

Systems engineering

As an industry, we have been “architecting” aerospace systems ever since the first Wright brothers biplane was built. But only in recent years have aerospace systems engineers begun to recognize systems architecture as an important discipline within systems engineering, one with its own unique methods and practitioners. This year, systems architecting further evolved toward becoming a standalone profession in the aerospace industry. The importance of a formal systems architecture discipline and the role of the systems architect have become vital topics for both systems engineers and their government and industry organizations.

What is systems architecture, and why is it important for systems engineers to recognize it as an independent discipline? Simply speaking, it is the conceptual property of a system: its purpose, top-level design, and overall structure. Systems architects bypass traditional systems engineering methods and use inductive reasoning and heuristics to develop system concept.

Conversely, traditional systems engineers use deductive reasoning and analytical tools derived from hard science and mathematics to analyze and trade options to develop system concepts. Now, however, increasingly complex systems are presenting more options than can be realistically analyzed and traded.

So, as system complexity increases, the systems architect becomes more important. Systems engineers and others who develop today’s system architectures must study architecture processes and heuristics and develop their inductive reasoning ability in recognition of the character of this “front-end” systems engineering process for highly complex systems.

To further understand the unique role of the systems architect, consider the metaphor of the civil architect and civil engineer. In the classic civil case, the architect works for the client and with the builder. The architect does not ask the client for his requirements; rather, that is

the reason the client has hired the architect—to convert his needs and desires into requirements for the builder. The client also expects the architect to resolve tradeoffs that occur in the building process. And just as important, the client expects the architect to certify the completed system as suitable for the intended purpose.

Clearly, these roles and relationships suggest the potential for a conflict of interest if the architect works for the builder. However, in the aerospace industry, where system technology is highly complex, the systems architect necessarily works for the builder, to more readily understand and evaluate alternative technologies and feasibility. Thus it is important that builders of complex systems formally recognize systems architects and keep them widely independent of project management organizations throughout the product development cycle.

In the future, aerospace systems will become more complex. Innovative architectures will be required for successful systems. A large body of knowledge already exists to train the systems architect. Processes have been developed and heuristics have been documented as tools for the architect. Formal architecture frameworks have been developed to document the system architectures. Systems engineers and others who develop and evaluate system architectures must study and apply the methods of systems architecting. Good systems architecture means having no regrets at system certification or for the life of the particular system.

Programs must recognize the need for disciplined systems architecting and allow for it upfront in the development process. They must avoid the temptation to get the operations view from the operator and the systems view from the builder. These views must be provided by a single systems architecting organization if new systems are to be successful. In the classic sense, the distinct roles of the client, architect, and builder must all be recognized if we are to be successful in building more complex aerospace systems in the future. ▲

Systems architects turn function into form using heuristics and inductive reasoning.



by Jimmy L. Allison