

Architecture 101
An Introduction to Architectural Concepts in Systems

Architecture 101

What does Architecture have to do with systems?

Years ago, software development, and systems work in general, was considered to be a black art without a lot of structure. Programmers received their instructions verbally, perhaps augmented by scraps of paper, or a chalk board discussion, and off they went. The programmers were managed by more senior developers, called Analysts, that spoke to the end user and formulated the system, translating the requirements into the program module specifications. The degree to which a system was documented in this environment depended largely on how large or small the development group was, and to what extent there was a need to communicate on paper rather than verbally.

Those days are gone, however, and a more structured approach is being taken. In an attempt to communicate the serious nature of systems development today, the term *Systems Engineering* has been pressed into service, and this has introduced the uninitiated to the concept of a scientific approach to systems development.

In the last 20 years, much has been done in systems science that supports that view of systems development¹. Systems science has, in particular, contributed concepts such as methodology, conceptual prototyping, and methods of representing systems graphically.

In the late 70's, a fellow by the name of Zachman took the step of comparing the building of computer application systems to the building of sky scrapers, or other physical buildings. That is where the term, Architecture, first got used in the systems engineering context. As it turns out, Zachman had identified a very reasonable and useful parallel.

What does a building architect do?

¹ A word about systems science. Systems science is not about automated computer systems, but rather, deals with all aspects of systems design (where a system is any structured approach to any topic or endeavor). To that end, it address the problems of how to describe ANY system in an unambiguous manner.

A building architect is brought into a project as soon as the business planning is done and a decision to build a particular building at a particular time is decided². The architect is shown where the building will be located, what its purpose and function is, and what the objectives of the owners are with respect to look, feel, user friendliness, and image.

The architect takes this information and follows a structured process to create one or more conceptual designs making different use of the land, landscape, and variations on the owner's vision. These conceptual designs (sketches, models, etc.) are then put before the owner for validation and selection.

Once the owner has selected the conceptual design that they like, the architect then begins an even more structured process with respect to project requirements. For a significant period of time the architect goes away and specifies the details of the building from various points of view (electrical systems, mechanical systems, ventilation, security, structural requirements, etc.).

Once the technical details are specified, they are issued to selected contractors for development and implementation. During the implementation, the architect is in place to ensure that the design is being faithfully followed and that the materials meet specification.

When the architect is involved in a larger project, say Canary Wharf, in London, England, then the process described above is multi-layered. The entire project area would be laid out at first, and then, on overall approval, the individual buildings are approached as described above. This multi-layered approach is, again, in agreement with the approach required to build enterprise wide systems for any organisation.

How does this relate to system architecture?

The system architect performs much the same role as the building architect. The degree of activity depends on the level of the project, whether building a single system, or participating in an larger suite of systems at the enterprise level. The discussion below outlines what is required in the larger context, and then develops into the role of the architect on the individual project basis.

² This implies that there is a strategic planning exercise and that key business direction and goals exist.

As with any project, the first steps come out of a planning exercise. In this case, the strategic business plan, which has as a subcomponent the strategic information systems plan.

Taken from the macro view, there are 4 architectures that fall out of a strategic information systems planning process. They are:

Data Architecture: The data architecture is a model of the information requirements of the business. The information requirements of the business express what the business must know in order to exist, and in order to meet its competition. This model is derived without regard to what will, or will not be, automated.

All of the modelling activities, at this level of analysis, are models of the business independent of what form the systems they represent will take. A system can be implemented on paper along with manual processes, or be fully implemented in an automated computer system; the mode of implementation is unimportant for this discussion. The essential element to capture is: what information must the organisation have. Once this is known, then decisions can be made to ensure that that information is in fact captured and is stored in such a manner as to make it useful to the organisation.

The data architecture is represented in the form of a conceptual data model³.

Application Architecture: An application architecture is a partitioning of the activities of a business into discreet, implementable chunks, and the assignment of priorities to those chunks (applications).

In order to know what the business activities are, a business activity model must be built. Just as the data model describes the information requirements of the business, the business activity model describes *what* the business must do, and what information is required to accomplish each activity. The activity model and the data model are built hand-in-hand, and only together do they describe the business.

The activities of the business are then partitioned into coherent groupings of functionality, and where possible, described as separate applications. This does not mean that the applications are not highly integrated, they should be, but it provides for a means

³ For more information on Data Modeling, see Data Modeling 101 by this author.

of breaking down the size of the overall requirements into implementable pieces. Priorities are assigned to each application, and based on this ranking, plans can be established for bridging, conversion, training, and infrastructure roll-out.

Communications Architecture: Depending on the size of the organisation, and how it is dispersed geographically, there may, or may not, be a requirement for a communication architecture. The communication architecture deals with the network infrastructure from the perspective of national service providers, switching systems, overall bandwidth, and the interconnections across heterogeneous suppliers. Sometimes seen as part of the technology architecture, it is separated in this discussion because, in large organisations, the negotiation and implementation of wide area network service involves a much different orientation than does the implementation and support of local area networks and client work stations.

Technology Architecture: A technology architecture deals with two aspects. The physical implementation of local area networks, and the implementation of all middle-ware products. Middle-ware products cover all of the software required to support development, maintenance, and in-service systems. (Examples of middle-ware: RDBMS products, development languages, configuration management tools, word processing software, imaging software, workflow management software, etc.)

These architectures all depend upon previously defined standards, or assist in defining the standards to be used across the organisation for all hardware, software, and middleware acquisition.

The application architecture has specified what systems will be built, and in what order. Using this information, overall project plans can be discussed and financing projections put forward. At some point, an application system is selected and funded to go ahead.

Application System Approval and Onward

First Steps

Once a given application system is identified for development, the more detailed work can begin. To start with, the definition of the application area has to be expanded, in that more detail has to be gathered. The parts of the business activity model that are addressed by the application, state the real business requirements, but these must be fleshed out in a more procedural specification. The shift in emphasis is from *WHAT* the business has to do, to *HOW* the business organises itself to carry out the required activities. The

corresponding information requirements also must be fleshed out to account for the functional requirements of the application.

These steps are supported by any number of methodologies. Whether traditional Gane and Sarson, or the more modern Object Oriented approach, the data and processes must be modelled sufficiently to identify exactly what the business requires with respect to information and process. Under the Gane and Sarson method, a logical data model would be developed in conjunction with a data flow diagram. Under the Object Oriented philosophy, the data and processes get modelled together in terms of classes and methods (perhaps in a UML format).

Regardless of the approach, the end result is a more detailed view of how the organisation is to go about its business given the opportunities of the new application⁴.

The discussion has to this point only dealt with the functional side of the application. One of the more significant aspects of application development within the client server paradigm has to do with Graphical User Interface (GUI) standards, and the development of base classes and/or frameworks⁵. In a client / server environment, the human engineering aspects of the design contribute significantly in the determination of the success of the project.

The GUI is the direct contact with the end user.

Standards

The role of standards cannot be over stated. Standards that are required at this stage are those that guide the content of the functional specifications, the look-and-feel of the application from the users perspective, and the nature of the underlying technologies and the network topology to be employed (distributed versus enterprise, client server, 1, 2, or

⁴ Typically, the value added at this stage is a definition of who does what, where, and when. The sequence of operation, the clustering of functions, and access requirements are all identified.

⁵ A base class is a set of objects that form the basis for all applications developed using those objects. Examples of these objects: windows, menus, drop down list boxes, data calculation functions, navigational aids, hot keys, etc. These are basic, unrelated tools that are accessible to all developers, as a class which they can inherit and build upon.

A framework is a larger concept than a base class. A framework attempts to encapsulate more functionality into its structures than a base class can. A framework element may not even use a class, or may represent a class or multiple classes. The framework approach comes a step closer to the concept of a business object library, where developers would ask for services from an object without regard for what goes on inside the black box of the framework element itself.

3 tiered, etc.). These are assisted in large measure by the architectures previously developed, but are fleshed out in more detail based on the tool sets and technologies available.

Standards can be implemented many different ways, and in fact, several methods may be used on a single project.

Written documents

Some standards are best conveyed in written form. These must be promulgated within the organisation, whether in paper form or via an Intranet, and should form the basis of an orientation package for new hires. This type of standard will deal with philosophical issues and methods of communication within the project.

For example: the fact that there are standards for a variety of topics has to be communicated. The project worker must understand that standards are important and that senior management stands behind them. There must be an intuitive explanation of what the development process is, what is expected of the developer, and how work status is communicated within the project.

Quality Assurance Reviews

Other standards are communicated simply through the process. If a developer cannot 'complete' a unit of work until it has been : peer reviewed, unit tested, signed off by his/her team leader, and submitted to configuration management, then there is a certain setting of standards enforced through those activities.

Tool Set

There is a certain nominal setting of standards by the corporate use of a standard tool set on the project. At the very least, there is confidence that project work done by one person will be accessible by someone else. The selection of tool set must be carefully considered so as to ensure that the tools fit the development environment and provide an adequate suite of functionality. Changing tools part way through a project can be very costly indeed.

Application of Object Oriented Principles

Without getting into the debate of what Object Oriented development really is, the important point is that one can take an object oriented view of a system, or application, regardless of whether the tools (compilers, CASE tools, etc) support object oriented development. The basic principle of value in terms of standards setting, is encapsulation.⁶

⁶ Encapsulation is the bringing together of data and process, and hiding the details behind a simple

This approach to large systems design has been around for a long time, only the names applied have changed over time. In the 60's and 70's, the industry called encapsulation a copylib or copy book. In the late 70's and early 80's, the name changed to system library with dynamic run time loading. In the late 80's and 90's, the name has changed again to include terms such as reuseability, distributed objects, widgets, and more recently, applets, and JAVA Beans.

The principle is that of identifying logical clusters of functionality and data, and building the components to deal with those clusters as stand-alone, callable 'things' that present to the developer a black box functionality. The benefits of this approach fall on the side of streamlined development, consistency of approach, ease of maintenance, etc.

interface. This 'object' then can be handled by the programmer from a purely functional use perspective without concern as to how the object works internally. An example of this might be the handling of addresses within a system, no matter what the function, an address should be dealt with in a consistent manner with respect to entry, validation and ultimate use. An address object would provide this functionality, providing the programmer with an interface that allowed him/her to insert, select, or update an address in a standard manner.

Separation of Church and State

Architecture is design, and architecture is not design. Architecture is development, and architecture is not development.

The statement above seems to be contradictory, but it is not. Architecture does indeed cross the boundaries between design and development, but it must never be considered to be one or the other. Just as a building architect is not a tradesman, nor an engineer, yet controls the activities of both, so too a systems architect. The system architect must be above the design and development, else project pressures will be applied to pervert the architecture inappropriately.

Is Architecture Impractical?

Often, architecture gets identified with an ivory tower or academic approach to systems, implying that the architecture is impractical and cannot, or even, should not be implemented. This is not the case however. Architecture, if properly positioned, represents the most pure form of the business requirements. Deviation from the architecture, is a deviation from what the business needs to do its job. This is not to say that deviations will not occur, but when they do, they will be done with full knowledge that they are a deviation, and justification for the deviations should be required on the basis of a business case.

Without the standard of the architecture, deviations occur without the sanity check of what the real business requirements are, and without the ability to back track should the business case for the deviation dissolve due to changed business environment, or the introduction of new technologies.

What is Architecture's Place

Architecture sits as a separate entity at a peer level to design and development. As such, architects should be involved in all design and development discussions where issues that affect architecture are being dealt with. What are these areas? Any decisions that affect: the meaning or intent of a data attribute, changes in the use of a data table in the database, changes in the functional approach to presenting information to the user, modifications to the basic look and feel of the application, changes in the scope of the application, or discussions regarding common code libraries, are all candidates for architecture involvement.

How is Architecture Staffed?

In a large project, the following architecture positions would be required. In smaller projects, a single person may adopt one or more of these areas.

- Data Architect
- Functional Architect
- Interface Architect
- Technical Architect

The Data Architect is concerned with the maintenance and integrity of the Conceptual Data Model, the development and maintenance of the Logical Data Model, and the generation and implementation of the Physical Data Model. They are responsible for the identification of data naming standards, data distribution requirements, backup and recovery mechanisms, disaster recovery processes, and of course, the proper interpretation of the data model to designers and developers to ensure proper use of the model.

The Functional Architect is responsible to ensure that the business requirements are accounted for in the system design documents and that development follows those designs. The functional architect provides the overall business insight to redirect discussions to what the real business is, and what the real problem to be solved is. Their role is very interactive with the business users, business analysts, systems analysts and the testing team.

The Interface Architect is the person that is most concerned with the user interface. This extends to the entire user environment including systems that may be outside the scope of the application being developed. Questions such as : how the end user will be able to do their work given multiple application systems; how close are the screens to the actual work requirements; what is the best paradigm to use in presenting the system screens to the end user (notebook approach, interactive expanding/collapsing structures, pre-formatted forms, dynamic forms, etc).

The Technical Architect is concerned with the ability of the application system to co-exist with other applications, and to work given the mix of technologies, middle-ware layers, and performance issues. The middle-ware layer provides the base class / framework services to the developers in addition to third party products.

