A Disciplined Approach to Managing
Enterprise Information Systems Architectures

Abstract

Organizations, in both the public and private sectors, are undergoing relentless change. This dynamic environment has heightened the demand for the right information as a critical resource for organizational success. In addition to the information, the technology for accessing it, also continues to change. More than ever, this requires the information systems departments to be both reactive and proactive in their support of these changes. Many CASE tools are available for business analysis and systems design; however, given the rate of change, the artifacts must be integrated, accessible, analyzable, and reusable by developers and decision-makers. A promising strategy for satisfying this imperative is through a meta-information repository. Repositories are mechanisms for defining, storing, accessing, managing, and relating (via a meta-model) all the information about an enterprise, its data, and its software systems. There have been some noteworthy efforts to develop an effective repository. Unfortunately, the existing repository solutions are still "rudimentary" or "immature." One exception may be a repository, called DesignBank developed by Metadata Management Corporation. DesignBank is different than any other repository for a variety of reasons. Some of the most important include: (1) extremely user friendly, (2) web accessible, (3) captures all the metadata -- beginning with everything captured in requirement analysis, (4) artifacts are all integrated (e.g., the relationship between a business function and a candidate application), (5) artifacts remain intact -- you get out what you put in. The repository is a critical resource for anyone doing enterprise planning. The DesignBank repository not only protects an organization's investment, but also provides an integrated environment for developers and decision-makers. It is organized around the Zachman Framework. This paper will present arguments for a new systems development paradigm, which considers the enterprise as a whole and uses a meta-information repository as a mechanism for managing the information assets.

Introduction

Organizations face the inevitability of environmental change, politically, technologically, and economically. This dynamic environment has brought with it a demand for the right information as a critical resource for organizational success. It's not merely the availability of information, but more importantly the quality of information. In addition to the information, the technology for managing it also continues to change. The good news is the scientific and industrial communities have continued to develop state-of-the-art information technology. In a static environment, once a system is developed and implemented, very little change is required. However, in the dynamic environment, missions change, strategic plans follow, and new organizational goals and objectives are set. This changes the needs of the decision-makers and front-line users of information (Ridlon, 1993). As their information needs change, there must be a mechanism for effectively and efficiently identifying these needs.
and, once identified, translate them into a solution that is accepted by the information user. Spewak (1991) says that "In current dynamic business environment, users can no longer afford to wait six months, a year or longer for MIS to respond to requests" (p. 6). The response was business reengineering, streamlined information access through new technology, and software development methodologies that have put the concepts of reuse as requisite. Many CASE tools and analysis tools are available, as well as methodologies for using their artifacts (i.e., architectures and models) in the systems design process. However, given the rate of change -- the artifacts must be integrated, accessible, and reusable by developers and decision-makers.

These questions remain: (1) How to accurately keep pace with the change and capture the user's requirements; (2) How to transform these requirements into forms that can be used in the design of the system; (3) How to build applications and database systems that can be shared and reused; and (4) How to manage change by integrating the artifacts together from one perspective to another (e.g., a strategic plan to application). Today, most organizations are struggling with multiple systems, which do not share data very well, if at all. This is not only a serious limitation for the users of the information, but it's extremely expensive for the organization to maintain these systems and the interfaces between them that allow for information exchange. An organization can't properly manage the information resources without a repository -- especially a large organization. If the organization analyzes the cost of maintaining the code, interfaces, numerous systems, data which you can't access, etc. -- it becomes apparent very quickly that it's necessary to migrate off of the legacy systems, develop and enforce a standard data model, and build toward a corporate database.

What is needed is an enterprise strategy for managing information systems architectures which begins with understanding the enterprise and its information infrastructure and includes employing an automated mechanism for storing, retrieving, and integrating the artifacts that comprise and define that information infrastructure. Business process reengineering became very popular a few years ago -- but, how many times can an enterprise afford to reengineer? How does an organization protect its reengineering investment? Perhaps one of the reasons, the reengineering wave lost momentum in the last couple of years, is due in part to the idea that once all the work went into the reengineering process, the artifacts of that reengineering may not have been managed in a way that allowed an economical iteration of the total reengineered solution. The TO BE of the reengineering effort rapidly became the AS IS that needed to be changed again. An organization can't afford to be stuck on a perpetual reengineering cycle especially in light of the substantial investment required to do reengineering.

With a meta-information repository, the artifacts of the reengineering can be maintained, accessed, and reused. More importantly, these artifacts can be integrated with all other artifacts so the impact of change can be shown from one functional area to another, from data model to data schema, from one level of abstraction to another, from strategic plan to software code. Sowa and Zachman (1992) said "since the processes, entities, and locations are all abstractions of the same enterprise, each is related to all the others. Therefore,
Enterprise Information Systems Architecture Management

In 1977, John Rockart at MIT developed the concept of Critical Success Factors or CSFs (cited in McNurlin and Sprague, 1989). These CSFs, usually less than 10, are designed to capture the information needs of an organization. An IS organization is no different. Here are seven pretty good CSFs for Enterprise Information Systems Architecture Management along with their definitions.

Information, data, and Metadata which is:

<table>
<thead>
<tr>
<th>Easily Accessible</th>
<th>Anytime, anywhere, any type user</th>
</tr>
</thead>
<tbody>
<tr>
<td>Readily Usable</td>
<td>Compatible with commonly used tools and development methods</td>
</tr>
<tr>
<td>High Quality</td>
<td>Accurate; enforced and maintained</td>
</tr>
<tr>
<td>Standardized</td>
<td>Governed by policy</td>
</tr>
<tr>
<td>Cost Effective</td>
<td>Managed Efficiently</td>
</tr>
<tr>
<td>Reusable</td>
<td>Standard components for assembly of new artifacts</td>
</tr>
<tr>
<td>Integrated</td>
<td>Allow impact analysis of change. Relationships among artifacts</td>
</tr>
</tbody>
</table>

If we view Enterprise Information Systems Architecture management as a process, which it should be (Figure 1), these CSFs become attributes of the output products of the process. The presumption is that some change took place which added value to the information. The development artifacts and metadata are the inputs. Controls include the company policies or even the laws of the country such as the Privacy Act of 1974. The methodologies include the various systems development methods (e.g., spiral, waterfall, etc). Mechanisms, which enable the process, are of two types: One is Tools and Resources. This includes the tools used to develop systems such as, modeling tools, programming languages, other artifact generators. This brings us to the sine qua non (or essential ingredient) for the process -- the meta-information repository. However, before discussing the repository, there is one more argument to present: the need for using a framework for describing the enterprise.
A Framework for Organizing the Artifacts which Comprise and Describe an Enterprise

Economics and the need for a way to efficiently manage an enterprise's information drive a systematic approach to enterprise information architecture management. A strategic plan that articulates an organization's goals and objectives for the organization is the first step. From there, numerous methodologies are available for systems development. The basis of any systems development effort is a well-defined process by which an application is conceived, developed, and implemented. It gives structure to a creative process. Zachman (1987) proposed a framework that is capable of describing an enterprise using different architectural representations (Figure 2).
Each cell in the framework represents one of these architectural representations by combining or matrixing two concepts. First, the perspective of different participants in a system building process is represented (e.g., an owner's perspective, builder's perspective, etc.) which is defined by the rows of the framework. Second, a description of the products themselves are represented (e.g., data, process, technology, people, time, and motivation) which are defined in the columns of the framework by answering the interrogatives of what, how, where, who, when, and why respectively. Zachman's framework outlines what is needed to capture and define a complete set of architectures for an enterprise. Zachman suggests that the framework will integrate methodologies and tools which in turn establishes "credibility and confidence in the investment of system resources."

Strassmann (1985) looking ahead knew that in order for an organization to survive and operate, it needed to pursue technology that met its needs. He said "The technical design of information systems should be guided by the needs of the business strategy" (p. 244). Dr. Steven Spewak developed a process, outlined in his book *Enterprise Architecture Planning: Developing a Blueprint for Data, Applications, and Technology*, for developing the top row of the Zachman Framework, the business model. Spewak's Enterprise Architecture Planning (EAP)
methodology serves as a "how to" for determining the high-level architectures in the Zachman Framework and also provides a plan for implementation of those architectures. The EAP methodology is based on the functional business model and the organization's data requirements. Spewak suggests the answers to the questions of "What do you do?" and "What information is used to conduct the business?" define the enterprise business model. EAP defines the architectures from the business perspective and provides a plan to implement those architectures.

**Storing, Retrieving, and Integrating Artifacts**

It takes a dynamic management tool to facilitate the change management necessary to keep pace with the impact of changes in high level abstract strategic plans on process models or changes in data models on system implementations. Establishing these relationships is critical to enabling this impact analysis and change management. In an enterprise, the business entities, business functions, database schemas, and application are all related abstractions. Any changes in a business function will potentially impact all of the other architectures. Therefore, there needs to be a way to maintain these relationships. Adrienne Tannenbaum (1994), author of the book *Implementing a Corporate Repository*, explains this integration as creating relationships between the successive phases of the system development lifecycle. These vertical relationships can also be established between tool components (e.g., Planning tool B, Analysis Tool A, and implementation using Tool C). She argues that without a repository, it is questionable whether this can be done. In addition to the integration requirement, there needs to be a mechanism for making models [artifacts] accessible to the people who need to use or reuse them. Also, there must be a standardized set of common modeling constructs. This brings up a myriad of issues such as model [artifact] management (access control and configuration management (Tannenbaum, 1994). Zachman (1996) recently published some examples of these issues by posing the following questions: (1) What happens when the technologies change? or (2) the systems state-of-the-art changes? or (3) the business structure, processes, locations, cycles, people and rules change? or (4) the regulatory/competitive environment?. Zachman realized the necessity for integrating artifacts back in 1987. Unfortunately, the capability for doing it was limited until the meta-information repository.

**The Meta-Information Repository Concept**

Although, the concept has been around since the mid to late 1980s, successful implementation of operational repositories is still in its infancy. Numerous repository development efforts are being conducted and are now being documented in the literature. Unfortunately, these repository solutions are still "rudimentary" or "immature." Essentially, the repository functioned as a database for database schemas or other metadata artifacts such as data models or process models. Using a standard export format, these artifacts could be parsed, extracted, and imported into the repository. The "parts" were placed in as data. In order to get the model back for reuse, the reverse took place and the parts reassembled. This approach has some serious limitations including a limitation on the types of artifacts -- only a few of the modeling products are targeted for
extraction. In other words, extractors need to be built for any artifact and only a few were actually developed and made available. Taking a model apart piece by piece, storing the pieces, then trying to reassemble them back together again is another challenge. This proves to be a problem because what goes in isn’t always 100% reconstituted. The extraction process is also very time consuming, sometimes taking hours to do. It is not surprising that the popularity of repositories is not very high even though the need for one is there.

Today at least one repository, Metadata Management Corporation’s DesignBank has overcome these limitations and at the same time maintained the functionality of a true repository. First of all, the interface is designed specifically for the primary users. Many of the other repository projects are geared to providing a repository administrator interface or, at best, service to a very limited number of technical users. With DesignBank, the users range from strategic planners to the modelers. The plan for managing the metadata artifacts uses a "package" concept. Design artifacts such as a data model, don't usually stand-alone. There are often other documents, graphics, etc., that support decisions or enhance the primary design artifacts. These are lost with some other repository strategies. DesignBank captures the attachments as well. Moreover, the whole package is configuration managed (and versioned). Unlike any other repository, that the author is aware of, DesignBank keeps the artifacts intact instead of parsing for storage. This insures that what goes in is exactly what comes out. A subset of the metadata is extracted for a "card catalog" that can be used for browse and integration activities. This extraction is done from a copy of the artifact, so the artifact stays intact.

There is a client front end for the users with "write access." These are the primary developers of the artifacts. Others that will use the artifacts for reuse or as templates or as baselines for transformations, etc., can access the repository via worldwide web to browse, conduct impact analysis, and retrieve copies of the artifacts they need.

A Case Study for Enterprise Information Architecture Management

Like the private sector, the Department of Defense, the Air Force, and the Air Mobility Command (AMC) are undergoing change. More than ever, this required their IS department, the Command, Control, Communications, and Computer (C4) community, to be both reactive and proactive in its support of these changes. Many modeling tools were being employed for capturing the business requirements and transforming them into implemented systems. Given the rate of change (as discussed above) the high-level requirements were changing which impacted the architectures and subsequently the applications. Moreover, the drive for eliminating hundreds of systems (with many redundancies), stovepipe applications, and expensive interfaces had mandated some significant endeavors -- data standardization and software code reuse. To accomplish this, all the artifacts about the AMC enterprise had to be accurately captured and managed in a way that allows integration of artifacts and, just as important, these artifacts needed to be accessible to a variety of people in all phases of the System Development Life Cycle. As we have already indicated, a critical component for satisfying this imperative was a meta-information repository.
The initial stages of the project included development of the business layer of the Zachman Framework (using the Enterprise Architecture Planning methodology) and the installation of a prototype repository in 1994. These initial efforts demonstrated the feasibility and promise for using a repository as a central integration, storage, and management tool. At the same time, there were issues that related to performance, schema scale-up, integration of artifacts, and feasibility of managing entire designs completely within the repository.

As a second phase, Metadata Management Corporation (MMC) developed a new repository which welded together the various repository components, design tools, and artifacts into an integrated environment useful to developers, administrators, and other end users. Part of the focus was to provide user-friendly repository environment tools that implement the primary capabilities requested by end users. In the summer of 1996, the enhanced repository, called DesignBank, was delivered and put into initial operational use in February 1997. Development continued to provide additional capabilities specified by the customer. Today's capabilities include lifecycle partitioning of artifacts, version control, user/access management, audit trail, check-in/checkout, configuration management, and metadata browse of artifacts.

The computer platform for the repository is a high performance, expandable SUN SPARCcenter 2000 (Solaris Operating System), specifically designed for client-server computing. It supports the repository meta-model and artifacts using the Sybase Relational Database Management System (RDBMS). The client environment includes Pentium PCs with Windows NT. Supported client-based information engineering tools include MetaSoft Design/IDEF and Platinum Technology (LogicWorks) ERwin and BPwin. Other supported tools used for capturing information about the enterprise include MS Project, MS Word, and MS Access.

AMC is using the repository to store, retrieve, and maintain artifact metadata. Developers access the repository to identify, and possibly reuse, an existing model or entity from the same or similar function in which they are interested. Design artifacts and the associated attachments (e.g., presentation graphics slides, word-processed documents, project management files, spreadsheets, etc.) which comprise a modeling effort are configuration-managed and maintained as data in the repository. These artifacts and attachment files are stored together as packages.

**Conclusion**

Organizations must actively pursue an enterprise approach to managing their information system architectures, especially in the realm of the critical resource of the information. The imperatives are understanding the enterprise and its information infrastructure; employing a taxonomy or framework for organizing the products as architectures; and the *sine qua non* of an automated artifact management mechanism (i.e., a repository) for storing, retrieving, integrating, and corporately configuration managing these products.
Be Prepared with a Cost/Benefit Analysis. It's important to understand the benefits of managing enterprise architectures from many perspectives. If the organization analyzes the cost of maintaining the code, interfaces, numerous systems, data which you can't access, etc. -- it becomes apparent very quickly that it is necessary to migrate off of the legacy systems, develop and enforce a standard data model, and build toward a corporate database. How many times can an enterprise afford to reengineer? How does an organization protect its reengineering investment? Furthermore, these reengineering products need to be configuration managed. An organization can't afford to be stuck on a perpetual reengineering cycle. If an organization (especially a large organization) wants to manage all of their information resource architectures (i.e., technology, data, and application) from high-level abstractions through lower level transformations -- it can't be done without a repository. This is basic protection of a substantial investment. With the repository, the artifacts can be maintained, integrated, accessed, and reused.

BIBLIOGRAPHY


