner(s) for Learning Solutions (MCPLS). He is a Microsoft Certified Systems Administrator (MCSA) and Microsoft Certified Systems Engineer (MCSE) in Windows 2000/2003, as well as an early achiever in Windows 2000/2003, Microsoft Certified Trainer (MCT), Microsoft Certified Database Administrator (MCDBA), and Solutions Architect MVP. Visit Leandro's profile.

Follow up on this topic

- Virtualization for Decision Makers
- TechNet Magazine on Virtualization (October 2008)
- Virtualization Resources for IT Pros

Virtualization for Development Organizations

by Matthew Pyle

Many software-development organizations today require an increasingly complex end-to-end infrastructure-optimization strategy. Each distinct business unit in an organization has specific resources that it requires from the infrastructure of the organization. With the use of Microsoft System Center Virtual Machine Manager 2008 (SCVMM) and Self-Service Portal components, server-resource requisitions can be templated, deployed, and archived without intervention from the Information Services department. With the resources that are gained, the organization is better equipped to plan for the future. And the future is here, now!

The advanced boot from .vhd features of Microsoft Windows Server 2008 R2 and Microsoft Windows 7 are amazing. Virtual hard disks that were provisioned by using SCVMM can then be copied and booted to physical hardware. The ability to boot natively from .vhd files right to the physical hardware opens virtualization technology to a level of physical interaction that before was unheard of. When these .vhd files are booted and interacting with the physical hardware, they can use this hardware for more complex functionality, such as advanced hardware graphics and USB device support. An organization that uses a combination of the SCVMM Self-Service Portal virtual-machine provisioning and the advanced boot of .vhd features really has the best of both worlds.

Microsoft even has a solution for the users in your organization who want everything to work reliably, all of the time, and with minimal intervention on their part. Microsoft has hit another home run with the Presentation Virtualization options of Windows Server 2008. Through the Presentation Virtualization Remote Desktop Web Access, Remote Desktop Gateway, and RemoteApp, you can provide users with a secure, easy-to-use, and reliable way to access their applications both inside and outside of the organization.

By offloading the processing power that is required to run applications from the workstation to the server, organizations have the option of building a more reliable and scalable virtual infrastructure, while minimizing the investment in the desktop. With proper utilization of these technologies, I can even envision users working on ultralow-power workstations and netbooks.

For more information, visit my blog.

Matthew Pyle (mpyle@interknowlogy.com) is a Senior Systems Engineer at InterKnowlogy.

Engineering in the Cloud: An Engineering Software + Services Architecture Forged in Turbulent Times

by Peter Williams and Simon Cox

Summary

This article describes both the architectural challenges that are inherent in implementing "Engineering in the Cloud" and an architecture that we call "Engineering Software + Services."

Introduction

Companies in all sectors are looking to cut costs, develop new revenue streams, and increase productivity. This is especially true in turbulent times. They are also looking to become more agile in a competitive landscape, reduce their time to market, create better products, and adhere to compliance requirements. This is particularly true in the engineering sector, in light of increasing globalization and the requirement to remain competitive.

This article describes the architectural challenges that are inherent in implementing "Engineering in the Cloud" and an architecture that we call "Engineering Software + Services." We illustrate this with a case study, drawn from dezineforce.com, and demonstrate how this architecture can deliver the functionality that is required by engineering companies who want to transform the way in which they work and prosper in turbulent times and beyond.

Engineering in the Cloud

Computationally-aided engineering has matured over many years, with computer simulations able to predict accurately the real-world characteristics of engineering designs. This can significantly reduce the need for physical testing that has traditionally been carried out at considerable expense. However, engineering companies are increasingly finding themselves having to acquire IT skills to manage and maintain these packages, as well as the complex infrastructures on which they execute.

Computation facilities in the Cloud could clearly provide many benefits. But would it really meet the needs of engineering companies, particularly in these turbulent times? It might reduce costs and make technology that had previously been available only to the largest of corporations available also to small and midsize companies. But how do we, as architects, harness the power of the Cloud to really transform the way in which engineering is conducted?

In order to affect a step change and meet the current needs of engineering companies, we believe that "Engineering in the Cloud" must also provide a level of intelligence and rich interaction that allows engineers to gain additional insights into their designs. Therefore, we define it thus:

> *Engineering in the cloud is a combination of cloud services and rich interactive applications that provides integrated, intelligent, self-service engineering services over and above engineering-application hosting and computation—allowing engineers to create, explore, and discover better designs faster.*

Engineering Software + Services Architecture

Many publications exist on the challenges of building a Software + Services platform. The general concerns of security, availability, and reliability, among others, all apply as much to engineering as they do to other sectors. Instead of reiterating these more general concerns, we will focus on the less common challenges that are addressed by the "Engineering Software + Services Architecture" as a means of implementing an engineering cloud service.

There are five key, specific architectural challenges on which we will concentrate:

- Engineer interaction
- Engineering intelligence
- Engineering process orchestration
- Engineering computation
- Long-term, large-scale data management

The various subsystems in an engineering Software + Services architecture can be thought of as a number of layers and crosscutting concerns, as shown in Figure 1.

Security and management are not part of our set of five specific challenges, but they are briefly included for completeness. Within dezineforce, these challenges are implemented using the technologies shown in Figure 2.

Figure 1: Generic high-level engineering Software + Services architecture

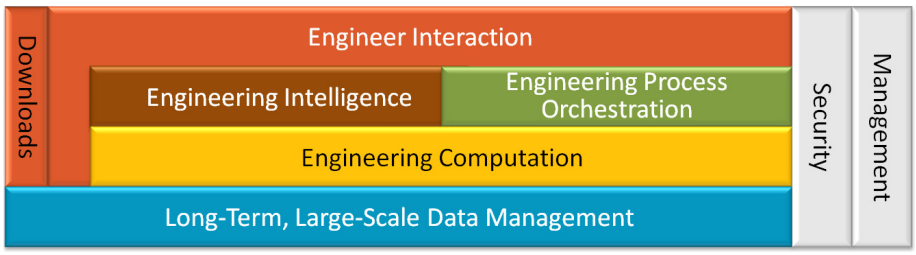
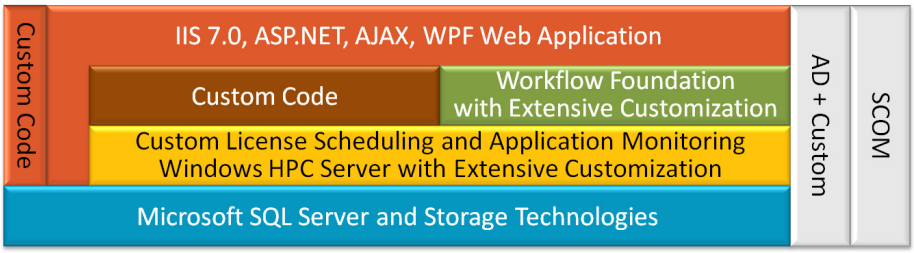


Figure 2: The dezineforce high-level engineering Software + Services architecture



Engineer Interaction

Engineering processes might take hours, days, or weeks to complete, and engineers must remain in full control of the processes throughout their execution. Therefore, it is essential that they can see into the processes to understand the progress and current status of their exploration. This ranges from understanding which loop the process is in, right down to inspecting the individual output files of currently executing jobs. The ability to detect and react quickly to issues reduces wasted compute cycles and elapsed time, which reduces costs and improves time to market.

These processes create very large volumes of data, with tens of gigabytes for a single run being fairly normal; hundreds of gigabytes are not unheard of. In order to increase engineer productivity, the architecture must allow engineers to conduct intelligent and powerful searches and interrogate results without having to download the complete data set. The service must provide tools to allow engineers to view summaries of the data and relevant sections of the output files. These tools can be embedded in the service or provided as rich client tools.

Within the dezineforce service, engineer interaction is implemented using Microsoft Internet Information Services (IIS) 7.0, ASP.NET, and AJAX—providing a good experience, with broad reach. There is also a rich 3-D visualization tool that is implemented as a Windows Presentation Foundation (WPF) browser application to view and interact with the output of the optimization process. This technology was selected because it provides a rich, hardware-accelerated, interactive 3-D experience within the browser—without the need to install software explicitly—and was a good match for the existing Microsoft .NET skill sets of the team.

The use of WPF allows for rich rendering of the mathematical models by using lighting effects and camera positioning to see details that otherwise are difficult to bring out. For example, lighting effects are particularly effective in side-on views of complex surfaces, as shown in Figure 3.

Because many design engineers already have high-performance CAD workstations, the ability to leverage their hardware capabilities allows for rich interaction with the models, while still providing high-quality visuals.

Engineering Intelligence

During the design process, an engineer is faced with a great deal of choices, requirements, and constraints that are often at odds with each other—requiring a set of trade-offs to be made.

As the number of variables increases, it is not feasible for engineers to explore all possible design options exhaustively. It is also unrealistic to expect them to discover counterintuitive designs without a thorough systematic investigation—particularly, on new types of design of which there is little prior industry knowledge.

Figure 4 shows two guide vanes (situated at the inlet of a jet engine). The non-intuitive design is significantly better, but it is very unlikely to have been arrived at by designer judgment and hosted computation alone.

(These designs are the result of a study that was carried out for Rolls-Royce using the optimization toolkit that is incorporated in the dezineforce service. They are included courtesy of Professor Andy Keane [University of Southampton] and Rolls-Royce.)

Achieving these high-performance designs requires far more than just raw computation and automation. It requires a layer of engineering intelligence, while keeping engineers in control—guiding them through this maze in a systematic, efficient, and informative way.

This approach can reduce costs significantly, because it requires an order-of-magnitude fewer simulations. Given finite computing resources, this also reduces time to market. The ability to discover non-intuitive and counterintuitive designs leads to better products and allows engineering consultancies to offer new services to their clients—thereby, generating new revenue streams.

dezineforce provides a layer of intelligence through the use of advanced Design Search and Optimization (DSO) algorithms. This optimization suite was developed over many years and informs the process orchestration. A detailed explanation of these algorithms is beyond the scope of this article (see the "Resources" section for further reading).

The output of these computational methods can be used by an engineer to visualize the design trade-offs. In this real-world example of a simple problem, it can be seen that with very few simulations (indicated by the colored spheres in Figure 5), the optimization algorithms have predicted the shape of the surface and directed

Figure 3: Ambient light (L) and directional light (R)

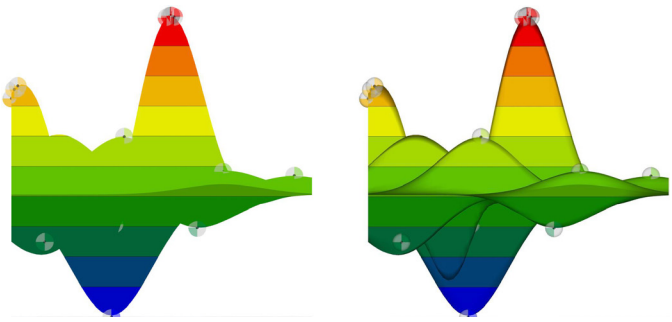


Figure 4: Intuitive guide vane (L) and non-intuitive (but superior) guide vane (R)

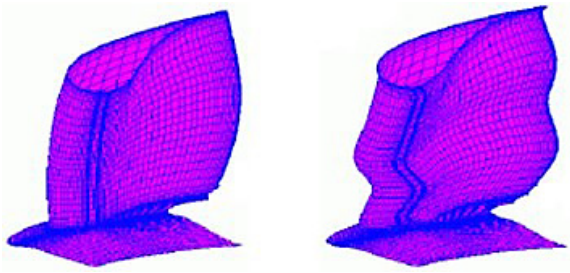


Figure 5: Surrogate model displayed in a WPF browser application

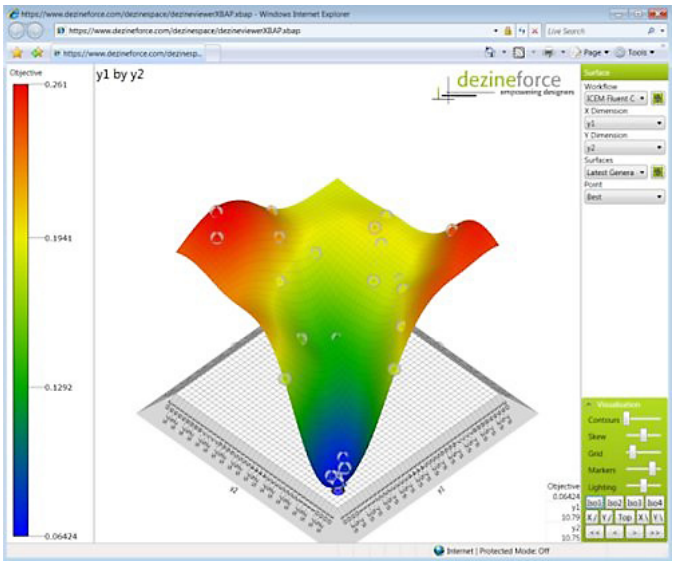
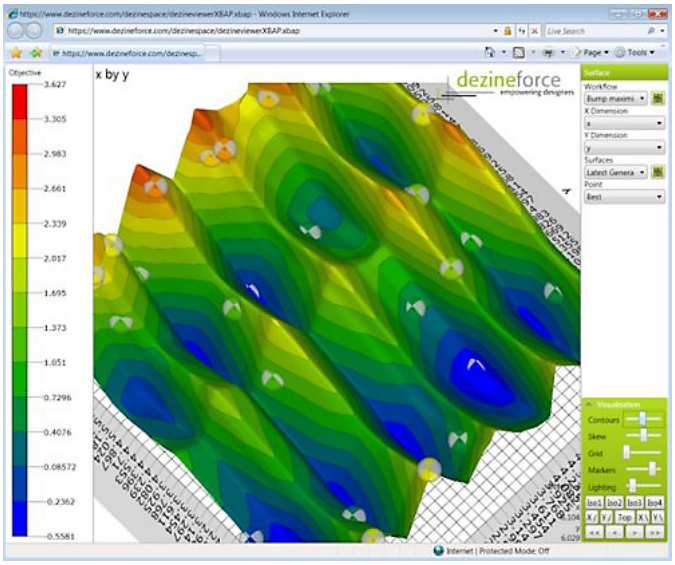


Figure 6: Surrogate model accurately predicting peaks and troughs with very little information



the process orchestration to focus new simulations where the better designs are to be found—in this case, near the lowest point.

In the second example, which is shown in Figure 6, we use a complex mathematical function to provide a more challenging problem. Note how few simulations (colored spheres) have been carried out; yet the existence of peaks and troughs is predicted with remarkable accuracy.

This enables engineers to gain insights into their designs, with far fewer resource-intensive simulations, and find high-quality non-intuitive and counterintuitive designs. This results in better products, with reduced costs and reduced time to market.

Engineering Process Orchestration

An engineering process can be as simple as a single simulation. More likely, however, it is made up of several different types of computation in a defined sequence, which may have some form of parallelism. The data from one simulation will often have to be passed to the next simulation and possibly modified in some way. A more complex systematic exploration will involve multiple loops that have a wide degree of parallelism, which results in many threads of execution.

The architecture must be able to coordinate and monitor these parallel computations, reacting to their state changes in an appropriate manner. It must ensure that each computation executes within its own private area, within the correct security context. It must scale to support a large number of simultaneous processes in a way that is fair to its multiple tenants and provides a good user experience.

Figure 7 shows a high-level schematic of a simple optimization workflow.

This level of automation can greatly reduce the amount of engineer time that is required to set up and launch the simulations—reducing costs, increasing productivity, and reducing time to market. It is also a significant step towards compliance, because it ensures a consistent, well-defined, and repeatable process that is fully audited.

Within the dezineforce implementation, Windows Workflow Foundation (WF) forms the basis of process orchestration. Its ability to support episodic execution—and thereby load and unload workflows dynamically from memory—allows a large number of long-running processes to be serviced with relatively little overhead. This holds true, even when an individual workflow has several hundred parallel branches running concurrently. It also has many extensibility points that have allowed it to be heavily customized to meet requirements.

Engineering Computation

The architecture must provide powerful engineering tools and high-performance and high-throughput computing. This reduces the capital expenditure of engineering companies in terms of both software licenses and hardware. It also gives them access to a wider variety of applications, which allows them to take on new engineering challenges and open up new revenue streams. The ability to scale on demand also improves their agility.

Figure 7: High-level schematic of simple optimization workflow

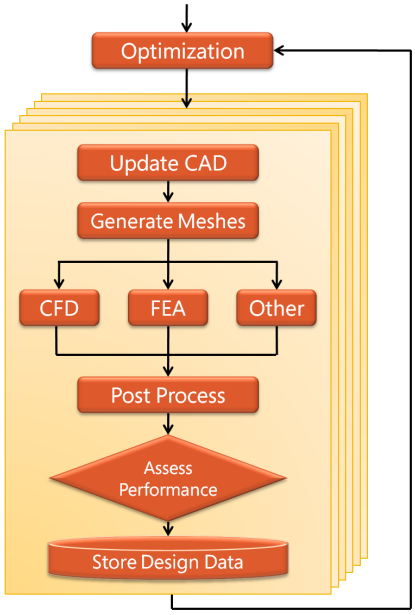


Table 1: Mapping of architectural challenges to customer needs

	Cost reduction	New revenue streams	Productivity	Agility	Time to market	Better products	Compliance
Engineer interaction			■		■		
Engineering intelligence	■	■			■	■	
Engineering process orchestration	■		■		■		■
Engineering computation	■	■		■			
Long-term, large-scale data management			■		■		■

Given that an individual engineering process can execute a large number of simulations in parallel and that many engineering processes can run concurrently, there is a need for the architecture to coordinate the use of scarce resources. These are typically software licenses and hardware.

Hardware scheduling ensures that the compute nodes that are used to execute the engineering calculations are loaded efficiently. This is typically carried out by a job scheduler that distributes load across a single compute cluster. License scheduling ensures that software licenses are used effectively across all compute clusters.

Process orchestration must work with the hardware scheduling and license scheduling to submit the right jobs at the right time to the right cluster, based on current load conditions and license availability. It should also take into account job priorities and fairness.

To work efficiently, the architecture must support two forms of multi-tenancy. Multi-tenancy is often taken to mean supporting multiple organization or tenants on the same physical infrastructure. Although this is true for engineering Software + Services, it is also highly desirable to run different combinations and versions of engineering applications on the same compute nodes at the same time for different tenants.

Each engineering application is different and reports errors in different ways: some through exit codes, some through output streams, and some by just freezing. The architecture must be flexible enough to detect errors in a variety of different ways. It must know how to terminate a rogue application cleanly and restart an application cleanly when the error condition indicates that a restart is feasible.

Within the dezineforce implementation, a license allocation, reservation, and revocation subsystem has been created as a custom application. Microsoft Windows HPC Server is used to control the scheduling of jobs onto the compute nodes. Its extensibility model has allowed it to be customized to interact with the other subsystems, so as to close the loop of process orchestration, license management, and job management. The use of heterogeneous compute nodes also has been shown to be possible in this architecture.

Long-Term, Large-Scale Data Management

As engineering processes generate very large data sets rapidly, the architecture must provide high-speed access to large storage areas.

In many industries, engineers are required to keep design and simulation data for the lifetime of the products. For aviation, this is often several decades. Therefore, the architecture should allow data to be kept for extended periods of time to support compliance. If this capability is provided, checks must be in place to ensure that data has not been corrupted—with non-repudiation to ensure that changes to data are attributable.

It is common practice to use a previous design, possibly many years later, as the starting point for a new design. As well as locating the original design it must be possible for the new engineer to rapidly understand how the original design was reached, the different variants that have already been explored and how the existing design performs. This improves engineer productivity and reduces time to market.

Although a UI must provide lightweight methods to explore the data without bringing it down to the client, a subset of the data often will need to be downloaded, especially if long-term data storage is not provided. The architecture must provide for high-performance, secure, and resumable downloads for very large data sets. This might be required to support compliance.

Within the dezineforce implementation, storage technology has been used to provide large-scale, high-speed file access with very high availability. It also implements local data mirroring and data mirroring between different sites to support compliance and high availability.

Microsoft SQL Server is used to store non-file–based data such as the execution history and the underlying values that are used by the optimization process to support rich querying of the data.

Security

Within the dezineforce implementation, general authentication is based on Microsoft Active Directory, with network-level security implemented within the switch configurations. Role-level authorization is then used to distinguish between different levels of user and their permitted actions.

Security on the compute cluster is more complex to implement. Many of the engineering applications execute by using journals (a proprietary form of script) that are capable of launching arbitrary code; therefore, it is especially important that each job be scheduled to run within a security context that has a limited sandbox. Because jobs can be submitted deep within the architecture, long after the user has left the site, it is not possible to use delegation in the normal sense.

Management

Within the dezineforce implementation, systems-center operations management is used to monitor all aspects of the system in conjunction with external services that monitor overall system availability.

Microsoft Windows Compute Cluster Server (today superseded by Microsoft Windows HPC Server 2008) provides its own management console, which is used to manage the compute clusters and the compute nodes within them.

Impact on Engineering Companies

By looking at our original customer aims of cutting costs, developing new revenue streams, increasing productivity, increasing agility, reducing time to market, creating better products, and adhering to compliance—and relating this to our five key architectural challenges—we can see a strong correlation between them, as Table 1 shows.

The availability of engineering computation/simulation in the Cloud can reduce costs, open up new revenue streams (by providing access to capabilities that are normally beyond the reach of an organization), and provide a level of agility through the ability to ramp-up on demand. However, it is the orchestration of these computations, combined with effective engineer interaction, that will provide the real productivity improvements and significantly reduce time to market.

Engineering intelligence further improves efficiency by focusing the computational effort on the areas that are likely to produce good designs. It is also an effective route to finding high-quality non-intuitive or counterintuitive designs. Importantly, this can be achieved while keeping the engineer in control of the process.

A recent study by Catalyzt analyzed the dezineforce solution and compared the costs against conventional design approaches, and across a wide range of engineering-design activities. This analysis covers different levels of design complexity, types of design "solvers," and areas of engineering design. Catalyzt concluded that "well-designed, well-executed SaaS services, such as the dezineforce service, can dramatically cut the costs of engineering design."

The full details of the cost comparison are available from Catalyzt.com. The headline results are included in the following list, courtesy of Cartezia.

The following were key areas of the examined and compared costs:

- IT costs, including:
  - Analysis
  - Application licensing
  - Computing hardware
  - System management and support
  - Setup (procurement and commissioning)
- Designer time:
  - Analysis setup and postprocessing
  - Modeling
  - Assessing scope for design improvement
  - Making design decisions
- Risk management:
  - Late-in-cycle changes
  - Recalls/warranty claims/penalties

The cost-comparison curves that are reproduced in Figures 8–10 dramatically illustrate the cost advantage of the dezineforce offering over conventional design approaches. To this cost advantage should

Figure 8: Component-level cost comparison

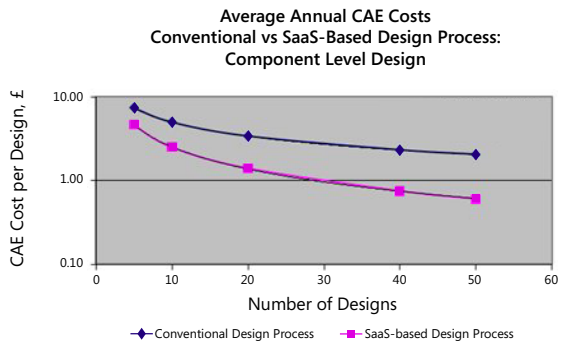


Figure 9: Subsystem-level cost comparison

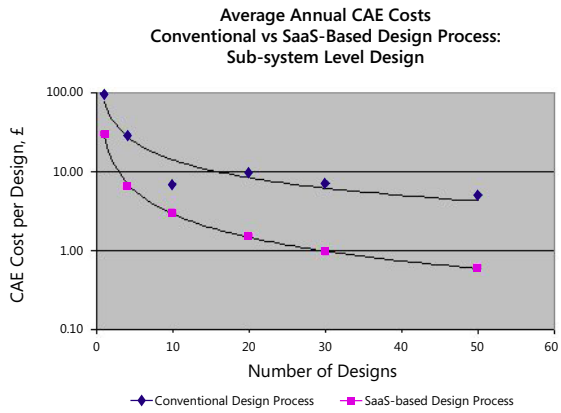
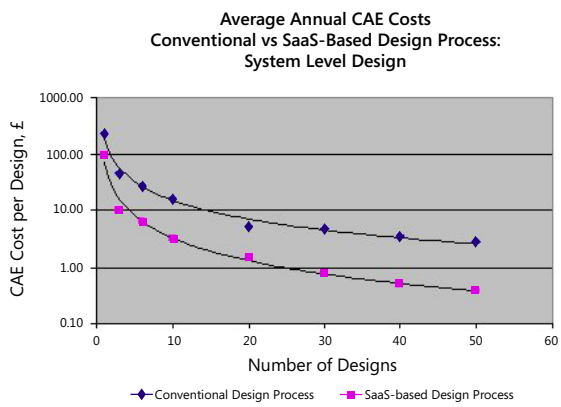


Figure 10: System-level cost comparison



be added the additional benefits of significantly enhanced design optimization (giving better designs) and the design flexibility that is enabled by use of a subscription-based service—with multiple tiers of subscription usage coupled with the ability to buy "top-up" design capability.

A move to this new model is also a shift from capital expenditure to operational expenditure, which is particularly significant in turbulent times.

By looking at current trends, it is clear that globalization is increasing within engineering, with a desire for design teams to span multiple geographies. The provision of globally accessible, centralized computation and storage addresses many of the challenges that engineering companies face when they work globally. It allows all members of the team to run simulations, without having to provide extensive computation facilities at each site.

The ability for engineers to search and view results without having to download the full data set significantly reduces the need to move large data sets around the globe—removing the time delays and network infrastructure requirements that are inherent in doing so. The net result is that engineers in diverse geographies can both follow the progress of and control the design exploration, in near–real time, over standard Internet connections. This has the potential to support a follow-the-sun model, as offices in different time zones come online.

The instant availability and scalability of a Cloud-based service—compared to the long lead times that are involved in procuring and commissioning a dedicated engineering-computation facility—should also not be overlooked.

Conclusion

We have defined a class of service called "Engineering in the Cloud" based on the real-world needs of engineering companies:

> *"Engineering in the cloud is a combination of cloud services and rich interactive applications that provides integrated, intelligent, self-service engineering services over and above engineering-application hosting and computation—allowing engineers to create, explore, and discover better designs faster."*

We have shown that the hosting of engineering computations on its own does not satisfy this requirement, and we have put forward an architecture, called "Engineering Software + Services," that can do so.

We have then used dezineforce.com as a case study to show how this architecture can—and has been—effectively created. During the case study, we walked through the key features of the architecture and described the technologies that are used to implement each of them.

We have shown how "Engineering in the Cloud," implemented as an engineering Software + Services architecture, can affect a step change in engineering and is a reality today.

Resources

The following resources provide further information on the design-search and optimization techniques that are used within the dezineforce service and, more generally, within engineering:

www.dezineforce.com

De Souza, A., R. Harding, A.J. Keane, and S.J. Cox. "Integrated Design, Search, and Optimization for All." NAFEMS World Congress, June 2009.

Forrester, Alexander, András Sóbester, and A.J. Keane. *Engineering Design via Surrogate Modelling: A Practical Guide*. Chichester, West Sussex, England: John Wiley & Sons Ltd., 2008.

Keane, A.J., and P.B. Nair. *Computational Approaches for Aerospace Design: The Pursuit of Excellence*. Chichester, England; Hoboken, NJ: John Wiley & Sons Ltd., 2005.

Keane, A.J., and J.P. Scanlan. "Design Search and Optimization in Aerospace Engineering." *Philosophical Transactions of the Royal Society A (Phil. Trans. A)*. Volume 365, No 1859, 2007, 2501–2529.

About the Authors

Peter Williams (peter.williams@dezineforce.com) is Chief Technology Officer of dezineforce. A Microsoft Certified Architect, he is responsible for the architecture of the dezineforce "Engineering in the Cloud" implementation and for ensuring that the technology strategy supports the business strategy. Peter joined dezineforce from Microsoft Corporation, where he held a number of technical and managerial roles; more recently, he held the position of Senior Solutions Architect within the Microsoft U.K. professional services organization, where he led the development of numerous enterprise-scale solutions. Previously, he was group manager for software and infrastructure development within the Microsoft U.K. solution-development center and spent a number of years in Redmond, WA, working in the product groups.

Simon Cox (simon.cox@dezineforce.com and sjc@soton.ac.uk) is Professor of Computational Methods in the Computational Engineering Design Research Group within the School of Engineering Sciences of the University of Southampton, as well as Chief Scientist at dezineforce. An MVP Award holder, he directs the Microsoft Institute for High Performance Computing at the University of Southampton and has published over 120 papers. Currently, Simon heads a team that applies and develops computing in collaborative interdisciplinary computational science and engineering projects, such as computational electromagnetics, earth system modeling, applied computational algorithms, and distributed service-oriented computing for engineering.

Follow up on this topic

- High Performance Computing: Windows HPC Server 2008
- Services in the Cloud: Microsoft Azure
- UI Technologies: Windows Presentation Foundation
- Rich Internet Applications: Microsoft Silverlight

Platform
Architecture Team

Microsoft

20

098-11048#