Chapter 23
Architecting Engineering Systems

Joel Moses

Abstract We discuss approaches to the overall design of large scale engineering systems, such as planes, cars and large software systems. Such approaches are usually called design methodologies. We discuss the top-down structured methodology, the layered or platform-based methodology, and the network-based methodology. Such design methodologies are associated with organizational structures or architectures, such as tree-structured hierarchies, layered hierarchies and generic networks. We discuss the relationship of these approaches to Aristotle's approach to the organization of Greek city-states and his logic-based problem solving, to Plato's organization of the Just Society, and to Darwin's use of evolution as an approach to design. We point out how these design methodologies relate to cultural attitudes toward engineering. We also point out that different engineering fields make different fundamental assumptions about properties of engineering systems, such as flexibility. We believe that undergraduate engineering education would be greatly improved if we taught design methodologies and their relation to philosophy. Similarly, engineering education would be improved if we taught foundational concepts regarding properties of engineering systems and the differing built-in assumptions regarding such properties found in various engineering disciplines.

23.1 Introduction

Design is the soul of engineering. Much of engineering analysis relies on mathematics and science, but design usually differentiates engineering from these fields. Our interest is in how one approaches the design of large scale and complex engineering systems, such as planes, cars and large software systems. It is difficult to teach the design of large-scale systems to undergraduates. Undergraduate design projects

J. Moses (✉)
Engineering Systems Division and Electrical Engineering and Computer Science Department, MIT, Cambridge, Massachusetts, USA

often simply emphasize getting a project done at all, given the limitations of time, staff and money. It is assumed that most issues related to large scale design will be learned by a practicing engineer on the job, often relying on approaches to design taught by his or her engineering firm.

General approaches to the design of large-scale systems are called design methodologies. Design methodologies, when they are taught at all to undergraduate engineering students, go by names such as systems engineering. We will discuss the advantages and disadvantages of several methodologies and how different methodologies relate to philosophical differences between Plato, Aristotle as well as others, such as Darwin. Of course, we make no claim that Plato and Aristotle were engineers. Our hope is that by analyzing different philosophical approaches and their relationships to design methodologies future engineers will be able to design better engineering systems.

We discuss the relationship of the design methodologies and organizational structures to different cultures. We indicate that there is a relationship between attitudes toward engineering as a field and the popularity of certain design methodologies.

Finally, we discuss the reasons for the development of the graduate program System Design and Management at MIT. One of the lessons learned in SDM is that individuals in different engineering fields make different assumptions about key system properties, such as flexibility and robustness. We believe that an analysis of such assumptions for each engineering discipline will greatly benefit all of our undergraduate engineering students.

23.2 Tree Structures

Design methodologies are not simply ways of architecting and designing technical systems, but are closely related to the structure of human organizations as well as general ways of attempting to solve problems. I believe that the first written description of a hierarchical structure for a human organization occurs in the Bible in *Exodus*. Jethro, Moses' father-in-law, visited him at the foot of Mount Sinai, just before the giving of the Ten Commandments. He watched Moses spending all his time judging. He told him that this is not good. Moses ought to appoint a judge for every thousand, every hundred, every fifty, and every ten. Then Moses would only need to judge the most difficult cases.

What Jethro is suggesting is a tree-structured organization of judges with Moses at the top, followed by a number of judges for every thousand, each followed by a number of judges for every hundred, etc. This is what I call a pure tree-structured hierarchy. Every judge has exactly one judge above him to whom he would deliver the most difficult cases he has. Such an organizational structure is very general and quite common, so much so that many people think all hierarchies are necessarily tree structured. Jethro's description is top-down, from Moses to the judges for every thousand, then to the level of judges for every hundred, etc. His description suggests that the cases to be judged are passed from the bottom up to the top. Aristotle used a

similar tree structured organization for administrating Greek city-states in *Politics*. He assumed that the tree-structured organization for a given city-state can have the number of its levels increased as the city-state increases in population.

Jethro's idea is in retrospect quite obvious. Why did Moses not think of it? My answer is that Moses had a different mind-set from Jethro, one that favored a very different organizational structure. In *Numbers* we find Moses creating the structure for the Israelite religious hierarchy. He places Aaron as the high priest at the top node of the hierarchy. He then places Aaron's sons (and eventually their male descendents) as priests at the top layer of the hierarchy. The rest of the tribe of Levi forms the middle layer, and the remaining tribes form the bottom layer. Layered hierarchies are different from tree structured ones is several respects. A judge in Jethro's structure has exactly one judge to whom he reports, and presumably he'll report to that judge for some time. The Levis, on the other hand, served whichever priests needed their support at any given time, and these priests could change from one day to the next. Priests worked in teams. So did Levis. In contrast, Jethro's judges likely were not intended to work in teams.

The structure of the Israelite religious hierarchy in *Numbers* is similar to Plato's Just Society in *The Republic* with a philosopher king instead of high priest, and a layer of guardians instead of the layer of priests. There are important differences. Moses has people born into their rank. Plato relies on testing to determine someone's final rank. I call such hierarchies layered hierarchies.

The two types of hierarchy can also be associated with approaches to problem solving. The tree-structured hierarchy is associated with the "divide and conquer" approach to problem solving. This approach works as follows: Given a problem, if you can solve it, do so and you are done. If not, break it up into parts and attempt to solve each part in turn as a subproblem. If you can solve a subproblem, do so and go to solve one of the remaining subproblems. If you cannot solve a subproblem break it also into parts and keep doing so until you either solve all the subproblems or you give up.

A treasure trove set of examples of top-down problem solving is in mathematical logic. Consider proving a theorem in Russell and Whitehead's *Principia Mathematica*. This process was actually automated in the 1950 s in an AI program by A. Newell, J. Shaw and H. Simon called the Logic Theorist (Newell et al. 1957). The approach taken by that program is a top-down one. I associate analytic philosophy, especially Bertrand Russell's version of it, with the top-down approach.

The design methodology associated with the top-down tree structured approach is the systems engineering approach, which became popular in the US in the 1950 s, largely in the aerospace field. Classic systems engineering is a reason for the success of the Apollo moon project in the 1960 s. A key reason for the success of systems engineering in the Apollo project was that the design was frozen early on. This avoided a weakness of the methodology, which shows up when changes are made to the design requirements in the midst of the design process or later on during the lifetime of the system. This weakness arises from the relative inflexibility of this approach.

23.3 Platform-Based Architectures

In a tree structure each node has exactly one parent node, and the parent node remains fixed for a long time. In a layered or platform-based system architecture, the parent node can change readily, and the members of a layer can be viewed as members of a team or set of teams. Current examples of human organizations that are structured this way include large partnerships, such as law firms or consulting firms. These are often structured in three layers – senior partners, junior partners and associates. Associates may be asked to work on a team with different partners in different engagements. Universities have vestiges of this structure – full professors, associate professors and assistant professors. Getting tenure is akin to making partner. In this sense, a university is a partnership of the faculty. A president who ignores this aspect of the organization of a university can get into serious difficulties with his faculty.

An example of a platform-based engineering system is an automobile platform. The bottom layer or the platform of the automobile is relatively fixed, and the top layer can vary to create different automobile models. The advantage of this approach to design is that it is usually cheaper to design a new car model based on an existing platform than it is to design the model from scratch. I claim that the relative ease of making such changes is a result of the flexibility of the platform-based design. Another advantage of the approach is that one can improve the platform over time due to its longer lifetime, which results in greater experience for the manufacturing firm, and thus the possibility that its processes can be improved over time. A disadvantage of the approach is that the initial design of the platform is more expensive than a specialized one for a new model. Another disadvantage is that the approach is somewhat restrictive. In general, all design methodologies have advantages and disadvantages. The choice of a methodology ought to depend on properties of the system being designed and its environment.

Automobile platforms are examples of power or energy-centered systems. It appears relatively difficult to create a hierarchy of platforms containing more than two layers in an energy-centered system. Hierarchies with three or more layers are easier in information- or communication-centered systems. We define the internal flexibility in a system to be related to the number of paths in it from a top node to the bottom nodes. Three layers are, in general, far more flexible than two using this metric. The architecture of the AT&T telephone system with area codes, regional numbers and local numbers indicates a three-layer architecture with a total of ten digits for each phone number. Each layer was originally implemented with a different architecture. The area code in the number indicates a node in a national telephone network, whereas the last four digits represent one of ten thousand possible phone lines in a local switch.

Software and mathematics provide many examples of layered or platform-based architectures. Consider a polynomial in x, y and z with integer coefficients. Any such polynomial can be written as a polynomial in x with coefficients that are polynomials in y, with the latter having coefficients polynomials in z with coefficients that are integers. Of course, the order of the variables can be changed. The order of

the layers will then change as well, although a polynomial expressed in one layered structure will be equivalent to one expressed in such a reordered structure. Plato was an idealist and close to mathematics, especially geometry. Although algebra was not well developed in his time, the algebraic examples would have been well appreciated by him.

Platform-based software systems abound. Consider a high level programming language, such as Fortran, the first popular programming language. Consider a system written entirely in Fortran, but one that runs on a particular microprocessor. The microprocessor does not actually run Fortran code. Fortran is usually translated to the machine language of the microprocessor through a series of steps. Thus we can say that Fortran provides a platform that separates the Fortran program from the actual bits of the machine language used by the microprocessor. Such a high-level-language-based platform provides expressiveness and flexibility to the system. An argument leveled against high-level languages is that their use loses some space and increases run time relative to a hand optimized machine language code. Such losses have been greatly reduced over the years as compilers have become better than most human programmers.

In fact, many software systems are currently based on a number of platforms, some of which are on top of each other creating a hierarchy of platforms that correspond to levels of abstraction. For example, data base systems, operating systems and special user interfaces can all be considered to be platforms. Given the prevalence of this approach to the architecture of large scale software systems, it is surprising that the methodology associated with platform-based design is not normally taught to computer science undergraduates. Such a methodology would have the architect of a new system spend significant time determining a new platform that would make it easier to design a new large-scale system. Most such platforms would be placed on top of existing platforms. Sometimes, especially when efficiency is a major consideration, new platforms could be interspersed between or below existing platforms.

In recent centuries the multi-layer architecture in the social and behavioral sciences was most popular in German speaking countries. In particular Marx and Freud can be said to use multi-layer architectures in their thinking. Freud emphasized the conscious and subconscious layers, but as a medical doctor he certainly could not ignore the physical layer of the brain underlying them both.

German speaking countries are not the only ones where can see a layered or platform-based approach to organization, problem solving or design. Japanese firms also use such an approach, and it is a reason why they were so successful in manufacturing relative to the US for decades. The Japanese used platforms in automobile design well before it became the norm in the US. Large Japanese firms tended to rely on an organizational hierarchy, which is partly based on age and provides little salary differentials for many years. They also rely on teamwork at each level. An advantage of this approach is that, within limits, it is very flexible. A disadvantage is that it does not work well outside of certain limits. Hence the appropriateness of the book on Japan entitled *Flexible Rigidities* (Dore 1986).

Management thinkers often emphasize the top level of a firm (especially the CEO). Japanese firms recognize the importance of middle management levels. Middle managers can play key roles that Americans often do not fully appreciate. One of their roles is to promulgate the firm's culture to their staff. Another key role is to get their staff to work well in teams. A related role is to build trust in the staff in employees in other parts of the organization. Then when new teams need to be formed, the team members can begin to work with each other relatively quickly and effectively. For an analysis see (Ouchi 1981).

Let us summarize some of the strengths and weaknesses of design methodologies that are associated with tree-structured and layered hierarchies. Tree structures are quite general. Most problems can be attempted with a top-down design approach. The approach is very logical. It works best when problems are relatively small or do not change much. It does not work well when the rate of change is relatively high since tree structures are quite inflexible. Flexibility in an organization occurs when there are alternative paths to a solution. Tree structures, unfortunately, have exactly one path from the top node to any particular node at the bottom. We associate Aristotle with this approach. According to my late colleague, Thomas Kuhn, even Aristotle's physics relied on tree structures with the Unmoved Mover at the top of the hierarchy.[1]

Layered hierarchies can cope with changing environments better than tree-structured ones due to their increased flexibility. This flexibility arises partly through increased cooperation and interaction between members of the same layer. Components of a layer can also have more than one parent node at a higher layer. The approach, however, is not nearly as general as the one associated with tree structures. It is relatively easy to break most problems into parts, although the break-up may not be unique, and a given break-up may not be best in the long run. It is harder to design a new platform because many issues or trade-offs must be taken into account. Thus greater attention must be paid to each such new platform design, and system architects need to be both experienced and creative. On the other hand, a platform can provide greater flexibility and expressiveness, as we noted earlier. We associate the layered approach with Plato, although he clearly did not emphasize the need for a capability for change in complex systems in his writings.

23.4 Network-Based Architectures

In recent years we have lived with technologies, such as the Internet, that undergo many changes all the time. Although the architecture of the Internet is based, at least initially, on the International Standards Organization seven-layer model, the topology with which we as users are most familiar is that of a network. A network-based architecture with nodes and edges is very general. All tree structured hierarchies and layered hierarchies can be modeled as particular networks, yet most networks

---

[1] Kuhn T., personal communication.

are not hierarchies at all. The physical and biological worlds can be modeled using networks. Networks are extremely flexible, even exponentially so as a function of the number of nodes. As a result a number of physicists and biologists have gravitated to the modern network theory. Watts (2003) gives an overview of this approach.

Networks are in general highly decentralized. This can be an advantage and a disadvantage. Decentralization gives a system great freedom and flexibility. The economy of a market-oriented nation can be modeled as a decentralized network of interacting actors. Centralization, such as that of the military, can give a fair amount of control over the actions of the various actors, more control than would be the case in a general network.

Networks are associated with problem solving approaches, just as hierarchies are. The determination of the price of a commodity in a market economy is a result of a multiplicity of actions of actors in a network. The open software movement in recent decades relies, in part, on actions of many programmers over the Internet. Wikipedia relies on a similar approach in its articles.

A related change to problem solving and design is due to the great success of modern biology. Evolution can be viewed as an approach to design that is related to a network-based architecture. The structure of relations of species to each other can be viewed as a tree structure, as Darwin noted. Whether the tree grows bottom-up or top-down is not of great importance here. Paths within a cell or an organism violate tree-ness and hierarchy sufficiently that the overall structure is best considered a network.

23.5 Attitudes Toward Engineering in Various Cultures

One of my concerns over the years, especially when I was Dean of Engineering, was why engineering was viewed differently in various industrial countries. The typology presented above of design methodologies and their relation to organizational structures and philosophy helped me understand some of these differences. Japan did not become a modern nation state until the Meiji restoration in 1868. Germany was finally united in 1870. These two countries thus have a better memory of their medieval past than Britain, for example. Layered organizations were relatively common in the Middle Ages, and became less so during the Enlightenment. The US was founded by people who were strongly opposed to a layered class structure, and the Founding Fathers were strongly influenced by the Enlightenment.

Science relies on logical arguments and tends to emphasize competition as well as specialization. A tree structured approach to problem solving fits rather well in science. Britain holds science in very high esteem. Engineers in Britain are often thought to be people who operate engines. Engineering usually relies on science, but also relies on teams to implement designs. Combining layered structures in addition to the tree structures works well in engineering. Thus I am not surprised that attitudes toward engineering are better in Germany and Japan than in Britain and even the US.

Hofstede and Hofstede (2005) analyzed the attitudes of IBM employees in sixty different countries. Geert Hofstede notes differences in the respective roles of the individual and the collective in different cultures. Tree structured organizations can be said to emphasize the individual, whereas layered ones tend to emphasize the collective. While he does not discuss the typology of organizational structures we use, the differences we note in national cultures show up in his work.

23.6 The System Design and Management Program

In my career I spent many years in the 1960 s and 1970 s arguing against the strong attraction that top-down tree-structured approaches to problem solving and design had to fields, such as Artificial Intelligence, Computer Science, Systems Engineering, and Management Science. In the 1980 s I found the Japanese approach to such issues a welcome endorsement of an alternative I had postulated in the 1960 s, namely layered or platform-based design. It was then that I realized that these differences in approach go back thousands of years, and that they are now deeply embedded in national cultures. In the late 1980 s American industry began accepting elements of this alternative approach as a way of coping with the Japanese success in manufacturing. Womack et al. (1990) analyze this approach, usually associated with Toyota. Yet I found that this acceptance of the Toyota approach was limited in scope. Toyota remains the best automobile manufacturer in the world today.

Just as it seemed that US firms were "getting it," the success of the Internet and modern biology led to the emphasis in the US on network-based methodologies. In my view, there is no methodology that is ideal under all circumstances. Each of the major methodologies has advantages in some situations. Networks are to be preferred when the environment changes very quickly or when the number of actors is extremely large. Layered systems are to be preferred in situations which are in middle range in size and rate of change. How to account for the quick skipping of the layered approach in the US? It cannot be that everything suddenly was getting very large and undergoing great change. Is it a surprise that Adam Smith, Charles Darwin and Bertrand Russell were British and that Marx and Freud were German and Austrian? I think not. Would it not be useful if one could encapsulate the various approaches to design, problem solving and organization of such thinkers so that future generations can use the appropriate approach in differing circumstances?

At MIT we had successfully introduced the Leaders for Manufacturing program in 1988 as a response to the Japanese success in manufacturing. LFM grants students two masters degrees, one in management and one in engineering. When I became dean of engineering I felt that LFM did not sufficiently address the issue of design in large scale engineering systems. This eventually led to the creation of the System Design and Management program, which grants a single masters degree in engineering and management. SDM has a core that includes a graduate course on system architecture. I had hoped to co-teach it, and make some of the points in this chapter about the architecture of engineering systems. Then I became the provost of MIT, and the teaching plan fell through. What happened is that the aeronautical engineers,

great teachers all, taught the subject and began with a base of systems engineering. They broadened the classic approach to systems engineering to include discussions of systems properties, such as flexibility. Yet the issues related to alternative design methodologies, such as platform-based design, were not addressed very well.

What I realized is that different engineering fields, not just national cultures, had different biases about systems issues. For example, a civil engineer would be happy to design a structure that had a single alternative in its design (e.g., a parking garage with four floors with the possibility of adding two more floors at a later point if demand so warranted). Computer scientists might not be interested unless a system had a billion alternatives in its operational use. Hence flexibility means different things in different engineering disciplines. In my recent courses on engineering systems I start with defining my national background, and that of my parents. I also describe the biases that I believe were introduced in my education in the fields of mathematics, AI, computer science and engineering. I think it would be wonderful if we had a deep analysis of the foundational assumptions of every engineering discipline. Our students would then have a far better notion of what they were getting into in their choices of engineering discipline. Likely engineering faculty members are too close to these issues. They could use help in such discussions from faculty in the humanities and social sciences.

23.7 Summary

Three major approaches to the architecture and design of large scale engineering systems are discussed. These are the tree-structured hierarchy, the layered or platform-based hierarchy and the network architecture. These architectures are related to approaches to the structure of human organizations. We indicate the relationship of these organizational structures to ones used by Aristotle and Plato. Evolution is considered as a design methodology in networked systems.

We note that national cultures have a closer relationship to some of these approaches to design than to others, although no single approach is ideal under all circumstances. We also note that different engineering disciplines make different assumptions regarding fundamental properties of systems, such as flexibility and robustness.

Ideally an undergraduate engineering education ought to discuss all these issues. Engineering faculty members are likely too close to these issues, and could use the help of other faculty members in such discussions.

References

Newell A., J.C. Shaw, and H. Simon. 1957. Empirical Explorations with the Logic Theory Machine. In *Proceedings of the Western Joint Computer Conference*, 218–239.

Dore R. 1986. *Flexible Rigidities: Industrial Policy and Structural Adjustment in the Japanese Economy*. Stanford, California: Stanford University Press.

Ouchi, W.G. 1981. *Theory Z*. Reading, Mass: Addison-Wesley.
Watts D.J. 2003. *Six Degrees: The Science of the Connected Age*. New York: Norton.
Hofstede G., and G.J. Hofstede 2005. *Cultures and organizations: Software of the mind*, 2nd ed. New York: McGraw-Hill.
Womack J.P., D.T. Jones, and D. Roos. 1990. *The Machine that Changed the World.* Cambridge, Mass: MIT Press.