

# Position Paper for Science of Design Workshop

Larry Rudolph  
MIT CS/AI Laboratory  
Cambridge, MA 02139  
email: `rudolph@csail.mit.edu`

The evolution of heterogeneous and cognitive computing systems and our frustrations with getting them to work even somewhat error-free highlight the need for a science of design. Over the years, there has been steady progress in how to design systems that are written in a one programming language always running under the same operating system and on top of the same computer architecture with the same set of basic peripheral I/O components. But what is starting to work for these current systems is unlikely to work for tomorrows. In the future, systems are likely to be much more heterogeneous. For example, one component might be a high performance application written in C and executing on a grid. Another component might be a medium performance application written in Java. Additional components might be some python code to "glue" the components together along with some use of web services. In addition, the high performance code might dynamically migrate some critical procedures to the hardware or firmware level. Reconfigurable hardware is starting to become critical to achieve high performance.

Designing systems that integrate such components is a challenge for current design schemes. For example, strong abstraction barriers are considered a good thing, but it makes it very difficult to migrate functionality across these boundaries. When all the components of an applicaiton are strewn across multiple platforms, runtime systems, and adstraction layers, current design practices make it difficult to identify all the pieces. As usual, a failure in one place may have unforeseen consequences elsewhere, but when the pieces are everywhere, it is difficult to track such interactions.

What is needed is to present the programmer with a unified virtual view of the world and as parts get distributed, to maintain this fiction. This is similar to time sharing: For the most part, there is a fiction of a single (virtual) dedicated machine. We need a similar fiction for pervasive, ubiquitous computations. What makes this hard is that the design itself is often fragmented along the component lines. But a unified virtual view is not sufficient since it leaves out many issues, such as secure communication, authentication, and localized failures. The question becomes, how much of this fiction should be exposed to the design?

To make progress, it is important to be able to indentify, manipulate, and understand the principles of any design in order to apply them to they newly emerging heterogeneous world.