IT Architecture - BluePrint For Client-Server/Open Computing

by James A Hepler, Senior Technical Consultant
Hewlett-Packard
Novi, Michigan, 48375-8024 USA
810-380-2207 - Fax 810-380-2568
jim_hepler@hpatc2.desk.hp.com

The Chattanooga Choo-Choo

In the United States in the 1850s, the Southern states had realized that for economic success they had to be able to ship their textile goods, tobacco, and other products to the more heavily populated Northern States in order to maintain their needed growth in business. The technical infrastructure to support this shipping was very primitive with most shipping being done by horse drawn wagons or a limited volume of slow moving river barges or ocean-going freighters. A major breakthrough was the construction of a network of train lines throughout the South to move goods hundreds of times faster with increased volumes along established internal routes. This was accomplished by individual entrepreneurs connecting Southern cities with their own train lines. There was no overall plan of action by the Southern States. Problems began to emerge.

There was not a standard gauge or size of track that trains ran on in the South like there was in the North. As goods were moved from one city to another, they had to be unloaded from one train, placed on wagons, and then moved to another train yard to be reloaded onto another incompatible train. Eventually goods destined for the North would make their way to hub cities like Chattanooga, Tennessee where they would once again be unloaded and reloaded to trains bound for the North. These Northern trains did have standard gauge tracks, so that cars could be routed where desired without unloading goods. The North had a railroad architecture based on standards -- the South did not. The South could manually connect to the standard environment, but the cost and time to deploy made their cost higher and made it impossible to ship perishable goods. They were at a major competitive disadvantage.

When the War Between The States was fought, the North could use this standards based infrastructure to move supplies and personnel quickly -- the South could not. The North could cripple the South by disabling the few hubs like Chattanooga that the South had created to overcome their lack of architectural planning. The North however had a flexible environment where they could reroute through a different hub if necessary. The
Southern train lines seemed to be successful for a while, but ultimately, the lack of planning and increased maintenance needed to ship goods proved to be a disaster.

This is similar to the evolution of many IT departments. Is your department like the Southern or Northern train lines?

Introduction

Many Data Centers/Information Technology departments have implemented or are implementing client-server or open systems environments. This is commonly done with great success without a formal or complete architecture plan. As things progress in their application deployment or technology advances, it is increasingly complex to make decisions on how to best take advantage of the benefits available. As business needs change, it is not always clear how to adapt the current environment or integrate new modules or feature sets efficiently. Often departmental decisions are made which don't quite fit with the rest of the company and bridges are made to create linkages. As this process continues, a spider web of technology results with thin strands of connectivity between various components. Unlike a real spider web however, the connections are made of differing materials which do not bond well together and require much maintenance. Eventually the inefficiency of the operation leads to budget overruns, cumbersome environments to change, and an increasing backlog of maintenance problems -- the environment most operations are making major efforts to avoid.

The necessity of businesses to respond to change is the challenge of the Nineties! The successful IT implementation must be based on standards but with a flexibility to adapt. An enterprise with a strategy to move towards Open Systems and/or Client-Server Computing is positioning itself to be ready for emerging technologies and to enable utilization of these technologies when there is a business advantage to do so. Many Information Technology departments have gradually evolved, moving slowly into new applications - often driven by the users rather than sound business planning. This environment eventually creates maintenance and support problems resulting in increased IT costs and poor deployment of new application technologies rather than the positive impact the movement to Open Systems should provide. The creation of an Enterprise IT Architecture provides a Principles Based BluePrint or Guide for efficiently progressing through the partially charted waters of open standards.

The Architecture Plan allows more rapid, accurate decision making on building the Open and/or Client-Server Target environment your enterprise needs for agility to respond to change. When modifications are necessary, there is already a guide in place for acquisition and
components of requirements definition. IT Architectures can include many areas such as hardware, system and application software, data communications, adopted standards documentation, IT infrastructure and enabling technologies, deployment and transition strategies, skills requirements, operations and system management, and many others.

Architecture/Smarchitecture

The word architecture is used many ways in the computer industry today -- much like the term Open Systems has been used in the recent past. There is much debate and confusion about different aspects of computing architecture and the interrelationships between the types of architectures being described by consultants and industry pundits. A partial listing of types of architectures and architectural components would include:

Application Architecture
Architectural Instance
Architectural Framework
Centralized Architecture
Client-Server Architecture
Communications Architecture
Computing Architecture
Data Architecture
Development Architecture
Disaster Recovery Architecture
Distributed Architecture
Enterprise Information Architecture
Enterprise Information Technical Architecture (EITA)
Information Technology Architecture
Open Architecture
Overall Architecture
Productecture
Reference Architecture
Security Architecture
Solution Architecture
Sub-Architecture
System Architecture
Technology Architecture

Many of these terms are somewhat self-describing and are therefore often used in casual architectural conversation today. Encyclopedia Brittanica defines architecture as *the art and technique of designing and building, as distinguished from the skills associated with construction.* In information technology terms, architecture practitioners help businesses map their current business needs into a plan which will provide a framework for future IT decisions. An architect has a vision of what the IT environment will be used for, who will be using it, and what is needed to support current and future needs. In other terms, the IT Architect looks at the People, Processes, and Technology requirements and creates a vision that will support them.

The list of architecture terms above all share parts of this definition. A solution architect looks at a problem and creates a vision of a solution. An IT Architect might review one business entity and help create a vision of their IT needs. An Enterprise Architect would expand that to include the entire enterprise. Sub-Architectures such as Security, Communications or Development are components of IT Architecture.

A building architect analyzes client needs and creates a concept or vision without a great deal of detail, but with an understanding of component requirements. He also considers budgets, current standards of construction, and infrastructure as well as likely trends for the future. The client is involved with this process and when the architecture is approved, the detail design can begin. Construction crews can take the detailed design and layouts and actually build the building.

The architect knows that to support the number of people who will use the building, there will need to be a certain amount of floor space, environmental controls such as heat, cooling, and light, electricity and plumbing, etc. But he doesn't define the details. The architect doesn't worry about the gauge of the wire or the size of the pipe to support the amount of water needed for the lavatories in the building. He might consider growth however by envisioning land requirements for expansion or building placement so that wings can be added. He might also define a location near public transportation or parking. He creates the vision based on all of these pieces of information and known principles of what will provide a successful design and meet the occupants needs.

In a similar way, the IT architect helps the client formalize their policies on how they will use information to be successful in the mission of their business. The vision for IT created can then be used to make design decisions for implementation of IT systems in the most rapid and efficient way possible. Without the architectural vision, every decision in the future will require repeat of parts of the same process used to create the overall IT vision. With an architecture, the framework for decision making is in place, and reviews of designs can be done unemotionally, from predetermined principles and policies evolved from them.

Who Needs It?

For those IT shops who *"know what they need"*, or *"don't have time to do an architecture"*, or "
"we will complete the architecture later", you are normal. You are also like the Southern States in the earlier discussion -- things will go well for a while, but your long term chance of success is questionable. It is common for IT shops to be driven by application decisions which are likely to be successful on the whole, but do not leave time for creation of an Enterprise IT Architecture.

"Our CEO has decided we will be replacing our current systems with SAP!"

"We have already begun implementing PeopleSoft Human Resources and Oracle Financials!"

"We are designing new applications around Relational Data Base and PowerBuilder because we need client-server capabilities!"

These are all commonly heard statements and usually are from CIOs or IT Managers who do not have the benefit of a standards based, policy driven IT Architecture. They are handed many design decisions with the applications their company is purchasing or designing. To coin a phrase -- this is *de facto architecture*. It is easy for the CIO to then say, "we don't need an architecture because it is built in", or "SAP has designed everything we need". This is clear for some things, but what about network architecture, IT Management principles, Security Policy, DeskTop Strategy, etc., etc. What will tie together disparate application architectures if you purchase multiple applications? How will all of the systems deployed in your client-server architecture be managed in a consistent supportable way if each has a different application architecture and design?

An application like SAP, PeopleSoft HR, or Oracle Financials have an internal architecture. This might more properly be called a design, but is commonly referred to as application architecture. Components of those might be:

Relational Data Base
MS-Windows GUI
SNMP traps for management information
Encryption techniques for network security

These types of components are not generally part of the Enterprise IT Architecture, but could be part of the design layer detail or Application Architecture below the

principle definition. These layers of detail would typically follow the categorization of Architectural Principles.

By establishing Architectural Principles, guidance for future decisions can be made. Application vendors can be provided guidelines and asked to respond with how their products comply with your organization's IT policies and principles and where they don't. Some vendors can be eliminated much easier using this selection criteria as a filter. This is not to say that one hundred per cent compliance is necessary, but that an evaluation as to the ability of an application to fit into the enterprise IT Architecture can be easily made.

There is nothing wrong with *de facto* architecture. It can be used as input to the EITA. What is important is that the architecture plan is established and that principles are created for the future. An information architecture provides a consistent approach for IT Planning. It creates stability within an IT organization for integration of new requirements and emerging technologies. It is a fundamental part of the management philosophy for evolving and complex technologies.

Information Architecture Principles

Principles are the philosophical foundation of Information Architecture. With the changing business and technical environments all IT shops are forced to deal with today, it is necessary to develop principles for continuity and consistency. Principles provide the basis for analysis of all parts of IT decisions and evaluations. Statements of Principles define how an organization prefers to use information technology. Specific technologies or standards are generally not part of principle definitions. Many vendors discuss architecture and then explain that the best architecture contains their products. This is productecture and is not generally useful to today's open minded IT organization. Product information is useful when selection is being done, but not as part of principle definition or enterprise IT architecture.

Frameworks provided by principles are designed in such a way as to make choices easier. These constraints on decisions should not make costs prohibitive or restrict the capabilities needed to meet business goals. The intent is to provide a nurturing environment providing for leverage of changing technologies while controlling support costs. Looking up principle in Roget's Thesaurus gives words such as chastity, virginity, innocence, and goodness -- probably not nuances of the word that IT architecture should encompass. Preferable is the veracity, integrity, and equity part of the principle alternative.

Examples of actual client principles follow:

> The user computing environment must be accessible from any network connection.
> Security for data must be available and applied consistent with its value and confidentiality to the corporation and its need for business usage.
> Data owners are responsible for data integrity and distribution.
> Data is captured once and validated at the source.
> Management anticipates and plans for the replacement of obsolete application systems.
> Information Technology planning recognizes and supports the way people work.
> Successful information systems depend on well-trained staff.
> IT projects must have an executive sponsor and active business unit involvement.

Principles are usually organized in logical categories or sub-architectures chosen by the organization. An example of these groups is:

Infrastructure Principles
Application Principles
Data Principles
Organizational Principles

Another way of organizing principles is:

Technology Principles
Information Systems Principles
Data Principles
IT Management Principles

A more detailed approach might contain the following categories:

Application Principles
Data Principles
Technology Principles
Development Principles
Security Principles
IT Management Principles
Communications Principles

The categories chosen are a matter of style. The number of categories should be kept small however. As part of the principle statement, there is usually a simple rationale or intent with an implication description given as well. An example of a complete statement of one principle from an actual IT shop is:

Principle 6
Data is captured once and validated at the source.
Rationale:

Currently data may be captured and/or re-keyed multiple times.
To reduce redundancy, errors, and costs, data should be captured as close as possible to the originating source.
Interpretation questions are most effectively answered at the point of data capture.
Well-designed processes take into account source data collection.

Implications:

The organization needs to continue to explore and recommend alternative technologies for data capture at the source.
Data collectors/validators need to understand what the data means and why it is important.

Solutions must account for responsibilities for data validity even though the data may be collected elsewhere.

An improved work flow and simplified work process will reduce the resources needed for data capture and increase the availability of data.

IS developers need to develop work flow analysis skills and incorporate them into the systems development life cycle.

With a complete collection of principles covering all aspects of IT, it becomes much simpler to justify decisions based on the principles defined. The question might be then, "How does an organization create and formalize Statements of Principles?"

Principle Creation

There is a temptation for management to create IT Principles to save time or just hire an architecture consultant to create them. Another approach is, "Can't we just copy them from someone else?" These approaches have all been used, but a method which is often the most successful in the long term is to establish a committee or architecture task force. Empower them with the charter to interview people, do research, and give them appropriate time to do the job. They should be cross functional -- end users, business decision makers, IT management, developers, etc. It is also critical to have an executive available to be the sponsor of the effort and the tie breaker if that is needed.

It may be useful to hire an architecture consultant to facilitate a work shop to create a working statement of architectural principles. The benefits to this approach are many. With a cross functional team there is involvement from all parties and a built in commitment to adopt the principles created. In this way the principles become part of the IT and business culture. The team approach also allows for varying points of view that may provide insight beyond what either IT only or management only might envision. The workshop environment is also very conducive to rapid development of principles that can be of greater value than the alternative approaches. The bottom line is that Principles create an Architecture which is a shared vision of the future success of the organization.

Sub-Architectures

After the Statements of Principles are created for the Enterprise Information Technology Architecture (EITA), the underlying sub-architectures can then be completed. There probably are already components in place such as those discussed in the de facto architecture discussion earlier. Referring to the list of Architecture components earlier, some of those other topics can now be addressed. The EITA is the creation of the Vision of how information technology for the Enterprise will be used by the people, processes, and technology aspects. The layers below can now provide more detail for specific topic areas or sub-architectures, but the level of detail is still high.

For example, the Application sub-architecture could be used to group technologies to aid in platform selections. Included in this area should be guidelines for partitioning applications and integration decisions between various application groups. High-level logical models of applications would be described including descriptions of user interfaces, task logic, transaction control and logic, data interface, and extensions to other applications.

There may be information on approaches to integration with other applications. For example, the architecture might describe the use of Middleware to exchange data with legacy data bases. Different categories of applications may have different architecture components or requirements. For example, Decision Support types of applications have ad hoc inquiries, are read intensive, and generally are not mission critical. Scientific and Engineering applications are compute intensive and may require heavy use of graphics. Perhaps their GUI requirements are different. In effect, high level models of the applications are typically included with the application sub-architecture.

Detailed design and functional specifications are definitely not included in architecture, but will be needed for implementation of the construction level returning to our building a building model. Vendor selection and product choices are not part of Application architecture either. However, the application models and the principles they flow from will constrain the choices to be made, making the selection process faster.

There is clear overlap between the various sub-architectures no matter how the IT organization defines them. Security sub-architecture certainly has overlap with Data, Application, and Technology if those are other sub-architectures. Definition of these overlaps is critical to the success of the Enterprise Information Technology Architecture. Every organization will have different components in their architectures even if they are in the same industry, running the same application suite. Their

business strategies could be similar, but their policies and company cultures and experiences with IT will be different.

An Architectural Statement is the basic documentation in a sub-architecture. It is often accompanied by block diagrams, tables or other illustrations and cross-references to other sub-architectures and EITA principles. An example of a Data Architecture Statement follows:

XYZ Corporation Data Architecture
Introduction

The Data Architecture defines data location and how it is accessed. It is based on the Architectural Principles and Guidelines of the XYZ Corporation as defined in the Enterprise Information Technology Architecture.

Description

The overall Data Architecture contains several components which coincide with the Application Architecture components. There are shared data areas for use with many applications. These shared data areas contain basic information on products, customers, and company history to support strategic and tactical decision making. This shared access is managed by access modules invoked by transaction routing and managed by a transaction monitor. Business rules are implemented and lower level integrity is controlled.

Private data storage areas are also managed by access modules owned by the specific application that owns the data. This type of data does not require a transaction monitor, although one could be implemented in the future if the data becomes shareable.

Customer information is also contained in replication data stores at the regional data center. Updates are done periodically during the day on a master-slave relationship. Also distribution centers have subsets of replicated data loaded to them once a day for faster access of commonly used, but stable information. If a
customer fulfillment is required from a distribution center without that customer's data locally available, a transaction is created by the application which seeks that information from the central store.

The Data Warehouse is the ultimate source for all information for shared data.

Conclusion

The Encyclopedia Brittanica definition of architecture continues with, *"the practice of architecture embraces both aesthetic and utilitarian ends that may be distinguished but not separated, and the relative weight given to each can vary widely from work to work. Thus, at one end of the scale are purely functional structures (that nonetheless possess certain aesthetic qualities, intended or not), while at the other are purely decorative ones with no genuine practical function at all.*

In the Enterprise Information Technology Architecture, it is normal to concentrate on the utilitarian parts of this definition. Clearly the Enterprise IT Architecture must be functional and efficient to be of value. An organization without a functional enterprise IT architecture will eventually suffer from frustrated users, difficulties adapting to new technologies, increasing support costs, and loss of competitive advantage -- all the business drivers that are important to most IT departments for the 1990s. By formalizing the utilitarian EITA, the aesthetic values will become obvious to all involved with Information Systems. The true beauty will shine through!