Editor: **Eoin Woods**
Endava
eoin.woods@endava.com

What's the Architect's Role in an Agile, Cloud-Centric World?

Murat Erder and Pierre Pureur

THE DELIVERY SPEED expected from enterprise IT groups is significantly increasing. Simultaneously, technology's overwhelming expansion into daily life is driving end users' expectations of accessibility and 24/7 availability.

As a result, the software architecture pendulum is swinging away from traditional practices—particularly enterprise architecture—and toward agile and continuous practices. The gap between a commercial enterprise's agile delivery and architecture practices is wider than ever. Our book *Continuous Architecture: Sustainable Architecture in an Agile and Cloud-Centric World* discussed how to address this gap.[1] Specifically, we, as practicing architects, used our marketplace observations and industry research to describe the software architect's evolving role. This article summarizes our insights.

The Architect in Industry

The architect's role in the world of technology, while widely accepted, isn't clearly defined. Let's look at some traditional views and how architects function in agile environments.

Traditional Practice

In general, the term "architect" exemplifies the individuals who make key decisions about a system's structure and behavior, be it software or another technology area. Although "software architect" is the most common term, there are also network architects, storage architects, and information and security architects. Any technology area likely has people who consider themselves architects.

Architects share these characteristics:

- They have a deep knowledge in the domain of interest.
- They deal with higher levels of abstraction than the executable code by defining a system's structure in the domain of interest.
- They have explicit or implicit authority to make decisions in the domain of interest.

In *The Mythical Man-Month: Essays on Software Engineering*, Fred Brooks offered a good starting point for defining the architect's role: architects are accountable for the conceptual integrity of the entity being architected.[2]

Architects deal with a higher level of abstraction than do implementers, developers, or engineers. They are concerned with the main system components as well as the component interfaces and interactions. They must be able to traverse across concepts, models, and implementation details.

0740-7459/16/$33.00 © 2016 IEEE

Pre-agile practices	Agile practices
• Analyze and define requirements. • Design the system. • Serve as the interface to key business stakeholders and customers. • Bid on work for teams. • Understand how the entire system works.	• Analyze architecture epics. • Collaborate with business stakeholders and development teams. • Get implementation feedback and development estimates from teams. • Prevent teams from being side-tracked by changing business priorities. • Filter potential changes through an analysis funnel. • Maintain system models, and model future states on the basis of new epics.

FIGURE 1. The architect's responsibilities and accountabilities in pre-agile versus agile contexts.[3] An architecture epic is a large technology initiative to improve technology platforms that deliver business functionality (www.scaledagileframework.com/enablers).

Agile Practice

The architect's role in agile development is hotly debated. Some agile practitioners believe that an overall architecture isn't necessary because the architecture "emerges" through the implementation of stories and refactoring. We believe that this applies only to simple, self-contained projects. If you're architecting a product or operating at a large scale, you must consider the architecture. Although you don't necessarily need to involve a separate architect, the architect's role is still crucial to your project's success.

Figure 1 compares the architect's responsibilities and accountabilities in pre-agile and agile approaches, on the basis of Dean Leffingwell's work.[1,3]

Defining the Architect's Role

On the basis of our observations and research, we now turn to a few aspects that will help us define the architect's role.

Unit of Work

One way to evaluate architects is by their key unit of work. Rather than fancy diagrams, logical models, or running prototypes, we believe that an architect's basic unit of work is the *architectural decision*. Any architectural activity's most important output is the set of decisions made along the software development journey. Surprisingly, most organizations expend little effort on making and documenting architectural decisions consistently and clearly. So, we propose that an architect's key responsibility is to drive architectural decisions to a conclusion.

We purposely use the phrase "drive to a conclusion" because it implies that architects engage with stakeholders and work through the options to arrive at a decision. Architects don't make architectural decisions independently but facilitate decision making and ensure timely decisions.

Product Focus

Architects should focus on architecting products, not just solutions for stand-alone projects. This means that architects must take the long view, because products and software vastly outlive projects. A product-oriented approach also lets architects focus on stakeholder needs, as opposed to the schedule and budget. So, it follows that architects should operate as part of a product team; consequently, their communication and collaboration skills become highly relevant.

In the context of a single product, architects are key product team members. Their main responsibility is to balance the client's demands as represented by the product manager, with the delivery focus provided by the product delivery manager (see Figure 2). Architects must ensure a coherent, sustainable product architecture. Especially in the product life cycle's initial phases, negotiation often occurs between product management

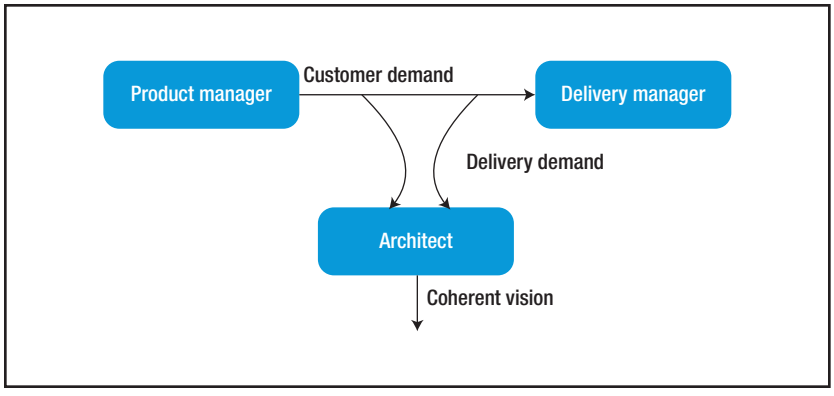

FIGURE 2. As product team members, architects must balance client demands as represented by the product manager, with the delivery focus provided by the product delivery manager.

and the "art of the possible." This might lead architects to proactively feed possibilities to product management. Managing this conversation is an essential architect responsibility.

Realized Architecture

Architects also must continuously incorporate feedback into new versions of the architecture. This architecture is represented by the code running on the physical infrastructure—what we call the *realized architecture*. We believe that the days of architecture as represented by a set of documents are over. The architect's role is to understand, influence, improve, and communicate this realized architecture.

So, architects need to not only understand the code running on the physical architecture but also be able to write code when necessary. Executing code is often the best way to settle an argument between two architects!

Final Definition

We summarize our perspective with the following definition of the architect's role:

> *The architect enables a software product's implementation by driving architectural decisions in a manner that protects the product's conceptual integrity.*

Making the Role Successful

Besides having the right technical skills, architects must display skills in the following critical areas.

Decision Making

Architects must think in terms of a minimum viable architecture. This architecture starts with a small number of decisions based on facts, not guesses, and expands only when absolutely necessary. When a decision is required, architects must act quickly to facilitate decision making. Organizations value architects for being able to make the correct decisions in ambiguous contexts. That ability stems from architects' experience (most are senior technologists with significant experience) and knowledge of applying the relevant tools at the appropriate time.

Communication and Collaboration

Architects can't overcommunicate. More than 50 percent of architects' time must focus on effective communication and collaboration.

Specialized architects (business, information, application, infrastructure, solution, and enterprise architects) tend to work in silos—hence the "ivory tower" label often used to describe architecture groups. Architects must share their knowledge rather than keep it within their group. In addition, getting outside-in views (including views from other industries) is important. Architects are encouraged to look outside their respective competency domain for new ideas they can leverage for various enterprise projects.

Delivery Team Participation

In a continuous-delivery project, the architects' overall goal is to clear roadblocks and give the delivery teams the resources they need to get things done. This includes proving that new technologies will actually work by creating executable prototypes, perhaps as part of an *architectural runway*.[4]

Test and Deployment

In this age of continuous delivery and DevOps, it's no longer sufficient for architects to focus only on software architecture. Architects should also design systems that can be incrementally tested (including system, quality assurance, and performance tests). This will require architects to fully understand the testing processes. In addition, architects must design systems that can be easily and quickly deployed to the various environments (development, test, and production) and, potentially, on internal and external clouds. This will require architects to have a good grasp of the deployment processes.

The architect's role will likely evolve from that of a specialist in the

traditional architecture domains to that of a generalist, not unlike a solution architect's role in a project.

What about enterprise architects? There's definitely a need for roles operating at this scale. However, the focus shouldn't be on creating artifacts with no immediate relevance for the realized architecture. Instead, enterprise architects should focus on creating a product-centric architecture that leverages enterprise services and on engaging more effectively with delivery teams. Traditional architects tend to produce a lot of e-paper. We recommend that they emphasize implementable code to avoid BARF (Big Architecture Up Front) syndrome.

Finally, architects should cycle through different delivery teams over time. The more experience architects have, the better they can facilitate architectural decisions. A key way to help them gain this experience is to expose them to different problem and technology domains. Deliberately cycling architects through different development teams benefits both the individuals and the enterprise. We also encourage architects to minimize multitasking across simultaneous projects, which is counterproductive.

The architect's role has changed significantly. To be successful in this new world, architects must move away from their traditional role. They must emphasize products, drive decisions, focus on the entire software delivery life cycle, understand the code produced by the team, and operate in a more decentralized fashion. In addition, nontechnical skills such as communication and the ability to operate in ambiguous contexts are increasingly critical.

References

1. M. Erder and P. Pureur, *Continuous Architecture: Sustainable Architecture in an Agile and Cloud-Centric World*, Morgan Kaufman, 2015.
2. F.P. Brooks Jr., *The Mythical Man-Month: Essays on Software Engineering*, Addison-Wesley, 1975.
3. D. Leffingwell, *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*, Addison-Wesley, 2011.
4. "Architectural Runway Abstract," Scaled Agile Inc., 21 Apr. 2016; www.scaledagileframework.com/architectural-runway.

MURAT ERDER is the director of IT architecture and engineering for a leading financial-services organization. Contact him at m_erder@yahoo.com.

PIERRE PUREUR is an enterprise architect at a Fortune 500 Company. Contact him at pierre.pureur@gmail.com.