> conceived is too complex to be usable and should not be built. In other words, the mind of a single user must comprehend a computer architecture. If a planned architecture cannot be designed by a single mind, it cannot be comprehended by one. (1997)

Do you need to understand all aspects of an architecture in order to use it? An architecture separates concerns so, for the most part, the developer or tester using the architecture to build or maintain a system does not need to deal with the entire architecture at once, but can interact with only the necessary parts to perform a given function. This allows us to create systems larger than a single mind can comprehend. But, before we completely ignore the advice of the people who built the IBM System/360, one of the longest-lived computer architectures, let's look at what prompted them to make this statement.

Fred Brooks said that conceptual integrity is the most important attribute of an architecture: "It is better to have a system...reflect one set of design ideas, than to have one that contains many good but independent and uncoordinated ideas" (1995). It is this conceptual integrity that allows a developer who already knows about one part of a system to quickly understand another part. Conceptual integrity comes from consistency in things such as decomposition criteria, application of design patterns, and data formats. This allows a developer to apply experience gained working in one part of the system to developing and maintaining other parts of the system. The same rules apply throughout the system. As we move from system to "system-of-systems," the conceptual integrity must also be maintained in the architecture that integrates the systems, for example by selecting an architecture style such as *publish/subscribe message bus* and then applying this style uniformly to all system integrations in the system-of-systems.

The challenge for an architecture team is to maintain a single-mindedness and a single philosophy as they go about creating the architecture. Keep the team as small as possible, work in a highly collaborative environment with frequent communication, and have one or two "chiefs" act as benevolent dictators with the final say on all decisions. This organizational pattern is commonly seen in successful systems, whether corporate or open source, and results in the conceptual integrity that is one of the attributes of a beautiful architecture.

Good architects are often formed by having better architects mentor them (Waldo 2006). One reason may be that there are certain concerns that are common to nearly all projects. We have already alluded to some of them, but here is a more complete list, with each concern phrased as a question that the architect may need to consider during the course of a project. Of course, individual systems will have additional critical concerns.

Functionality
: What functionality does the product offer to its users?

Changeability
: What changes may be needed in the software in the future, and what changes are unlikely and need not be especially easy to make in the future?