them; rather, design architectures should be decided on how the architecture assists analytical thinking about evidence.” We render our abstraction in models, and we use those models to reason about our abstractions.

The best models have two important characteristics: they codify each design decision in a manner that is predictive, and they serve as the basis against which we can ask questions. For example, back to my services scenario, a good model would be predictive in that, if I invoke a service, I should be able to understand what other services are triggered and thus collaborate as that service plays out. Similarly, a good model should allow me to ask questions, such as how long might I expect a service to take to respond, or where might the boottlenecks arise were I to trigger a set of services at such and such a frequency.

The Threads of an Architecture
You’ve heard me say in this column that the code is the truth, but it’s not the whole truth. In the common vision that an architecture-as-an-artifact provides, there are five interwoven threads:

- architecture as a collection of significant design decisions,
- patterns as the themes,
- cross-cutting concerns as the traces,
- rationale as the back story, and
- tribal memory as the human story.

Thus, a description of a system’s living architecture is formed from technical as well as social elements.

The first three threads are primarily technical. The significant design decisions we assert give rise to our system’s essential shape, much like the load-bearing walls of a building or the essential mechanisms that operate within a cell. The patterns we find name the societies of collaborating abstractions; a given abstraction may participate in my many patterns, and the presence of those patterns gives rise to elegance and simplicity in the face of essential complexity. Cross cutting concerns—elements such as security or performance—represent things that are architecturally significant but so deeply embedded in the very fabric of a system that we can’t reason about them unless we pop up a level of abstraction. These are things that, more so than patterns, give rise to emergent behavior.

As Carl Sagan noted in Cosmos, “The brain does more than just recollect. It intercompares, it synthesizes, it analyzes, it generates abstractions.”
Architecture as a Shared Hallucination

Grady Booch

In Jane Wagner’s play The Search for Signs of Intelligent Life in the Universe, Lily Tomlin’s character Trudy remarks, “What is reality, anyway? A collective hunch.” She goes on to suggest, “Reality was once a primitive method of crowd control.” So it is with the architecture of a software-intensive system.

An architecture is just a collective hunch, a shared hallucination, an assertion by a set of stakeholders about the nature of their observable world, be it a world that is or a world as they wish it to be. An architecture therefore serves as a means of anchoring an extended set of stakeholders to a common vision of that world, a vision around which they may rally, to which they are led, and for which they work collectively to make manifest.

When I say that an architecture is a shared hallucination, I mean that an architecture-as-artifact is a naming of the mutually agreed-upon set of design decisions that shape a software-intensive system. While an architecture is just an abstraction of reality, an architecture-as-artifact is a declaration of that shared reality. In this way, that shared hallucination represents a common vision among a set of stakeholders as observed simultaneously through several different points of view and represented by a set of interlocking models.

Modeling

Now, to be clear, a model is just an abstraction of reality; a model is not reality. Although he was talking about statistical models, George Box quite properly observed that “all models are wrong, but some are useful.” In other words, as Scott McCloud wrote in Understanding Comics, we use models to achieve “amplification through simplification.” A useful model focuses on a particular concept to raise it above the cacophony of the complexity that surrounds it.

Building useful abstractions is hard: they must have degrees of freedom and at the same time be unambiguous. For example, if I’m trying to reason about the mix of services I want to expose in a given enterprise, in one view I might wish to consider the physical components that deliver up those services and where they’re deployed in a given network; in another view, I might wish to consider the granularity of those services from a functional perspective. In both cases, I’ll want to have a fairly unambiguous statement of the semantics of these services. In the former case, I may abstract away the implementation of those services (to focus on their deployment), whereas in the latter, I may abstract away the location (to focus on their functionality). This is why one view is never enough: if you try to pile all of a thing’s meaning into one view, you end up with the code itself, and that complexity will obscure the more delicate yet important threads that shape the system.

As Carl Sagan noted in Cosmos, “The brain does more than just recollect. It intercompares, it synthesizes, it analyzes, it generates abstractions.” He goes on to say that “the brain has its own language for testing the structure and consistency of the world.” This is why I’m a rabid fan of graphical models: they allow us to present an abstraction with some degrees of freedom yet without ambiguity, and then let the brain do the reasoning. As Edward Tufte noted in an interview (Technical Communication Quarterly, Autumn 2004), “The point is that analytical designs are not to be decided on their convenience to the user or necessarily their readability or what psychologists or decorators think about.