

TiAC

# **A Framework Does Not An Architecture Make**

A White Paper of TiAC  
The Information Architects Cooperative

*By Jon Blunt*

## **A Framework Does Not An Architecture Make**

John Zachman, one of the leading evangelists of enterprise architecture, calls his approach “A Framework for Enterprise Architecture”. Unfortunately, it appears that any companies that adopt this and other enterprise architecture methodologies believe that the end deliverable of the process is an architecture.

This is unfortunate because frameworks are not architectures: there are significant conceptual differences. Also, the effort needed to collect the data to populate a Zachman Framework is not trivial. Indeed it is so significant that most companies working with it focus on specific areas, leaving large parts of the framework empty.

The tools of enterprise architecture are very useful and often under exploited. However, they do not deliver what most systems designers recognize as an architecture. In general if what is required is a systems architecture, a high level design of an application or service to meet defined requirements, completing an enterprise architecture framework will not get you there quickly or with any secure knowledge that the final system will satisfy those requirements.

### **Enterprise Architecture**

Enterprise architecture has been an IT discipline for over ten years now. Many companies have formal enterprise architecture programs, and every year more companies get started. The motivation is generally very simple. Companies spend a lot of money on IT every year and there are genuine concerns from executive management that this money is not being spent efficiently or effectively. Enterprise architecture promises to provide a more rational process for organizing IT investment and delivering business benefits. This premise is now accepted to the point that enterprise architecture is becoming an industry best practice.

At the same time many companies report difficulties in getting enterprise architecture started, disappointment at the results, and even outright failure. First, it needs to be accepted that the objective of enterprise architecture is hard to deliver on; if it was simple executives would not have the concerns that they do. Second, many IT organizations set unreasonable expectations for enterprise architecture.

One common misconception is that the purpose of enterprise architecture is to deliver a high-level design that can be implemented: the classic definition of an architecture. More seasoned practitioners talk about road maps, governance and frameworks. When the process works well enterprise architecture creates required inputs to the systems/software architecture phase for an application or service project. Those inputs are valuable, but by the strict definition they are not an architecture.

To understand the relationship of enterprise and systems architecture an analogy is perhaps useful. In traditional building we can talk about Roman architecture and this has

a clear meaning.<sup>1</sup> In contrast to Egyptian and Greek builders the Romans made extensive use of the arch in buildings, bridges and other large structures such as aqueducts. Also the Romans used brick and concrete extensively. To our eyes the buildings may seem massive, with thick walls to take the thrust of the round arches, but in the technology of the day they were a dramatic step forward.

We do not know what “blueprints” Roman engineers used, but the problems of assembling large structures that did not collapse under their own weight implies a degree of planning and organization that makes us believe there was an architect or chief engineer for these projects. In the analogy each building represents a project or program that delivers tangible results. In information systems the primary deliverable is an installed application.

However, to create that structure the engineers needed access to a set of resources including labor, materials, know-how, and infrastructure. The buildings from the Roman period therefore are representative of a set of technologies, just as 19<sup>th</sup> century railway stations with their iron spans and glass surfaces represent the technologies of a later, more evolved industrial civilization.

The purpose of enterprise architecture is to ensure that projects have access to the technology and resources they need to realize their systems, i.e., projects develop architectures for systems from the resources cataloged in the enterprise architecture. The architecture delivered by enterprise architecture therefore is the production technology and resources for systems

If we go back to our analogy, the Roman engineers depended upon infrastructure that allowed large quantities of cement and other materials to be moved to the work site. The Roman state created roads and ports to support trade of all types. Similarly the buildings of the 19<sup>th</sup> century depended upon a rail system, blast furnaces and coal to create and move the materials they used. 19<sup>th</sup> century building reflects a much higher energy input into the construction of the building than was possible 1,500 years earlier. The great railway stations with their large unsupported spans were simply not feasible with Roman technology.

Information technology evolves more rapidly than building construction. Organizations typically have systems of different ages, all in production at the same time. The work that goes into developing the enterprise architecture is largely directed at laying out the strategy for migrating between generations of technology to the maximum benefit of the organization. It answers questions such as “Do we have the technology and processes to implement and deliver 7x24 web services?” “When do we plan to deliver these systems at scale?”

---

<sup>1</sup> Information systems are not buildings. Software is more flexible than physical construction and we expect systems to evolve in ways buildings don't. However, for the purpose of this discussion this distinction is not important.

The passive role of enterprise architecture, first described by Zachman, is to document the resources available to projects. What platforms—operating systems, database, and integration technologies—does the enterprise support? What architecture patterns should you follow to build a secure internet application that meets corporate standards? What are the alternative data/object models for capturing information about customers? Having this information available reduces the research and decision making time for each project and promotes integration and consistency across projects. The Zachman framework is one of the most comprehensive attempts to catalog the meta information that would be useful to a project.

As it evolves enterprise architecture becomes more proactive in a number of ways. Enterprise architecture defines standards, i.e., moves beyond documenting a catalog to selecting from alternatives in the catalog. This can include managing the life cycle of third party products, ensuring that redundant and obsolete products are retired. In addition enterprise architecture is often a leading participant in developing the governance process to ensure that standards are not only defined but followed

A mature enterprise architecture program is typically active in a number of related areas.

- Research—What technologies does the enterprise want to adopt, which of the emerging standards does it want to align with?
- Prototyping—Getting some experience of new technologies into the IT organization as proof of concepts.
- Generalization for reuse—Ensuring a project delivers a shareable or extendable solution that goes beyond the direct requirement of a single client to support a larger community.
- Infrastructure—Creating common services that can be used by projects. The motivation for this can be improved integration, delivery of security and other services that cannot be delivered by individual projects, or reduction in lead time for projects.

Each of these infrastructure services, just like any other system, needs a systems architecture. Also, it is the responsibility of enterprise architecture to ensure there is a meta- architecture that shows how projects can build applications using integrated services. Indeed, probably the most important “architecture” deliverable of enterprise architecture is the identification and specification of reliable, scalable, common services. If this is done well then the leverage these services provide will add value to the projects and enable the organization to deliver larger scale projects than they would otherwise.

## **Systems Architecture**

That said, a framework is not an architecture and there are significant differences between enterprise and systems architecture. First, enterprise architecture is not time bounded. Projects and programs usually exist in well defined timeframes with clearly identified end points. In the course of the project the architecture represents an early stage deliverable.

Enterprise architecture is more fluid and deals with the evolution of technology, processes and services. Also there is always a gap between the enterprise architecture and the existing systems. This is often described as the gap between the As-Is and To-Be architectures as if at some point the existing systems will have morphed into the target model.

This is a polite fiction. There is always a gap between the architecture and the extant systems. The truth is that many factors determine what investment is made or not made. The relationship between enterprise architecture and the systems portfolio is similar to that between an election manifesto and the government budget.

### **Quality and validity**

This points to the second major difference between enterprise and systems architecture, how quality is measured. A systems architecture should be evaluated against a clear set of criteria. Among these are:

- How well are the function requirements satisfied?
- Can the system be delivered for the cost allowed?
- Does the proposed design/ implementation process control development risk?
- Will the system deliver acceptable levels of performance?
- Can the system scale with the business processes?

As an engineering principle, there needs to be a discipline or model that allows for a validation of the architecture against each of these criteria. Enterprise architecture does not have these disciplines to the same degree. Often feasibility is the most important criteria: what will we be able to be successful with? Practically, accountability for the delivery of the applications lies with the project. The effectiveness of enterprise architecture has to be judged against the alternatives that were not taken and that will always be subjective.

It is unfortunately true that traditional systems development methodologies have not been good at answering these questions either. Standard methodologies often leave performance testing and validation until after functional testing; i.e., some of the most important questions about acceptance of the system cannot be answered until most of the budget has been expended.

In these methodologies the architecture deliverables often consist of Powerpoint or Visio presentations or perhaps a UML model. The quality of these documents is difficult to prove. Most organizations do not have processes to formally evaluate these, and typically there is not enough information in the documents to support a fact based assessment.

### **Model based Architecture**

This gap is one of the motivations to invest in enterprise architecture. The standards delivered by enterprise architecture limit the design choices the project makes to proven

patterns and known vendors. This discipline is a substitute for a testable architecture at the system level.

However, we have seen the enterprise architecture is more a framework than an architecture and does not have the tools to validate its standards against specific requirements either.

The way forward is to adopt the engineering principles of Systems Engineering in IT, including a rigorous definition of a testable architecture. In practice this means moving to model based architecture where the model of the system is validated before the system itself is constructed. Progress in this direction is constrained by the inadequacy of our tools for describing information systems. UML 1.0 is not useful in this because it concentrates on the functional behavior, and many of the critical aspect of the system can only be determined when the functional components are mapped down to the physical domain. For example, a system architecture for a ERP system may be satisfactory if all users access the system over low latency connections but may fail miserably when implemented globally. The location of the servers, the size of the servers, the I/O capacity of the storage subsystem will all be critical in understanding how a system will scale.

Enterprise architecture became an IT best practice in the 1990's. It has tightened up enterprise management of IT and was highly effective in helping organizations prepare for Y2K and to migrate through each of the generations of technology.

In this decade we will see model based architecture reach the same level of acceptance. UML 2.0 has the semantics to support a description of the deployment of functionality to the physical infrastructure. Tools exist to make predictions of behavior from system models, and the understanding of these tools is growing.

Today, organizations can publish a framework and call it an architecture. In the era of model based architecture the distinction between a framework and an architecture will be clear and unambiguous.