MASSACHUSETTS INSTITUTE OF TECHNOLOGY

SYSTEMS ARCHITECTURE – A KNOWLEDGE DOMAIN

October 17, 2000

Sebastian Fixson, fixson@mit.edu
Technology, Management and Policy Program, MIT

Course: ESD.83 Research Seminar in Engineering Systems
Prof. David A. Mindell

.. Listen to those voices: Architecture is a metaphor! says one. Architecture is a computer program! says another. It's the voice of the underclass! It's Imagineering! – or is it something on the event-space-movement axis? Can nuclear physics help? Can literature? What questions! What assertions! It's like listening to people fighting dragons in the dark.

(Paul Shepheard, 1994)

1. INTRODUCTION

Systems architecture is not a closely defined discipline – at least not yet. It is emerging from the understanding that, together with the increasing complexity of the systems we create and use, the importance grows to develop a general understanding of the underlying relations and mechanisms of systems behavior. Borrowed from the civil engineering world, the term *architecture* is used to describe these fundamental relations. *Architecting* is the approach to design this very early, but very fundamental part of system development.

Today, 'system architecture' is used in fields as diverse as defense, health care, financial engineering, biology, astro physics, or computer science. Common to most approaches is that they consider systems as consisting of several elements (subsystems, components, chunks, etc.) that are connected such that they perform a function that could not be performed by the elements alone. Three major schools of thought and their approaches towards system architecting (the process of creating the architecture) are explained below. The differences in their approaches can mainly be contributed to the origins of the fields and the mind sets they have developed over generations of researchers and practitioners.

Increasing use of software in almost all systems as well as legal developments like deregulations create new dynamics and needs to understand the effect of systems architectures on performance, acceptance and survival of various systems.

2. THE APPROACH: SYSTEMS AND ARCHITECTURES

Despite the large variety of systems for which the term 'architecture' is used, two common elements can be distilled from the approaches. First, 'architecture' is a metaphor that helps coping with complexity. Most of the times it does so by abstraction, by giving order to the chaos. It helps to make sense of otherwise difficult to understand complex phenomena or systems. Second, 'architecture' establishes rules. An architecture assigns roles to the elements involved, it declares number, location and function of connections between the elements.

For several thousand years humans have architected civil projects: houses, bridges, palaces. This seems to be the reason that 'architecture' can be used as metaphor in other disciplines allowing a quick understanding of the idea. At the same time, the word 'architecture' carries with it a

certain structure of thinking about objects. This has had various effects on how disciplines developed their approach to system architecture.

Most descriptions of 'architecture' include the assignment from form to function. Function is what the system eventually does, the behavior which it will have. In contrast, the form is what is created by the architect, it describes the elements and the way they are connected, i.e. their interfaces. The form is the 'thing' that eventually executes the function

Different types of systems place different requirements on the systems architecture. Therefore, system definition, and in particular boundary determination, has a strong effect on which characteristics of the systems architecture become important.

As systems vary in a number of dimensions, so does the focus when developing their architectures. Systems vary in size and complexity, and how often they are produced. The electrical system in a residential house requires a different focus than a national air defense system. They vary in number and type of human interactions. Architecting a petrochemical plant that human interactions are limited to its personnel and neighboring communities is different from architecting a public transportation system that is additionally used by a large number of customers every day. Systems vary in number and type of their stakeholders (client, builder, beneficiaries, adversely effected persons, etc.). They may vary in the authority structure (pre-planned, pre-determined, loosely-coupled, collaborative), and in planned life time. However, common for all architectural thoughts is the notion of synergy, the idea that the system is more than the sum of its elements and that creating and maintaining a sound architecture allows better system performance and survival. Cantoni and Ferretti characterize this fundamental feature of systems architecture as follows:

> The notion of a system's architecture is somewhat more than the elements which are used and the rules for their composition. An example is the design of arches: the set of bricks placed one next to another in a curve to form an arch gives rise to new capacity of supporting weights that is not given simply by addition to the elementary property of each component: a well-structured *system* comes forth, able to overstep the crude joining of the parts. Arch designing points out directly this astonishing capability to bring forth new qualities not directly derived from the scaling of the ingredients.[1]

[1] Cantoni and Ferretti 1994, p.1, italics theirs

3. HISTORY: SEVERAL SCHOOLS OF THOUGHT

As mentioned earlier, system architecture is not a concept invented by a single person or a small group of people. Rather, it is an idea that emerged over the last decades in several somewhat distinct areas. Each of them has its own world view, affecting the mental concept of what an architecture is, and what it does. Three major schools of thought are introduced below.

3.1 Large-Technical-Systems based School

As the term 'architecture' in everyday use implies, it carries with it the notion of 'architecting,' a synonym for planning, for creating a concept or a structure. The importance of a concept grows with the size of the system or the performance requirements put on to it. This is because the risk of malfunction usually carries high costs for large complex systems.

Historically, the technical fields that were characterized by both systems of significantly large complexity and the organizational structure that allows central planning and (to some extent) management of the entire system were the military and related operations. It is for this reason that the advancement of weapon systems, air defense systems or space exploration programs developed the need for 'system architecting.' During the period between 1945 and 1990 the political and social environment in most Western Countries was comparatively stable[2], so that the focus on system architecting was placed first to ensure system performance of increasingly complex technical systems.

The enormous difficulty to transport a human to the moon and back, for example, was managed to overcome by developing an organizational structure that could develop, produce, and maintain a system pursuing this goal. This included the development of very elaborate procedural policies for development, procurement, testing and certification. For example, to ensure system compatibility military acquisition guidelines fill books. That is architecting the system, i.e. the 'product,' was possible only by simultaneously architecting the process creating it. One result of this development is the so-called waterfall model of systems acquisition. Basic idea is to organize the entire process in a sequence of steps from requirements identification to concept development to design to operations. To achieve high system performance each step includes a

[2] The time period of the late 1960s and early 1970s began to change the public perception, but had only limited effects on large scale financing of military and space projects.

detailed guideline what needs to be done, when, by whom, and how. This approach follows the idea that thorough planning and execution in each step allows to keep feedback loops local and ensure the high level of quality needed to make such complex systems work. The demand for high quality is particularly important for systems that either affect public health and safety or that have very high cost in case of a system malfunction.

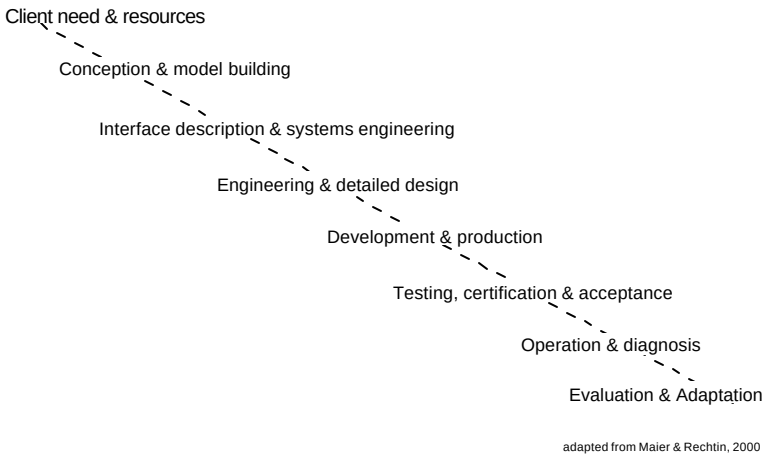

Figure 1: Waterfall model of acquisition

This type of system has been called builder architected system. Because of the technical complexity and the large number of unknowns, especially for unprecedented systems, past data is often of limited value. Thus, the scientific tools of classic engineering are often not sufficient. It is for this reason that systems architecting is often described more as an art than science and that good system architects often also have artistic qualities.[3]

In addition to builder architected systems a similar approach has found its way into another, historically hardware dominated, field: manufacturing. Since the system is now the production process, the waterfall model displays two intersecting waterfalls: one for product and one for process.

[3] Robert H. Liebeck, Lecture in System Architecture, MIT, October 13, 2000

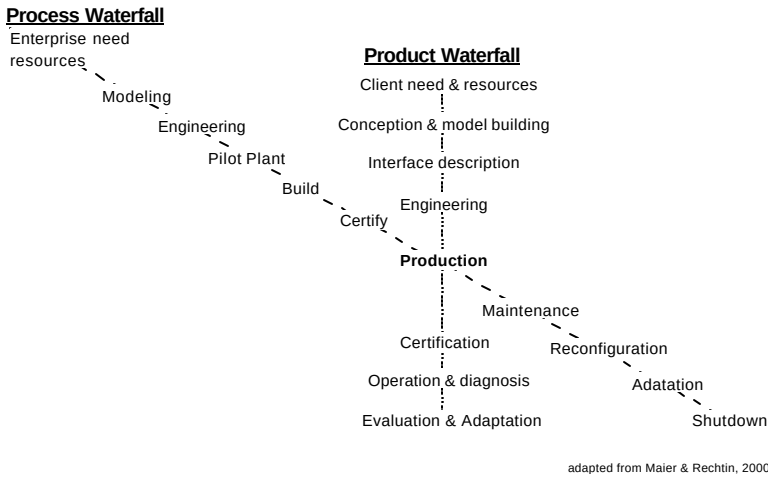

Figure 2: Intersecting Waterfall Models

Characteristic for the approach of system architecture that grew out of the large technical systems world is the mental picture of a hierarchy. In the hardware world hierarchies can be established unambiguously. A component either belongs to one subsystem or another – but not to both simultaneously. This fact finds it expression in sayings like "one person's system is another person's component" or "everything is part of a higher level super system and everything is composed of subsystems."

With the fall of the iron curtain in Europe in the early 1990s and the accompanying shift in the geopolitical situation, the financial and long-term planning condition for many military and space exploration projects has changed. This is one of the reasons that the architectural framework has been expanded beyond technical performance requirements to include the customer and his perspective on system performance.

Both the increasing importance of the customer's perception as well as a shift to systems with higher content in software are beginning to shift the classical system architecture paradigm away from one with procedures described in detail to one that is guided more by heuristics.[4] At MIT, Edward Crawley, dean of the Aero/Astro department, teaches a course titled System Architecture that, among others, includes many of the issues discussed above.

[4] This approach is especially promoted by Maier and Rechtin, 2000

3.2 Product Development based School

The architecture definitions that emerged from the product development world have – next to some similarities - a slightly different focus than those discussed in the previous chapter. This is not only directed to the difference in understanding of the terms *product* and *system*[5], but also to the purpose of architecting. While the focus of the large-technical system world has historically been on the problem to make the technical system work at all and under all conceivable conditions, the product development based school is more concerned with the question what does a certain product architecture do to the external market position of the product and to the internal process of product design and production. Both aspects affect directly the competitive position of the company making the product.

Similar to the large-technical systems world, the product development school has historically been hardware oriented. Thus, the definition of architecture has been described as "the scheme by which the functional elements of the product are arranged into physical chunks."[6] This assignment of the functional elements to the physical building blocks of the product includes the interface definition. Several authors have suggested that there are essentially two types of product architectures: integral and modular. While the former displays a complex function-form definition with often ill-defined interactions between subsystems and components, or chunks, the latter is characterized by a one-to-one function-form description and well defined interactions between the subsystems and components. The modular architecture is described as slot, bus, or sectional depending on the degree of interchangeability among the components.

The implications of architectural choices are far reaching. Product features like upgradeability, variety, serviceability, adaptability, use flexibility, or reuse options, are strongly affected by the product's architecture, and these features contribute to a product's success.

In a general sense, the underlying idea which is tied to modular product architectures is very similar to those promoted by the large-technical-systems based school, i.e. design robustness against changing environmental conditions, nature or enemy in the former, market preferences and competition in the latter.

---

[5] Although it is arguable that all products are systems, many certainly are.

[6] Ulrich, Karl T., Eppinger, Steven D., 2000, Product Design and Development, p.183

A tool developed for the architecting process is the design structure matrix, and its various derivatives. At MIT, Steven Eppinger has been developing design matrixes to map and optimize designs effort and relations between product architecture and organizational architecture. Others have developed procedures and methodologies to optimize module boundary definitions, i.e. architectures, along various dimensions as, for example, innovation dynamics (Balwin and Clark 2000), customer segmentation (Yu et al. 1999), product variety (Martin 1999), or product recyclability (Newcomb et al. 1998).

3.3 Computer-based School

The emergence of computers and software programs as means to control, calculate and operate system allowed the development of new ways to understand the systems' architectures. Despite its origin in software engineering, software development later began to make use of the specific advantages software offers. The fundamental difference that software embodies is that it is not bound to physical artifacts. This allowed to abandon the notion of hierarchy (or at least to give it a different meaning) and to create software with features as object-oriented and layered design.

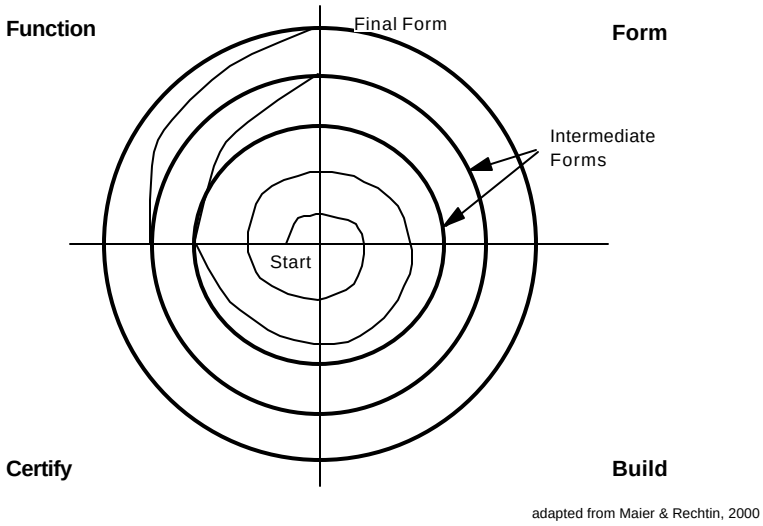

Figure 3: Spiral Model

These features together with totally different economics (almost all software cost occur during product development) changed the system creation and architecting process for software. The most common model used to describe software development is the spiral model. Here the

development is understood not as a well-planned long shot as in the hardware world, but rather as an evolutionary process from one generation to the next. The line between prototypes and 'real' products has been blurred. For these reasons software is far better suited to be adapted to fast changing environmental expectations than hardware.

Several, so-called architectural styles have been developed in software development. Pipe-and-filter architectures contain one type of component, the filter, and type of connector, the pipe. Object oriented architectures are built from components that encapsulate both data and function and exchange messages. Event-based architectures have as its fundamental structure a loop which receives events, interprets them and takes action based on a combination of both. Layered architectures emphasize horizontal partitioning of the system with explicit message passing and function calling. Blackboard architectures are built from a set of concurrent components which interact by reading and writing to a common area.

Although these descriptions show the implementer's point of view, what these styles demonstrate, is that software allows new ways of modeling problems. Historically, software has been moving successively higher in abstraction from computing hardware. As this seems to be the only economic way to build very large and complex systems it is to expect that the architectural software development will proceed on the path to languages and applications with higher levels of abstraction.

4. AN OUTLOOK

As the description of three different schools indicates, system architecture is not a well defined field, although it is moving towards this direction. Given its emerging character, it may be too early for a final evaluation. The wave of literature that has appeared during the 1990s in each of the three schools, however, seems to indicate that at least academia perceives systems architectures as a promising field. In addition, the increasing complexity of our world is equally likely to draw attention from the private sector to this domain. Recognizing the state of flux, I will discuss some anecdotal evidence where the development of systems architecture is headed.

Software is likely to take on the leading role in most products and systems, thereby increasing the weight of the 'software world view' in architecting. This may allow to develop new concepts to understand systems architectures in new ways. Some researchers predict that for fast pace

industries even the spiral model is too limiting. The insight that "we need to accommodate the fact that systems are integrated without prior design – out of components that were never intended to interact"[7] suggest the understanding the interfaces become the systems architecture.

Perhaps, it will offer a way to overcome the problem inherent to the classic hierarchical view of systems. In reality, many systems are not (only) hierarchically structured, e.g. the human body consists of several overlapping systems, i.e. it has a skeleton system, a nervous system, etc. Also, social systems often display structures other than hierarchical. "Often a complex system must be understood not as a simple hierarchy, but as a structure of overlapping and interlocking subsystems."[8]

In addition, a different type of system is gaining importance, which will require a different understanding of systems architectures. These are so-called collaborative systems, the most famous example being the Internet. "Long known as part of the civil infrastructure of industrial societies, they have come to greater prominence as high-technology communication systems have adopted similar models, centralized systems have been decentralized through deregulation or divestiture, and formerly independent systems have been loosely integrated into larger wholes."[9]

These latest developments introduce a dimension into system architecting which engineers are not necessarily familiar with. Ownership, control, but also purpose of the systems may not only be shifting over time but possibly be entirely distributed. Whether this process is equally changing the way engineers think as the move from feudal societies to democracies changed the thinking of political leaders remains to be explored in the future.

As systems architectures' creation and definition is moving continuously to higher and more abstract levels, systems architects may have to improve their interpretation skills. Higher levels of abstraction no longer determine the system itself (its elements and its connections) but only the rules with which it will evolve and operate.

---

[7] Perrochon, Louis; Mann, Walter, 1999, Inferred Design, p.48

[8] Mitchell, William J., 1998, The Logic of Architecture, p. 190

[9] Maier, Mark W., Rechtin, Eberhardt, 2000, The Art of Systems Architecting, p. 135

References and additional Literature:

Civil Architecture

Alexander, Christopher, 1964, Notes on the Synthesis of Form, Harvard University Press, Cambridge, Massachusetts

Mitchell, William J., 1998, The Logic of Architecture – Design, Computation, and Cognition, The MIT Press, Cambridge,Massachusetts

Shepheard, Paul, 1994, What is Architecture – An Essay on Landscapes, Buildings, and Machines, The MIT Press, Cambridge, Massachusetts

Large-Technical-Systems based school

Crawley, Edward, 2000, Lecture Notes to Course ESD.34: Systems Architecture, Massachusetts Institute of Technology, Cambridge, Massachusetts

DSMC, Systems Engineering Fundamentals, October 1999, Defense Systems Management College Press, Fort Belvoir, Virginia

Intelligent Transportation Systems, 2000, http://www.itsa.org/architecture.html, web page of the Intelligent Transportation Society

Maier, Mark W., Rechtin, Eberhardt, 2000, The Art of Systems Architecting, 2nd Edition, CRC Press, Boca Raton, Florida

Scott, Mark W. 1999, System Architecture Evaluation by Single Metric, unpublished MIT Master Thesis, supervised by Prof. Ed Crawley

Product development based school

Baldwin, Carliss Y., Clark, Kim B., 2000, Design Rules – Volume 1. The Power of Modularity, The MIT Press, Cambridge, Massachusetts

Eppinger, Steven D., Whitney, Daniel E., Smith, Robert P., Gebala, David A., 1994, A Model-Based Method for Organizing Tasks in Product Development, Research in Engineering Design, 1-13

Gulati, Rosaline K., Eppinger, Steven D., 1996, The Coupling of Product Architecture and Organizational Structure Decisions, Sloan Working Paper #3906-96

Martin, Mark V., 1999, Design for Variety: A Methodology for Developing Product Platform Architectures, Doctoral Dissertation, Mechanical Engineering, Stanford, CA, 172

Newcomb, P.J., Bras, Bert, Rosen, David W., 1998, Implications of Modularity on Product Design for the Life Cycle, Journal of Mechanical Design, Vol. 120, 3, 483-491

Steward, Donald V., 1981, Systems Analysis and Management: Structure, Strategy, and Design, New York/Princeton, Petrocelli Books

Ulrich, Karl T, The role of product architecture in the manufacturing firm, Research Policy, (24), 419-440

Ulrich, Karl T., Eppinger, Steven D., 2000, Product Design and Development, Second Edition, Mc Graw-Hill, Boston, Massachusetts

Yu, Janet S., Gonzalez-Zugasti, Javier P., Otto, Kevin N., 1999, Product Architecture Definition based upon Customer Demands, Journal of Mechanical Design, 329-335

Software based school

Barroca, Leonor, Hall, Jon, Hall, Patrick (eds.), 2000, Software Architectures – Advances and Applications, Springer, New York

Coplien, James O., 1999, Reevaluating the Architectural Metaphor: Toward Piecemeal Growth, IEEE Software, September/October 1999, 40-44

Cantoni, Virginio, Ferretti, Marco, 1994, Pyramidal Architectures for Computer Vision, Plenum Press, New York

Garlan, David, Allen, Robert, Ockerbloom, John, 1995, Architectural Mismatch: Why Reuse is so Hard, IEEE Software, November 1995, 17-26

Perrochon, Louis, Mann, Walter, 1999, Inferred Designs, IEEE Software, September/October 1999, 46-51

Shaw, Mary, 1995, Comparing Architectural Design Styles, IEEE Software, November 1995, 27-41

Journal of Systems Architecture, Else Vier Publishing