Less is More with Minimalist Architecture

Ruth Malan and Dana Bredemeyer

As an architect, you have been picked from your organization's top technical talent. Your architecture will 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.

Architecture and governance bring an element of control that can rub people the wrong way. Focus control where the payoff is highest and maximize your likelihood of success.

The solution is to take a minimalist approach to architecture—sort out your highest-priority architectural requirements, and then do the least you possibly can to achieve them! That is, keep your architecture decision set as small as possible, while ensuring that the technical staff meets key system priorities.

ARCHITECTURAL DECISIONS

Architectural decisions are those that must be made from an overall system perspective. Essentially, these decisions identify the system's key structural elements, the externally visible properties of these elements, and their relationships (Len Bass, Paul Clements, and Rick Kazman, *Software Architecture in Practice*, Addison-Wesley, 1997), and they define how to achieve the architecturally significant requirements. If you can achieve the requirement by deferring the decision to a lower level, it is not architecturally significant, and the decision is not an architectural one (at the given level of scope). Figure 1 (on p. 46) shows typical levels of scope found in an organization.

For example, if the system under consideration is an individual application, you should defer any decisions that component designers or implementers can make to them; these decisions should not become part of the architecture.

If the architecture's scope is a family of applications (or a product line), then you should defer any decision that relates to only a single application (or product). Such a decision belongs at the level of the application architecture and is not part of the application family's architecture.

This definition requires further elaboration about what decisions are architectural. These decisions certainly include those that relate to identifying the system's structural elements or designing the interfaces among elements, including the specification of externally visible behavior and properties. Architectural decisions concern

- maintaining system integrity—a single, unified overall design, form, or structure; and
- crosscutting concerns or system properties.

Some decisions might not involve the system's high-level structures yet affect the architecture's integrity. Other decisions addressing crosscutting concerns cannot be made from the isolated perspective of someone with a narrow focus of responsibility. These types of decisions are architectural.

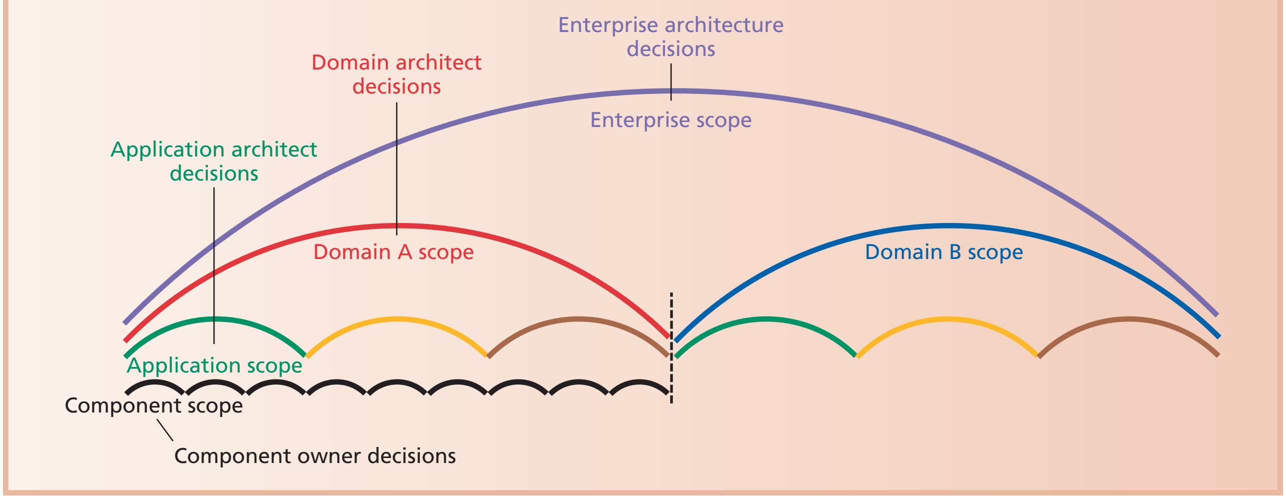
Remember, however, that the only justifiable reason for restricting the intellectual freedom of designers and implementers is demonstrable contribution to strategic and systemic properties that the organization could not otherwise achieve. Architects are highly valuable, essential technical assets in any company, and they should not squander their attention on decisions that are not truly architectural. Similarly, designers and implementers are also part of the critical capacity to produce innovation and value. An archi-

Continued on page 48

1520-9202/02/$10.00 © 2002 IEEE

Continued from page 48

Figure 1. Architectural levels of scope.


Enterprise architecture decisions
Domain architect decisions
Enterprise scope
Application architect decisions
Domain A scope
Domain B scope
Application scope
Component scope
Component owner decisions

tecture should not unnecessarily restrict their ability to do so, but rather appropriately channel efforts to fulfill the architectural vision and the business strategy it implements.

LIMITING ARCHITECTURAL CONTROL

Architects have a unique vantage point. Having responsibility across the whole system, they can solve problems in a way not available to component designers and implementers. Lack of visibility into other parts of the system, schedule crunch, and communication overload—many factors cause developers to make decisions optimized for a local scope and often suboptimal for the overall system. This situation becomes worse when you are talking about components for use or reuse across multiple systems.

This is pretty heady stuff: Your decisions, as an architect, impact the business' 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's decision set can potentially dilute the impact of all the other decisions. 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. The organization will be less likely to embrace a large architecture. The more you restrict the creative freedom that development teams have traditionally had, the more they will resist you.

So, in addition to every other tough job you have as an architect, you must figure out the right balance of architecture and anarchy for your organization. Starting out, it is better to err on the conservative side, implementing less, yet taking a few bold steps where they will make a clear difference. Down the road, when architecture is an institutionalized practice, you might be able to dispense with this consideration. For now, however, keep in mind that architecture changes the way people must work, so

Circulation: IT Professional (ISSN 1520-9202) is published bimonthly by the IEEE Computer Society. IEEE Headquarters, Three Park Avenue, 17th Floor, New York, NY 10016-5997; IEEE Computer Society Publications Office, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720-1314; voice +714 821 8380; fax +714 821 4010; IEEE Computer Society Headquarters, 1730 Massachusetts Ave. NW, Washington, DC 20036-1903. Annual subscription: $38 in addition to any IEEE Computer Society dues. Nonmember rates are available on request. Back issues: $10 for members, $20 for nonmembers. This magazine is also available on microfiche.

Postmaster: Send address changes and undelivered copies to IT Professional, IEEE Service Center, 445 Hoes Lane, Piscataway, NJ 08855. Periodicals Postage Paid at New York, N.Y., and at additional mailing offices. Canadian GST #125634188. Canada Post Publications Mail (Canadian Distribution) Agreement Number 1445669. Printed in USA.

Editorial: Unless otherwise stated, bylined articles, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in IT Professional does not necessarily constitute endorsement by the IEEE or the Computer Society. All submissions are subject to editing for style, clarity, and space.

IT Pro

don't try to do too much, too quickly.

INSISTING ON ARCHITECTURAL CONTROL

Now let's go to the other extreme: The architect (or governance organization) who avoids "laying down the law" at all costs, misses the opportunity to bring the organization any closer to a system solution. Architectures set constraints; they must take away some decisions from those accustomed to making any decision they see fit. But architectures do this to impose a systemwide 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 crosscutting issues at the architectural level.

Keep your architecture decision set to the minimum needed to achieve the most strategic architectural goals. Then work with all levels of management to ensure that these decisions stick.

In doing so, rather than taking an authoritarian approach, you can help others see value by showing them how crosscutting concerns they care about are addressed at the architectural level.

One approach is to insist that each architectural decision has a documented rationale. This allows for checks and balances on the architecture. The rationale must show how the decision is architectural. Now, anyone challenging your decision can only bring up alternatives that would substantially better achieve the architecturally significant requirements, without compromising higher-priority requirements.

EMPOWER WITHIN SCOPE

All this talk of control has no doubt caused you to bristle. You might already be fighting the notion that architecture disempowers. To some extent, it is a hard truth that architecture does place limits and take away some autonomy. In exchange, some people perceive (rightly) that architecturally based decisions are often suboptimal for narrowly focused interests. These realities add fuel to the resistance produced by the perception of being disempowered.

But the alternative is chaotic development that, frankly, is even more disempowering. One of the benefits of a good architecture is that structural elements with well-designed interfaces become the focus for design and implementation. This lets work progress on the structural units with greater autonomy—and small teams with strong ownership are the font of innovation and productivity.

Bear in mind, however, that you should only do with architecture what is absolutely necessary to achieve key, broadly scoped qualities for your system. And even then, remember that the less you ask your organization to imbibe early on, the more likely you are to succeed over the long term.

By all means, have an ambitious architectural vision, but stage your progress toward that vision. A smaller, more focused architecture costs less, is quicker to produce, and gets results faster. And, a slower rollout gives 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. n

Ruth Malan *is a principal consultant at Bredemeyer Consulting. Contact her at ruth_malan@bredemeyer.com.*

Dana Bredemeyer *is president of Bredemeyer Consulting, Bloomington, Indiana; http://www.bredemeyer.com.*

For further information on this or any other computing topic, visit our Digital Library at http://computer.org/publications/dlib.