

Making A Science Of Design Work For Engineering

Position Paper Submitted To The NSF Workshop On The Science Of Design

John C. Knight

Department of Computer Science
University of Virginia
151 Engineer's Way, PO Box 400740
Charlottesville, VA 22904-4740
434.982.2216 (Voice)
434.982.2214 (FAX)
knight@cs.virginia.edu

Science As A Basis Of Engineering

A science of design is an *applied* science because design is about the creation of artifacts. The role of applied science is to provide a sound basis for engineering. Two issues concern me when I consider the topic of this workshop from the perspective of support for engineering and they constitute my position:

- The impact of system design complexity on software *and vice versa* is extensive and we do not have the tools or techniques to develop designs with desired properties nor analyze the resulting designs.
- There is very poor understanding and very limited use of the *current* science of design in the construction of software-intensive systems with the result that what we know now is rarely applied.

I will discuss each of these in turn.

The Interplay Between System Design and Software Design

The use of computers in engineered systems has progressed beyond the era in which the digital component merely replaced an existing electro-mechanical or analog component. This progression has come about because we now focus much of our attention on using digital technology to enhance system functionality rather than merely to improve the implementation of existing functionality. This change has forced new issues upon designers because it has led to an extensive interaction between the system functionality and the software specification. In almost all modern systems, it is no longer possible to separate the complexity of the system from the complexity of the software. This notion was first noted by Lehman in the context of information systems, but it has become pervasive in all manner of safety-critical and other crucial engineered systems. A lack of appreciation for this point has led to many practical difficulties and to research being undertaken that is not focused on the right problem.

The impact of this interplay is to require that: (1) software be viewed as a system component subject to all the influences that affect the system design; and (2) that software itself must affect the system design. Thus, for example, if software in an engineered system has to provide a service upon which overall system safety depends, then software has to be shown to function correctly in any feasible system state and the overall safety goals of the system have to be shown to be met with all the practical limitations that are inherent in any software development. It is not realistic to assume perfect software specification or implementation nor that testing can provide a sound basis for analysis if the software is part of a system requiring high levels of dependability. Thus, the system design has to be developed and analyzed with the potential inadequacies of the software in mind. And vice versa.

Use and Utility of The Current Science of Design

What we know about design now is being applied only very sparingly with the result that many modern systems are not engineered as well as they might be. As an example, note that some of the most complex engineered systems arise in the aerospace industry. The constraints of weight, power, and so on dictate system designs for aerospace vehicles that are heavily computerized. The use of fly-by-wire and fly-by-light controls, for example, reduce mechanical power requirements, reduce weight, ease maintenance, and improve system flexibility. Inevitably, the result is systems that make extensive use of computing.

Despite the widespread and increasing use of computer-based solutions in aerospace vehicles, the currently available science of design is rarely applied carefully or thoroughly. Ad hoc design approaches abound and the result is disasters such as the Lockheed Martin F22 avionics system, a system which recently had an MTBF of 1.5 hours. Such systems can be designed with the current science yet the current science is not applied. The future looks even more bleak. The F35 (Joint Strike Fighter) is projected to need 5.6 million lines of software, a significant challenge. Yet the stated development approach is to construct the software in large “chunks” and integrate these chunks as they are produced by a variety of contractors. This “big-bang” approach to design has never worked and will not work in this case either.

A Way Forward

To deal with the impact of the interplay noted above, the first step has to be to understand the problem and make researchers and practitioners aware of its the breadth and depth. The best way to approach this is to present the community with instances of the problem in the form of real systems currently being built or planned. From these instances, general issues that need to be addressed can be distilled. In this regard, it must be kept in mind that there are many issues and they affect all design disciplines from real-time scheduling theory to requirements analysis to verification. It is the whole system that needs to be designed, and the spin-off of issues when one sees sample modern applications is likely to be extensive. Presently, practitioners fail to appreciate the technical depth of the problems they face and researchers fail to appreciate the scale and sophistication of the applications with which developers are involved.

To deal with the lack of use of the existing science of design, one has to ask why this situation has arisen. It has come about, in part, because industrial developers are unaware of the existing science of design. That the existing science of design is not applied by those who should apply it suggests that a program of technology transfer should be a component of any path forward.

But the lack of “take up” by developers is not entirely because of a lack of awareness. It is also the result of deficiencies in the present science. The deficiencies are not the type of thing that present long-term challenges to researchers but, taken as a whole, they suggest that care must be exercised by researchers in the future to ensure that the results they achieve come with sufficiently low risks of adoption that developers will be able to employ them relatively quickly. We still have a gap between the technical merit of research results in this field and the form that these results need to take to be applied successfully.

Author’s Background

The author is a professor of computer science at the University of Virginia. His research interests include ultra-dependable software for safety-critical applications. Prior to joining the University of Virginia, he spent seven years at NASA’s Langley Research Center engaged in a variety of research activities. One of his current research emphases has been on the use of formal techniques in the development of software for applications in the aerospace application domain. He has recently begun a research program on the analysis of major accidents and incidents that were the result of failures of digital systems. He was a member of the FAA/NASA committee on Streamlining the Software Aspects of Certification (SSAC) which studied the entire development process for flight-crucial software looking for opportunities of reducing the cost of software development for commercial aircraft.