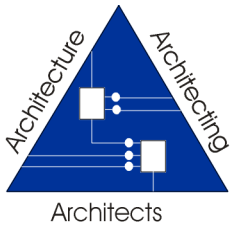
ARCHITECTURE RESOURCES
*For Enterprise Advantage*
http://www.bredemeyer.com




BREDEMEYER CONSULTING, Tel: (812) 335-1653

# Minimalist Architecture

As an architect, you have been picked from your organization's top technical talent. Your architecture is intended to guide and constrain, imposing your best ideas and lessons learned on designers and developers. But try to wield too much power and you will encounter resistance. Wield too little, and you make no contribution.

The solution is to take a minimalist approach to architecture—sort out what your highest priority architectural requirements are, and then do the least you possibly can to achieve them! That is, you should keep your architecture decision set as small as possible, while ensuring that your key system priorities are met. This column expands on this view.

## Architectural Decisions

Architectural decisions are those that need to be made from an overall system perspective. Essentially, these decisions identify the key structural elements of the system, their externally visible properties and relationships (Bass et al., 1997), and they define how the architecturally significant requirements will be achieved. If the requirement could be achieved by deferring the decision to a lower level, it is not architecturally significant and the decision is not architectural (at the given level of scope).

For example, if the system under consideration is an individual application, any decisions that could be made by component designers or implementers should be deferred to them and not appear as part of the architecture. If the scope of the architecture is a family of applications (or product line), then any decision that relates only to a single application (or product) should be deferred at least to the application architecture and not be part of the application family architecture.

**Minimalist Architecture Principle:** Sort out what your highest priority architectural requirements are, and then do the least you possibly can to achieve them.

This definition pushes us to identify the kinds of decisions that we would consider "architectural." Yes, there are those decisions that relate to identifying the structural elements of the system, and designing their interfaces (including specifying the externally visible behavior and properties) and relationships. Also, architectural decisions have to do with

- maintaining system integrity—having, or being conceived of as having, a single unified overall design, form, or structure, and
- cross-cutting concerns or system properties

Thus, some decisions may have nothing to do with the high-level structures of the system, but if they have to do with the integrity of the architecture, or if they could not be made from the isolated perspective of someone who has a narrow focus of responsibility, then they are architectural.

Remember, however, that the only justifiable reason for restricting the intellectual freedom of designers and implementers is demonstrable contribution to strategic and systemic properties that otherwise could not be achieved. Architects are highly valuable, essential technical assets of any company, and their attention should not be squandered on decisions that are not, truly, architectural. Similarly, designers and implementers are also part of the critical capacity to produce innovation and value, and their ability to do this should not be unnecessarily restricted but rather channeled appropriately to fulfill the architectural vision and the business strategy it implements.

**The RESOURCES FOR SOFTWARE ARCHITECTS web site (http://www.bredemeyer.com) contains a variety of resources to help individuals and organizations build their architectural competency.**

## Limiting Architectural Control

Architects have a unique vantage point. Having responsibility across the whole system, architects are able to solve problems in a way not available to component designers and implementers. Lack of visibility into other parts of the system, schedule crunch, communication overload—all manner of things cause developers to make decisions optimizing at local scope, which is often sub-optimal for the overall system. This situation is still worse when you are talking about components to be (re)used across multiple systems.

This is pretty heady stuff—your decisions, as an architect, impact the business's ability to execute its strategy in a way that those with local influence cannot. It is tempting, given how good architecture is for the business, to want to do more than the organization will absorb.

Yet each decision that you add to the architecture decision set has the potential to dilute the impact of all the others. That is, the bigger you make the architecture—the more all-encompassing, the more ambitious, no matter how well-intended—the harder it is for the organization to absorb, and the less likely it is to be embraced. The more you restrict the creative freedom that development teams have traditionally had, the more you will be resisted.

So, in addition to every other tough job you have to do as an architect, you have to figure out the right balance of architecture and anarchy for your organization! Starting out, it is better to err on the conservative side, yet take a few bold steps where they will make a clear difference.

Down the road, when architecture is an institutionalized practice, we may be able to dispense with this consideration. For now, however, architecture brings about changes in the way people need to work, and it is just as well to bear this in mind.

## Insisting on Architectural Control

Now let us go to the other extreme. The over-cautious architect who avoids "laying down the law" at all costs, misses the opportunity to bring the organization any closer to a system solution. Architectures lay out constraints; they must take away some decisions from those who have been accustomed to making any decision they desire. But architectures do this to impose a system-wide benefit, even though the decision may have local costs.

We know from bitter experience that the software problem only gets bigger and messier without architecture. You can help others see the value of solving cross-cutting issues at the architectural level.

**The Control Test**

The only justifiable reason for restricting the creative freedom of designers and implementers is demonstrable contribution to strategic and systemic properties that otherwise could not be achieved.

One approach is to insist that each architectural decision be accompanied by its rationale (at least—we also suggest documenting alternatives or counter-positions that were ruled out and why). This allows for "checks and balances" on the architecture. The rationale must show how the decision is architectural. Now, only decisions that would substantially better achieve the architecturally significant requirements, without compromising higher-priority architectural requirements, can reasonably be brought up in contention with the architectural decision.

## Empower within Scope

All this talk of "control" has no doubt caused you to bristle! You may already be fighting the notion that architecture disempowers. To some extent, it is a hard truth that architecture does place limits and it does take away some autonomy. In exchange, you get decisions that oftentimes are sub-optimal at more limited scope. This adds fuel to the resistance produced by the perception of being disempowered. But the alternative is chaotic development which, frankly, is even more disempowering. One of the benefits of a good architecture is that structural elements with well-designed interfaces become the object of focus for design and implementation, allowing work to progress on the structural units with greater autonomy—and small teams with strong ownership is the font of innovation and productivity.

Bear in mind, however, that you should only do with architecture what is absolutely necessary to achieve the key broad-scoped qualities of your system. And even then, remember that the less you ask your organization to imbibe early on, the more likely you will be to succeed over the long term. By all means, have an ambitious architectural vision, but stage your progress toward that vision. This will give the organization time to institutionalize architecture practices, rather than developing "antibodies" that will severely deter your attempts to impose any architectural control despite its benefits.