

THE ARCHITECTURE- CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

PRINTED: WEDNESDAY, NOVEMBER 09, 2005

© HILBERT COMPUTING, INC., 2005

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

TABLE OF CONTENTS

EXECUTIVE OVERVIEW.....	1
ENTERPRISE ARCHITECTURE.....	2
Pontificating from the Ivory Tower.....	3
A Structure for Negotiation.....	4
Motivating Technical People.....	5
PHYSICAL ARCHITECTURE.....	6
Vendor Lock-In.....	6
Products as Architecture.....	7
NETWORK ARCHITECTURE.....	8
Affect on Other Architectures.....	8
DATA ARCHITECTURE.....	12
Affect on Other Architectures.....	12
SECURITY ARCHITECTURE.....	14
Application-Specific Security.....	14
Affect on Other Architectures.....	15
OPERATIONAL ARCHITECTURE.....	17
Deployment Issues.....	17
Evolution of Diagnostics.....	18

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

Outsourcing.....	19
Decision Makers and Operational Issues.....	20
Affect on Other Architectures.....	20
APPLICATION ARCHITECTURE.....	22
Purchased Code.....	24
Affect on Other Architectures.....	25
THE VISION.....	26
Role of the CIO.....	27
Roles of the Architects.....	27
CRITICAL SUCCESS FACTORS.....	29
CONCLUSION.....	31
AUTHOR INFORMATION.....	32

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

EXECUTIVE OVERVIEW

Information Technology has never been more complex or more difficult and expensive to manage. Fortunately, as the complexity has increased, so have the skills to manage complexity. Most companies with which I have consulted have not effectively implemented these leading-edge IT skills to lower the cost of supporting their information infrastructure.

The key to managing complexity is to take an *architecture-centric* approach to managing IT infrastructure. The architecture describes the structural shape of the software and hardware services that support IT systems. It also describes the mechanisms and processes by which experts in IT areas have the opportunity to have voice and negotiate the technical merits of approaches in their specialized areas.

A very small amount of the enterprise architecture addresses the implementation technologies, vendors or products. An effective architecture empowers companies to more readily shift implementation technologies as the marketplace changes or adopt new technologies as they become available in the market. An effective architecture reduces the dependency on specific vendor solutions and thus increases the enterprise's ability to negotiate more cost-effective contracts with vendors. An effective architecture that includes the needs of the operational environment can enable companies to more readily detect, diagnose and repair outages in the IT infrastructure.

The architecture affects human resources as well. Not all IT professionals have the same skills. An effective architecture can substantially help leverage the skills of the more advanced IT staff so those skills can be exploited by less experienced staff. As a result, your organization gets more productivity out of your internal and outsourced human capital.

Very few of the companies with which I have consulted have the discipline to implement all facets of this approach. That's because this is difficult. This isn't a "silver bullet" for IT problems, but it does have the potential – with hard work, discipline and placing the right people in the right place – to solve the most difficult IT problems. The companies that have taken this approach are empowered in the marketplace because of their increased productivity and lower operating costs.

This document will cover what defines an architecture-centric enterprise, how to implement that approach, and the benefits in doing so.

ENTERPRISE ARCHITECTURE

What is an enterprise architecture? The architecture defines the *structure and shape* of the elements in the information technology infrastructure. In fact, the enterprise architecture isn't a single thing, but includes at least six distinct architectures, each with their own complexities. The six key architectures that will be considered in this document are:

- **Physical Architecture** – The physical architecture contains the specific vendor products and implementation technologies that have been deployed in the enterprise. This is the most tangible and tactical of the architectures and for many enterprises, it is the only recognized architecture.
- **Network Architecture** – This defines the structure of the network resources, including routers, redundancy, network resource addressing, address translation, etc.
- **Data Architecture** – This covers the organization of, and access to data resources. This includes the strategy to gather, verify and enable access to operational and decision support data.
- **Security Architecture** – This covers the approach to identify human and non-human users of information technology resources, enable access to those resources and audit access to those resources.
- **Operational Architecture**. This involves the structural elements that are required to support hardware and software once they are deployed. This involves process scheduling, diagnostics, problem resolution, capacity planning and deployment planning.
- **Application Architecture** – The application architecture defines how internal applications are structured when they are developed. For purchased applications, the analysis of the application architecture will define how the application will fit into the existing IT infrastructure and expose the hidden costs in supporting and integrating the application.

The Enterprise Architect analyzes the ways in which these architectural elements interact with one another and the affect that decisions have on the enterprise. In my experience, most organizations tend to focus on one or two of these areas to the detriment of the others. This may be due to individual personalities or reasons that have to do with corporate culture and history. Each technical domain listed above should have an architect that is responsible for that domain. In smaller organizations, a single person may cover multi-

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

ple architectural roles. When issues that arise in areas for which they don't have a strong technical background, they can use technical leaders within the organization or seek help through an outside organization with experience in a particular technical discipline.

The effective enterprise architecture has an appreciation for all of these architectural areas and the affect that decisions in one area have on other areas. Quite often, the ideal choice for one architecture has an affect on another architecture that is suboptimal. The right choice for the enterprise may be a compromise approach.

We will look at each of these architectures, the benefits of taking an architectural approach to managing these resources, and the risk in avoiding the architectural issues surrounding these areas.

PONTIFICATING FROM THE IVORY TOWER

Many organizations may have tried to bring an architectural focus to their IT systems in the past. Most commonly, a group of skilled professionals would work on strategic plans that would be used to modernize the infrastructure. I was in such a group in a large corporation. The focus was on creating what we called the “target state architecture.” This was primarily a physical architecture, but included some aspects of application and security architectures.

That organization had a very significant shortcoming that prevented it from being useful to the organization. This architecture group was not involved in the day-to-day development and operational environments. As a result, they didn't have an appreciation for the real problems that the architecture should solve. That group also omitted anything relating to a migration plan. They were purely a “think tank” organization. I bailed out after a year to go back into the operational realm. The group was disbanded about six months after that.

The lesson to be learned is that architecture has to deal with the current environment as well as the future vision. The architecture is embodied in the day-to-day activities of the IT organization. It is not a visionary group. It is a pragmatic one.

A STRUCTURE FOR NEGOTIATION

The architecture isn't just about technology choices and concepts. It's about people. When there are areas of complexity that overlap, as in the case of an enterprise architecture, there will be conflicts of interest. The architecture-centric enterprise creates a process by which technical conflicts are recognized, evaluated and mediated.

A key goal for the Enterprise Architect is to recognize the presence of technical dissonance and know how to ask the right questions of the architects in the technical areas experiencing a difference of opinion. The architects have an obligation to the organization to articulate the technical merits of what they want to do in a manner that is understandable by technically competent people who may not be well versed in a particular discipline.

My brother is a skilled data architect. He designed a data mart that had one aspect that made no sense to me. I have an operational, security and applications architecture background but am not particularly strong in data. He created a completely denormalized database table that didn't have a particularly intuitive object-to-relational mapping that I could identify in the business problem domain. I challenged him to explain the approach. I understood perhaps eighty percent of the explanation, but that was sufficient for me to have a general grasp of why the approach was important¹. In the end I agreed that it was best to take on the additional programming burden to implement this particular aspect of the data architecture.

There is a lot of nuance in that story that needs to be explicitly called out. First, we really listened to each other because the motivation was in doing the right thing, not doing it "my way." We eliminated the resentment that comes with a "just shut up and do it" approach that can occur when there is a dominant group or personality that is affecting the decisions made for all of IT. There is a lot of trust and respect that was built over a long period of time by people who are motivated by doing the right thing for the organization with respect to their technical domain.

By working through the issues that this aspect presented for each side, we both learned something. When done right, an architecture-centric approach to enterprise IT can serve as a powerful structure to cross-educate all IT personnel in the complexities inherent in the various technical domains.

¹ The bottom line was that it was a performance-related approach that avoided joining the largest of the tables in the datamart.

MOTIVATING TECHNICAL PEOPLE

People will work hard to achieve goals that they believe in. That is especially true of technical people. IT can be an exceptionally frustrating endeavor. There is a lot of complexity that requires an unwaivering attention to detail. Most of the technicians with whom I work do it because they love the work. They become less motivated when an inordinate amount of time is spent on processes and procedures that don't relate to solving the IT problems in their skill area.

While the asocial nature of IT professionals is a cultural stereotype, my experience suggests that is not entirely an undeserved stereotype. A large percentage of technical people would rather be left alone to do their job than spend a lot of time in meetings discussing what should be done and filling out paperwork related to project management and justifying the approaches and time needed to solve problems.

The architecture-centric approach can serve those in the trenches in two significant ways. First, this architectural approach provides the process by which innovative technical ideas and technical barriers to success have a forum to be heard, articulated and acted upon by fellow technicians. Every motivated technician wants to have a voice in the technical decisions that affect them. Second, this approach provides the structure through which those issues are can be handled by someone else, so they can do their job. The architect is there to arbitrate the issues without the need to drag the team into a set of meetings that require them to keep articulating the same technical issues to a group of people with mixed technical backgrounds. Effective negotiation is a key role of the architects within each technical domain.

It is vital that each department within IT understands that the architectural structure provides a forum for technical issues. If the implementation is such that there is an "architectural review board," then the implementation will be perceived as another burdensome process that must be overcome instead of being the advocate that will help them effect the right technical changes within the organization.

PHYSICAL ARCHITECTURE

The physical architecture defines the products and implementation technologies for the information technology resources in the enterprise. It includes the specific vendor choices for operating systems, programming languages, routers, network topologies and other products. It also includes implementation technologies, such as using relational or OLAP technology for data access, Kerberos for authentication and TCP/IP for internet-working.

This is the most tangible and the most tactical of the architectures. For many organizations, this is the *only* architecture. At first glance, this seems to encompass all aspects of the enterprise information technology, but it only encompasses the tangible parts. Enterprises that limit themselves to this view of architecture tend to be vendor-centric. The traditional “IBM shop” or “Microsoft shop” essentially delegates all architectural decisions to their small subset of chosen vendors.

There are many examples of successful organizations that have taken this approach. It greatly simplifies the decision-making process. For companies that don't have deeply technical staff skills, this may be the best option. For larger enterprises, or for those who value information technology as a way to gain competitive advantage, this vendor-centric approach can be very costly and very limiting.

VENDOR LOCK-IN

Regardless of how altruistic vendors appear to be, at best their loyalties are divided between offering the best solutions to their customers (you!) and pushing the products that best fit their business needs. Vendors thrive not only when they can sell their products, but when they can create an infrastructure that makes it difficult for you to purchase competing products. As vendor lock-in increases, a company's ability to negotiate reduced pricing on software becomes compromised because vendors are aware of the same thing that you are – a technology change out would be too costly and disruptive.

Perhaps more important, if an enterprise becomes substantially dependent only on the solutions available from a small subset of vendors, they are cut off from innovations happening in other areas of the industry. For enterprises that use information technology for competitive advantage, this can be especially harmful. Enterprises that view information technology as a commodity will have less of a negative consequence of taking a vendor-specific approach.

PRODUCTS AS ARCHITECTURE

In my experience, the companies with the most chaotic, expensive and fragile information technology environments are those that substitute best-of-breed product selection for a coherent enterprise architecture. This happens quite often in enterprises that enable the business customers to implicitly make architectural decisions by selecting software applications that fit their business needs, but for which there is no appreciation to the impact of that software when it is introduced into the IT infrastructure.

We have all seen this happen. A company's IT department makes a commitment to a couple of specific vendors, say Oracle and DB2 for their database technologies, and a vendor application requires the use of Microsoft SQL Server. While the application may have the best fit with the features required by the business unit, the costs of introducing a new vendor solutions for a significant product, such as a database engine, can be extensive for IT.

It can also happen within IT itself. One group with influence within IT may select a vendor product, such as Microsoft Office, and in doing so implicitly determine the desktop operating system choice for the enterprise as Microsoft Windows or Apple Macintosh, since those are the only two supported operating systems that support Office. This may end up being the appropriate solution for the enterprise, but only by coincidence and not through valid architectural analysis.

The architecture-centric enterprise may not prevent these scenarios from happening, but they will be able to quantify the costs to large degree and empower the CIO to negotiate with the business executives about the true costs. Depending on internal accounting practices, this may result in a charge back to the business unit to assist in covering the additional support costs. It will also empower the CIO with information if the issue needs to be escalated to the next level of management.

NETWORK ARCHITECTURE

The network architecture defines the following elements of the enterprise networking infrastructure:

- **Networking protocols.** For many enterprises, these are the IP protocols including TCP/IP and UDP/IP. For enterprises with a large mainframe presence, there may also be support for IBM's SNA networking protocols.
- **Physical layer.** This includes the cabling specifications, such as CAT5 or CAT6, which determine the maximum supported data rates. This also includes the approach to supporting wireless internetworking with technologies such as 802.11, GPRS and EV-DO. This may also include working with wireless service companies for the implementation of in-building systems.
- **Topology.** The topology determines the pattern of links connecting nodes in a network. The network architect determines how the network elements are interconnected to provide the maximum amount of network availability through redundant paths while keeping cost and complexity under control.
- **Network management.** The network elements, such as routers, hubs and computers, should be managed for availability and capacity. Network management is traditionally based on SNMP protocols. Infrastructure vendors have management software available to effectively manage the availability of network resources.

Most modern applications require network availability to function. Even batch applications often use internetworking to access critical resources such as database engines.

AFFECT ON OTHER ARCHITECTURES

Internetworking technologies, especially those based on TCP/IP, are the most open and independent of the other technologies. There are some areas of overlap that need to be addressed, however.

Any network architecture will have an element of security involved. Even the simplest network implementation will have one or more firewalls. A firewall will block access to many of the network ports. This can have an effect on the application architecture. If

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

the firewall prevents access except through port 80² then the application can not be designed to use other internetworking protocols. If the network architecture states the exclusive use of SSL³, then this can affect how application code is developed. An effective application architecture will mitigate the effect of these types of network architecture decisions, but there are issues that need to be handled.

For all relatively complex applications, there should be an architectural analysis that overlays the application components on the deployment model⁴ for the network architecture and analyzes the security implications. That's a pretty esoteric statement, so I will describe what I mean by looking at a real-world example with one of my customers. An application was designed to have a web-based user interface layer. Since the application was a public-facing application, it was deployed on a DMZ⁵ machine. Machines in the DMZ are considered to be more easily exploited than a machine in the internal, trusted network. The functional requirements of the application required audit logging of information pertaining to credit card transactions. That information was being logged in the local filesystem. That is a perfectly reasonable implementation from an applications point-of-view. However, that sensitive information was being written to a DMZ machine that was considered insecure, so this was unacceptable from a network and security point-of-view.

An application architecture review that encompassed an understanding of the network architecture, operational architecture with regard to deployment, and the requirements within the security architecture would have exposed the potential security flaw. This is a fairly complex confluence of different architectural skills and decisions and illustrates the need for an understanding and appreciation of the value of an enterprise architecture with architects skilled in the various specialty areas. Many of the data theft events become a public embarrassment when exposed in the media. With 20/20 hindsight, many security compromises appear to have been easily resolved, but they won't be exposed without a process that includes this type of architectural analysis.

In my experience, when these events occur, the enterprise will attempt to put a process in place that prevents a future occurrence of the security breach. While that may be effective at preventing a repeat of that exact same scenario, it won't necessarily prevent a sce-

-
- 2 Port 80 is the most common port for access to services via web browsers using an HTTP protocol.
 - 3 SSL is being replaced with TLS implementations, but most people still refer to secure TCP/IP sockets as SSL. SSL is the basis of the secure HTTP known as HTTPS.
 - 4 The deployment model is part of the operational architecture. This analyzes where part of the application reside within the physical architecture
 - 5 The term "DMZ" literally stands for demilitarized zone. In networking parlance, this is a machine that is exposed to the global internet with limited ability to interact with machines inside a trusted networking zone

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

nario that compromises information in a different manner. An enterprise that takes an architecture-centric approach to managing information technology resources has a much smaller chance of having these kinds of security compromises in the first place, since a focus on architecture helps ensure the appropriate use of IT resources in a context that is removed from merely viewing IT in terms of business functionality.

Some network architectures may involve the use of authentication at the network level, especially for access through a VPN. In most organizations, this involves the deployment and support of an identity management infrastructure that is distinct from identity management for applications or operating system access. Ideally, the security architecture would encompass the needs of the network access security and enable applications developed internally to use that information for identity management.

There is also overlap between the operational architecture, physical architecture and the network architecture with regard to management software. The routers, hubs and systems that are to be managed must contain the firmware and software to report events and statistics to the management software. Ideally, the network management software interoperates with the other operational management software for logging and diagnostic information coming from the operating systems, services such as servlet engines, and the applications running in the infrastructure. The *integration* of information can assist the operational staff in quicker diagnostics and more rapid problem resolution. The ideal management software handles event correlation so that network outages are understood in the context of service and application outages. That is, when there is a network outage with some part of the network, the operational staff understands that certain applications will be effected. They don't have to engage in diagnosing application failures since the problem source in this example was already identified as the network. Few organizations have this type of operational diagnostic capability because it is, quite frankly, difficult to achieve. There are definite benefits to moving to achieve this level of functionality. Customers are significantly more happy with their Information Technology department when outages are kept to a minimum. Staff resources are best utilized when the problem source can be identified quickly so the appropriate technical staff can be brought to bear on a solution. Too often, I have seen technical groups "finger point" as to the root cause of problems due to a lack of concrete analytical data.

Just as quality information is vital to executives in their decision support systems, quality information is vital to operational staff when trying to diagnose and optimally avoid problems within the infrastructure. You will have to pardon my soapbox here, but after spending well over a decade in an operational capacity in multiple large enterprises, it is frustrating to see that executive management is focused on infrastructure only in the midst of a crisis. When the crisis is over, there is little funding or attention given to an approach that tries to avoid a crisis in the first place. An effective architecture-centric

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

approach to managing information technology resources can do just that. Okay, end of that particular soapbox.

DATA ARCHITECTURE

The data architecture encompasses the framework for the implementation and planning of data resources. This involves the logical and physical representation of data within the enterprise. The data architecture defines the use of different data access technologies such as relational databases, OLAP data cubes or LDAP data stores. It will also define the use of organizational techniques, such as star schema, which enable multi-dimensional database functionality using a relational database technology.

The data architecture also involves the management of metadata. Metadata is data about data. It is typically managed as a repository of definitions for data elements so business users have a common terminology for business data. It also contains information about how timely the information is⁶, who owns the data, and the systems from which the data was obtained or derived.

AFFECT ON OTHER ARCHITECTURES

Obviously, data is core to enterprise information technology. Essentially all of the business functionality is based on the management, processing and rendering of data. In my experience, perhaps with the exception of the physical architecture, a focus on the data architecture tends to disproportionately dominate the other architectures. That is understandable given the visibility to the business, but a truly effective enterprise architecture recognizes and accommodates the needs of all of the architectural domains.

The data architecture has the greatest effect on the application architecture. Applications tend to be focused on populating software business objects or rendering report information with the persistent data found in the enterprise data stores. There is a natural mismatch between relational database technologies and object-oriented development that dominates most application architectures. Relational data technology is based on a tabular view of data and the relationships between the elements of data. Object-oriented designers and developers view information in terms of hierarchical relationships through the class hierarchy and though containment in the object composition model. This creates an understandable conflict between the applications development staff and the data administration staff.

The greatest risk in this overlap exists when the influence of the application development group exceeds that of the data group. To put it another way, the object model should

6 Referred in the data literature as “currency”.

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

adapt to the data model. The object model should *not* drive the data model. The data architecture deals with the management of the total lifecycle and effective use of data from both the perspective of an operational data store and a decision support data store. Simplifying the mapping of relational data to an object model is a very limited perspective on the use of enterprise data.

Prior to my coming on board, one of my customers had an applications group that tended to dominate the data group. They decided to use an early, but then current, specification of container managed persistence (CMP) in Enterprise JavaBeans (EJB). In this implementation, the object model dictated the relational data model since that specification had essentially no ability to map in a more flexible manner. They even went so far as to disband their group of data analysts because there was nothing for them to do. While that example is a case of the logical extreme, it illustrates the need for an enterprise architecture that has built-in checks and balances between the different perspectives on information technology.

Relational database technology is implemented with its own security model. There are user accounts and passwords that control access to the data in the database. Access is granted to users in order to create, delete, view or update the data. Most web-based applications will share a common user id to access the relational datastores used by their application. The application will then authenticate a user with more granular and controlled access to application resources. The user authentication technology is often a different implementation technology⁷ than the security that is integrated in the database engine. This mismatch has an effect on both the security architecture and the application architecture. The security architecture needs to accommodate the requirements of these disparate security implementations. The application architecture should ensure that these issues are managed by architectural elements outside the application *per se*, so that application developers don't need to be distracted from their pursuit to implement business functionality by these infrastructure issues.

Enterprises typically have a need to audit data access as part of the security architecture. The auditing tools available to the relational database engine are going to report on the common, shared user id. Typically, the enterprise needs to audit data access according to the more granular user that was authenticated within the application. It is necessary for the application architecture to accommodate the correlation of data access as viewed by the database engine with the security architecture as represented in the application architecture. Again, the part of the application architecture that implements security requirements should be part of a reusable security framework so that applications developers are not distracted from their primary goal of implementing business functionality.

7 Kerberos and LDAP are common security implementation technologies

SECURITY ARCHITECTURE

The security architecture defines the structure of software and hardware services that protect information technology resources. Application security centers around three broad areas: identity management⁸, authorization and auditing. Identity management is used to determine who⁹ is requesting access to resources and ensuring the authenticity of the assertion of identity. We are most familiar with identity management in the form of userid and password credentials that we enter to gain access to systems or applications. Authorization is the process by which we determine if the authenticated identity should be permitted access to resources. This are usually represented in the form of permissions. That is, a user will have permission to view a file or update data in a database. Most enterprises also have a requirement to audit access to secured resources.

The security architecture also needs to include network security. The use of firewalls will prevent access to elements in the network. There is also a need for intrusion detection. If the firewall is breached, the enterprise needs to be aware so that the scope of the breach can be assessed and remediated.

Security is complex when dealing with a single enterprise. With the increase in the inter-enterprise communication of information through web services, the complexity of effectively managing secure access to resources increases dramatically.

APPLICATION-SPECIFIC SECURITY

Most organizations don't have a comprehensive security architecture, so each application is responsible for implementing security that meets the requirements obtained (or more likely inferred) from the business community.

The first problem that this approach causes is a diversity of security data stores. When a user leaves the company, there are several security stores from which they should be removed. Frequently some of these are missed and there is an active account that can be used to compromise the system. There also tends to be a semantic disparity in the security stores. A group of "staff" in the payroll system may have a different meaning than "staff" in the accounts payable system. This inconsistency in the use of terminology can lead to human error when configuring permissions.

⁸ Also called authentication

⁹ "who" can be a human user or a software service

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

The second problem is that few application developers understand the nuance of implementing security. In order to be resilient to change and effectively administered, the security configuration should be external to the application program. That is, the authentication and assembly of permissions should be delegated to an external service. Most programmers approach security by looking at the groups in which the authenticated userid is a member and programmatically block sections of code based on group membership. Rarely is security auditing a part of application-specific security unless it is explicitly called out by the customers during requirements gathering.

Perhaps the greatest issue with application security is the inability to respond to change. There has been recent legislation, such as HIPPA and SOX¹⁰, that imposes legal requirements on the security of certain transactions and data. It is important that enterprises be able to quickly and cost-effectively comply with changes in the legal landscape. There has been an increase in the enterprise-to-enterprise transfer of information, due in a large part to the emergence of web services. This adds considerable complexity to enterprise security. An architectural approach to security is essential to being resilient to change and managing these emerging complexities.

AFFECT ON OTHER ARCHITECTURES

The security architecture most certainly has an effect on the application architecture. The applications need the services described by the security architecture so credentials can be authenticated and access to resources can be managed. It is essential that the application use external services for authentication, authorization and auditing so that the business is able to change implementation technologies or integrate additional features, such as enterprise-to-enterprise security.

Security is complex. Since we want the applications developers to be focused on application functionality, the best approach is to have the essential elements of the security architecture implemented with a framework that applications developers can use, but one that is decoupled from the application itself.

The security architecture has another effect on the application architecture from a networking perspective. If the application is a public, web-based application, then the machine on which it resides will most likely be in a DMZ. Key network ports, such as those to access internal databases, may not be available. Recording of sensitive information to local, persistent media in a DMZ machine is often forbidden by the security rules of the organization. This will require the application architecture to be able to support

¹⁰ SOX is also known as the Sarbanes-Oxley bill.

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

recording this information on remote, more secure systems in a secure way. There may also be requirements in the use of secure sockets (TLS or SSL) between systems. If the application is managing their own network resources, instead of relying on container services such as those in a servlet container, then the architecture should support the use of secure sockets and the management of the requisite PKI¹¹ resources.

The security architecture has an effect on the network architecture as well, especially if access to the network requires authentication. It may be useful to access the security credentials used to access the network as authentication credentials for an application. It may also be a security requirement that users coming from external networks have limited access permissions compared to access from internal networks. Since network authentication will often have a different security implementation than operating system or application security, it becomes the responsibility of the architects in these two areas to determine an appropriate solution for the enterprise.

¹¹ “PKI” stands for “public key infrastructure” and typically refers to the issue and management of X.509 security certificates.

OPERATIONAL ARCHITECTURE

The operational architecture consists of the elements in the enterprise that are used to support the IT infrastructure once it is deployed in production. This includes resources that can be used to determine the health of the application. The architecture describes the structural elements that enable the production of information at times of failure so that the operational staff have the diagnostics they need to resolve the situation. It also contains metrics on concurrency and throughput. These data can be used to create response-time service level agreements with customers and can be used to proactively perform capacity planning analysis.

DEPLOYMENT ISSUES

The operational architecture also encompasses deployment choices. Many infrastructure services, such as servlet engines and database engines have the capability to be in a clustered environment. Clustering allows a single logical resource to be deployed across multiple instances, likely on multiple machines. This has the potential to increase the availability of an application, because a single instance can become unavailable through a failure in the service or a network connection to the service. The resource, such as a database, can still be available to the application through a secondary instance in the cluster. Determining how to best place these resources in the infrastructure can be a complex analytical process.

Grids are starting to emerge in the industry. A grid takes the clustering concept further to support the concept of “on-demand” availability of computing resources. It is likely that in the near future, enterprises will have the option of obtaining external resources from a grid vendor, much the same way that electricity is available on demand now. This holds opportunities for lowering the cost-of-computing and handling transient, seasonal workload peaks. It also complicates deployment issues for operational staff. An architectural approach to deployment of resources can position an enterprise to take advantage of this emerging capability.

Deployment will involve the distribution of resources even without the use of clustering or grids. It may make sense to segregate the deployment of applications to different systems based on their resource utilization characteristics or their availability requirements. There is a significant difference in deploying a resource over sixteen low-power machines versus two higher-powered machines. While the increased number of machines

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

can improve availability, it also increases the support burden for applying software patches or diagnosing the scope of an outage to a resource.

Deployment issues aren't purely technical. Some vendors license the use of their software based on the number of operating system images. Some license based on the number of CPUs on which it is deployed, regardless of how many operating system images are used to manage those CPUs. The operations architect needs to work with the contract negotiators and the financial organizations within the enterprise to find the appropriate balance of technical requirements and fiscal prudence.

EVOLUTION OF DIAGNOSTICS

In simpler times gone by, the bulk of large enterprise applications were run on IBM mainframes using the application and operational architecture provided by system services such as CICS, IMS and TSO. IBM traditionally was strong in enterprise operational issues, so performance and diagnostic information was often available with little or no effort on the part of the applications developer. Systems programmers could obtain these metrics from the underlying system services.

As applications moved to midrange and desktop environments, those operational facilities were not available in those operating systems and systems services. Most applications developers didn't appreciate the need for that sort of information and the few who did had to focus resources on delivering application functionality, not operational functionality.

The most disturbing trend to me in the IT industry has been the significant decline in the operational infrastructure of software systems. The ability to diagnose problems is substantially *reduced* compared to when I started in the business as a mainframe systems programmer in 1982. There are a couple of primary reasons for this. The first is the emergence of small, desktop computers that were initially incapable of sophisticated diagnostics. While operating systems such as Linux, Unix and Microsoft Windows are certainly capable of many of the diagnostic capabilities found in large mainframe systems, the decline of the influence of IBM over the industry in favor of companies without a strong enterprise background prevented a focus on diagnostics. In addition, the competitive pricing pressures and a pressure to come to market as early as possible, ex-

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

acerbated by the dot-com boom and the concept of “internet time¹²”, eviscerated any remaining focus on these important operational issues.

While the focus on the operational aspects of information technology decreased, the complexity of information technology deployment became exponentially more complex. Nearly all applications have a networking element associated with them. Even the venerable batch job is likely to access relational database resources that reside on a networked database server. Interactive applications rely on many networked resources, such as LDAP servers, Kerberos servers, Active Directory servers, NFS or SMB-mounted logical disks, storage area networks (SANs), and the list goes on. A failure in any of these resources can result in a partial application failure. For the reasons mentioned above, applications are often not very explicit about the reason for their inability to accomplish a task when an infrastructure service is unavailable or dysfunctional. This makes problem determination by the operational staff very difficult.

If that weren't enough, enterprises are interacting with other enterprises now more than ever. The emergence of web services will likely accelerate this trend. Internal diagnostics will now have the additional complication of needing to interact with other systems for which there is little or no familiarity. Problem diagnostic and resolution times will be extended by the need for more communication between the staffs of your enterprise and the staff of your business partner.

OUTSOURCING

The management of the operational infrastructure of an enterprise is one of the most unrewarding jobs in the Information Technology industry. Metaphorically, it is the offensive front line of the football team. When it works well, no one notices. When it fails to work well, it is *very* apparent and *very* critical. This leads to an interesting side-effect in human psychology. I think a lot of executives view IT operations in a negative light because the only time they are aware of what they do is when there are problems. If there is a part of the business that has the perception of struggling to be effective and there is a vendor that purports to be able to do it better, there is a compelling case to be made for outsourcing.

The issue is that in most cases, proficiency by the operational staff isn't the problem. It is the lack of a architecture and a lack of funding for the exceptionally unglamorous in-

¹² “Internet time” was a widely believed, but essentially vaporous, notion that the pace of change for internet-based services had to occur faster than similar changes a few years earlier. Often, it was an excuse to do things haphazardly.

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

frastructure issues. When the root cause is misunderstood, it is easy to make the wrong decision regarding a resolution. Outsourcing an IT organization that doesn't have a strong architectural foundation will not resolve any of the current problems. It will, however add a few new problems to the situation. An outsourced IT organization often involves a substantial increase in the need for communication and management of resources. Contractual and political issues can also exacerbate the existing complexity for resolving problems.

Outsourcing isn't necessarily all bad. There is a potential for enterprises to lower their IT costs by moving some resources outside the enterprise. It is my opinion that a strong architectural foundation is a prerequisite to successful outsourcing, not a technique to avoid the complexity of operational support.

DECISION MAKERS AND OPERATIONAL ISSUES

Understandably, the executive decision makers are focused on funding information technology to support business functionality. There is little appreciation for the value of funding development and implementation of resources to support the operational architecture. Managing today's IT infrastructure is expensive. Managing an IT infrastructure in tomorrow's environment will be cost-prohibitive without a strong operational architecture.

AFFECT ON OTHER ARCHITECTURES

Most of the decisions made in support of the other architectural areas have an impact on the operational architecture. The need for clustering or grid support in certain infrastructure services affects the physical architecture since products and implementation technologies must be able to support that. The use of clustering and grid technologies have an effect on the application architecture since that imposes requirements on the application's ability to write to local resources, such as a filesystem. The application architecture needs to define the abstractions that allow developers to use resources that appear to be local, but function transparently in a clustered environment.

The ability to support the operational architecture requirements for effective diagnostics, performance service levels and capacity planning means that the application architecture must define the way to deliver the diagnostic and performance metrics. An architecture-centric organization recognizes the value in doing so and ensures that applications that

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

are deployed within the enterprise supply the requisite information. Since we want the applications developers to be focused on delivering application functionality, not operational functionality, the ideal implementation is to encapsulate the needs of the operational architecture in an application framework that also encapsulates the requirements of the application architecture. This frameworks-based approach to delivering application functionality can bring substantial operational richness to an organization without incurring much, if any, loss of productivity within the application development staff. This is one example of how an architecture-centric approach to information technology can create synergies that leverage skills and IT resources in ways that aren't possible when each area is viewed in isolation.

Certainly, there is some overlap in the operational architecture and the network architecture. The implementation of redundancy through clusters and grids often imposes additional load and redundancy requirements within the network architecture.

APPLICATION ARCHITECTURE

The application architecture defines the structure and shape of the applications that run within the enterprise. For applications that are developed internally, the application architecture enables the enterprise to deliver custom business functionality in a way that is compliant with the needs of the other architectures. Internally-developed software is the most malleable, so it should be subordinate to the needs of the other architectures.

If the application architecture has to accommodate the needs of all of the other architectural areas, how does the application functionality ever get built and deployed? First, the enterprise must realize that all application developers don't have equal skill. The most advanced topics in application design have to do with design patterns. Design patterns are used to decouple parts of an application from one another and increase the resilience to change. These design patterns can be used to create application and infrastructure frameworks that decouple the application functionality from the functionality that is required from the other architectures.

What if I told you that it was possible to change from Kerberos-based authentication to LDAP-based authentication within an application without changing a line of application code? It seems implausible without an understanding of advanced object-oriented programming and design pattern skills, but such capabilities already exist in royalty-free, open source code. It is also possible to obtain performance and basic application health information with zero additional code to be developed by applications programmers. Logging and metrics frameworks can be developed and implemented within an application framework in such a way that basic information is made available to the operational staff with no programming required of the application development staff. These kinds of capabilities aren't even envisioned without an architecture-centric approach to information technology. With an architecture-centric perspective and the right technical skills to implement the vision, it can be accomplished with relatively little effort¹³ when compared to the benefits it provides to the enterprise. It isn't rocket science, but it is definitely computer science.

Resilience to change is easy to understand, but it takes a strong architectural foundation to get it implemented correctly. Again, the architecture is embodied in a set of frameworks that implement the design patterns to decouple the parts of the application from one another. Decoupling is key to accommodating change. Here's why.

When there is a change or addition to the application that is required from our customers, we have to change existing code or create additional code that encapsulates new func-

¹³ Many of the fundamental concepts embodied in this document have been developed by the author in less than 100,000 lines of Java code.

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

tionality. As soon as we alter existing code that has been quality-tested and has reached a certain level of stability, we have the potential to introduce code defects that reduce stability and correctness. In order to ensure that existing functionality isn't corrupted, a significant amount of work is involved in trying to assess the side-effects of the desired changes. Decoupling allows the application to compartmentalize the functions within the application in such a way to reduce the interdependencies and limit the unknown effects of change.

The effort to decouple code is not free. It takes a little extra time and code to accomplish this. In addition, not all programmers have the skills to do this properly or an understanding of why this is important. Even for those who do understand, time-to-market pressures may limit their desire or ability to implement the appropriate level of decoupling. The enterprise needs to be aware that time-to-market and time-to-develop don't necessarily represent the bulk of the lifecycle costs of an application. Supportability and the ability to change according to future demands are substantial contributors to the cost-of-ownership for any given application.

Binding is another concept that is related to the level of coupling within the code. Binding refers to the time at which code dependencies become established. Back when I started programming, object-oriented programming hadn't taken off, so the best-of-breed programs were written using procedural programming techniques and languages such as PL/I, COBOL and C. In most of those applications, there were some key data structures, such as the "customer name and address record" or the "invoice" record. The programs that used these key data structures were bound when the program was compiled. Changes to these key data structures had a profound effect on the programs that used them. Those elements of the program had early (compile-time) binding and tight coupling. An improvement over that was to create reusable subroutine libraries that encapsulated behavior and exposed only the parameters and outcome of the call to the routine. This approach deferred binding to program linkage time (link-edit binding) and reduced the coupling to a moderate degree. Dynamically-loadable modules¹⁴ moved the binding level to early in the runtime of the program by loading shared code when the program was first loaded into memory. Slightly more sophisticated programming techniques could defer the loading of dynamically-loadable modules until the code was called.

Object-oriented techniques, especially those that leverage design patterns, have the potential to substantially delay binding and increase the level of granularity of objects that are bound well into runtime. In order to exploit this exceptionally-late binding, a framework must be developed that implements factory¹⁵ objects to construct objects that are bound at a relatively high level of abstraction. If the object model in the framework is

14 These are "DLLs" in Windows, shared modules in Unix and link-pack resident code in OS/390.

15 This is the factory design pattern and is perhaps the most useful in frameworks development

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

able to factory-construct and bind objects well into runtime, there is a high probability that there is a very low level of coupling within the application. This approach has two substantial effects on the applications. They are very receptive to change that adds function. Most often adding functionality requires no changes to existing code¹⁶. Requirements that change existing functionality have the benefit of structural elements described in the architecture and embodied in the framework that limit the ripple effect of change to a small part of the code.

The second effect of this architectural approach is that there is an increased opportunity for reuse. Object-oriented programming held the promise of reuse through inheritance¹⁷, but that promise never fully materialized. The use of frameworks, which leverage the more fundamental aspects of object-oriented programming such as inheritance but push the implementation to a higher level of abstraction, have fulfilled the promise of reuse. I should have said that the promise has been fulfilled from a *technology capability* point-of-view. It still takes processes - and a focus on architecture - to enable that promise to come to fruition.

PURCHASED CODE

Every organization needs to assess the build versus buy equation when trying to determine how to implement required business functionality. Typically, this is driven by the business functionality the software provides, the cost of acquisition and the time-to-market. Purchased software often has a shorter time-to-market and a smaller acquisition cost than internally developed software. It typically has a much higher integration cost and is essentially incapable of responding to changing requirements. Features will be added when the vendor adds them.

Depending on how the application plays in the existing infrastructure, the cost of support can be substantially higher. I have seen my customers make a commitment to a some technology, say Microsoft Active Directory for authentication. They will purchase a product that uses an internal database for security. That creates a process change for security administration, which increases the manpower to maintain the application. It also

16 That sounds like it would always be true, but in fact that is rarely the case. Without a strong application architecture, adding code can be as difficult and error-prone as changing existing functionality.

17 Inheritance is a feature of object languages that allow basic functionality to be placed in a parent class so that child classes only have to implement additional features or override behaviors in the parent class.

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

creates an additional opportunity for a security breach since two disparate security stores need to be maintained.

This isn't meant to imply that businesses should develop all software internally. It just means that an architectural analysis should be done prior to acquisition so that the total cost of ownership could be more accurately assessed. Until that step is done, decision makers can't make an informed decision regarding buy versus build.

AFFECT ON OTHER ARCHITECTURES

For internally-developed applications, this should have a minimal effect on the other architectures. The best application architecture accommodates the needs that are derived from the architects in the other areas.

Purchased code can have a substantial effect on internal architecture. The worst cases of the “tail wagging the dog” in my experience has been with purchased applications. I have seen purchased applications dictate the physical architecture (runs only on Windows and Microsoft SQL Server), the security architecture (uses an internal database), application architecture (requires the .NET framework) and operational architecture (it can't run in a clustered environment).

It was noted previously that the best choice for a single architecture may not be the best choice for the enterprise because of the effects on other architectures. For the same reasons, a purchased application with the best fit in business functionality may not be the best fit for the enterprise if it forces a substantive change in the way that the application is deployed or supported.

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

THE VISION

The vision of the architect-centric enterprise is based on an ideology that honors the state of the art in computer science and information technology and tries to implement those ideals in practice. Most enterprises are focused on their core business, not IT, and so those business considerations tend to have more weight than IT considerations. That's the way it should be. However, the business decision makers must appreciate that their business is becoming more reliant on effective management of IT resources. To say that another way, information technology has become an integral part of most medium to large businesses, even if their core business doesn't directly relate to IT.

The goal of the architecture-centric enterprise is to decouple the architectural disciplines from one another. How many times have we seen where the same disagreements between areas are hashed out every time there is a new development project or change to the infrastructure? How many times have we seen a room full of the top technical talent trying to reach consensus on issues that are only peripherally connected to their area of expertise?

I have seen a pattern of behavior in many very technical IT professionals. *They tend to not know what they don't know.* A highly-skilled application architect tends to believe that they know all there is to know about data and security architectures as well. I have seen highly skilled data architects who think that the issues raised by applications developers are moot because the developers are just too lazy to program. After all, the data is the center of the IT universe, right?

Once an enterprise architecture is established and articulated, those discussions have to happen only when the needs of the enterprise start to outgrow the architecture¹⁸. When that happens, the process for having those architectural discussions is well-established. This approach empowers each of the groups to have a structural basis that serves the needs of their discipline with as much autonomy as possible. It creates a structure for articulating and negotiating the areas of overlap with other architectural areas so that undue burden isn't placed on one discipline through the decisions of another discipline.

Although the vision has an ideological basis and a technical focus, the effective architecture-centric enterprise realizes that the goal is to serve the business. Those two statements are not in conflict at all. Creating an architectural structure for IT within the enterprise is *how* we in IT can best serve the business.

¹⁸ In my 25 years in the industry, the second most difficult and costly problems were caused when the architecture was outgrown. The most difficult and costly is when there is no architecture at all.

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

ROLE OF THE CIO

It is the role of the CIO to be an advocate for the needs of IT while ensuring that the IT organization fulfills the needs of the business community in a timely fashion. The CIO that has a long-term vision that encompasses an architecture-centric enterprise will ultimately be able to serve the business community. That is accomplished by increasing resilience to change, shorting the time-to-market for change, increasing code quality and reducing outages by increasing the capability of the operational staff to diagnose problems more quickly and accurately. These are long term goals that sometimes mean that the immediate desires from the peers in the business community must be delayed or that a compromise solution must be negotiated.

ROLES OF THE ARCHITECTS

The CIO has an Enterprise Architect as close technical consultant. While the CIO manages the business aspects of IT and negotiations with the peers, the Enterprise Architect deals with the more technical aspects of how the architectures interact with one another. In addition to being the key technical advocate and consultant for the Information Technology group to the CIO, the Enterprise Architect must be a skilled arbitrator in the issues that arise among the architects in the various architectural domains. The Enterprise Architect is responsible for maintaining the balance of power in the influence of each of the architectural areas.

The Enterprise Architect must also be able to respond the realities of implementing the architectural vision in each of the areas. The devil is in the details and Information Technology has a lot of details. In spite of the best planning and skills, there are going to be unforeseen events. If an architectural decision places an undue burden on the technicians within an area, the Enterprise Architect has to negotiate a means to mitigate the pain.

The architects assigned to the functional disciplines must be skilled in the state of the art in their areas. They are the key negotiator and advocate for creating positive change that increases the effectiveness of new technologies in their area. Information Technology is one of the most rapidly evolving occupations. It is difficult to keep up with the changes in a deeply detailed, technical area. Each architect should assess what new skills and technologies are appropriate to the organization and how those can be exploited. They work with the Enterprise Architect to assess what change, if any, that has on the other architectural areas. An architecture-centric approach makes it more possible for new

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

skills, techniques or technologies to be introduced in a way that is non-disruptive to the adjacent architectural areas.

Finally, each architect must articulate the value of architecture to the less experienced staff and ensure that each employee rallies their efforts in support of the architecture. They can ensure compliance with the structure established by the architecture by creating a culture that understands the value in doing so.

CRITICAL SUCCESS FACTORS

Creating an architecture-centric organization has the potential to increase the effectiveness of IT in areas of code-quality, availability, time-to-market and cost control. There is the potential to profoundly affect the business. As with anything of complexity, there isn't a simple solution to these issues, regardless of how much your vendors try to convince you otherwise. It takes a lot of work and there are lot more ways to fail than to succeed. Being aware of these success factors can help you assess if your enterprise is capable of implementing these kinds of changes and the things that can take this effort off track.

The first issue is that the Information Technology group has to establish credibility and trust with the business organizations. Quite often, IT is seen as the barrier to implementing the needs of the business instead of an enabler. If there is an antagonistic relationship between the business groups and the IT leadership, the implementation of an architecture-centric organization is considerably more complex. During the initial implementation of the architecture-centric approach, there may be increases in the time-to-market while those technical underpinnings are being established. There has to be a level of trust that there will be a payoff to the business community in having a strong architecture on which to build.

Every successful organization remembers who their constituents are. It really is about serving the needs of the customer. While an architecture-centric organization is based on an ideology, we IT professionals must remember that we are here to serve the needs of the business, not play with the coolest new toys or have experience with the latest buzzwords on our resumes. Since the short-term needs of the business may be at odds with the long term goals that are embodied in an architecture-centric enterprise, we have a dilemma.

When I was in my mid twenties, I asked my mentor, "How do you tell a customer 'No'?" He said, "You don't. You have to convince them that what they want to do is not in their best interests." That is a much greater challenge, but it is the right mindset for an organization that cares about its customer. The right Enterprise Architect has the skills to be able to articulate those issues in a way that makes sense to the business customers.

That leads us into the next critical success factor. Architecture is not about managing people or budgets. It is very much a technical endeavor. Declaring a manager to be an architect is a recipe for failure. Architects must possess the deep and current technical skills required to understand ramifications of their decisions. They must also possess very strong skills in language, diplomacy and negotiation. That is a rare combination. Deeply technical and intelligent people often have a "black or white" view of the world

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

and can be terse when dealing with others. When you find that special talent with technical and people skills, do what it takes to keep them. Empower them and reward them.

Architects must have the ability to envision not only what is currently needed by the enterprise, but how to accommodate unforeseen future needs. A good architect can do that, but they will often take an approach that is on too grand of a scale to be implemented in a cost-effective manner.

A decade or so ago, there was a trend in data architecture to create the “Enterprise Data Model.” This effort was of a grand scope that endeavored to define and document every field-level business datum. These efforts almost universally failed. I know of one business that was literally bankrupted by their pursuit of an enterprise data model.

While it is vital to be able to have an architecture that can scale to accommodate unforeseen future requirements, it is important to recognize that the implementation doesn't initially have to envelop all aspects of the architecture. This “architect big, implement small” philosophy can be used in all of the architectural areas. The Enterprise Data Model of a decade ago has evolved into the implementation of data marts that integrate information from different, largely independent data stores. There is still an underlying, grand data architecture, but the implementation with data marts is more manageable in scope and is able to come-to-market in a reasonable amount of time with a manageable cost.

CONCLUSION

The Information Technology industry is not short on “silver bullets.” The problem is that the gun to shoot the bullet typically doesn't work very well. The architecture-centric approach to IT is the way that an enterprise can construct the gun¹⁹. When there is a funding shortfall or a staffing shortage, these architectural issues are typically the first areas to be cut. While that may serve the short-term need, there is always a negative long-term effect.

In almost every enterprise I have encountered, architectural and infrastructure issues are given the short shrift until they fail. We have seen examples of this basic human nature outside of IT. No one cares about how the basement in their house was poured until it leaks. When it leaks, it suddenly becomes more important than curtains, cabinets and carpets. Few give credit to the offensive line of a football team until they fail to do their job. When that infrastructure of the football team doesn't get the job done, it gets a lot of intense focus.

When I talk with the technical IT professionals in the trenches about the needs to support infrastructure, I get the sense that I am preaching to the choir. What is missing in most enterprises is the recognition by executive-level management that infrastructure and architecture are key to controlling costs and managing complexity inherent in today's IT systems. I hope that this discussion has increased the recognition of the problem and presented some tangible ways in which an enterprise can create an architecture-centric enterprise.

¹⁹ ... if you will allow me to extend the metaphor a bit.

AUTHOR INFORMATION

For more information on the concepts discussed in this document, contact:

Gary Murphy

Hilbert Computing, Inc.

13632 S. Sycamore

Olathe, KS 66062

(913) 780-5051

glm@hilbertinc.com

Alphabetical Index

architect big, implement small 30
author 32
Binding 23
build versus buy 24
capacity planning 17
CMP 13
concurrency 17
data cubes 12
data marts 30
data theft 9
decouple 23, 26
deployment 17
diagnostics 17
DMZ 9
EJB 13
Enterprise Data Model 30
Grids 17
HIPPA 15
identity management 14
inheritance 24
Kerberos 22
LDAP 12, 22
OLAP 12
on-demand 17
PKI 16
pricing on software 6
Roles 27
service level agreements 17
silver bullets 31
SOX 15
SSL 16
tail wagging the dog 25

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY

throughput 17

TLS 16

vision 26

VPN 10

THE ARCHITECTURE-CENTRIC ENTERPRISE

IMPLEMENTING SUPPORTABLE INFORMATION TECHNOLOGY
