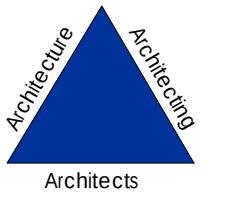

Technology
Architect Competency Elaboration

As an architect, you need a thorough knowledge of your organization's product domain, relevant technologies and development processes. But even in the technical area, your key activities are different than those of the developers. The problems are less well-defined, often with unclear or conflicting objectives, and you play a significant role in clarifying what the objectives are. Your focus is more on the implications of organizational objectives on technical choices. You take an overall system view. You are building models of the problem and solution space, exploring alternative approaches, preparing documents and explaining the architecture to sponsors and stakeholders.

The personal characteristics really essential to success in this domain are a high tolerance for ambiguity and a lot of skill working consistently at an abstract level. We know of at least one case where an otherwise qualified junior architect did not get the senior architect position because of his need for clear and unambiguous objectives.

by Ruth Malan and Dana Bredemeyer
Bredemeyer Consulting
ruth_malan@bredemeyer.com
dana@bredemeyer.com

Competency: Technology

Background

As an architect, you need a thorough knowledge of your organization's product domain, relevant technologies and development processes. But even in the technical area, your key activities are different than those of the developers. The problems are less well-defined, often with unclear or conflicting objectives, and you play a significant role in clarifying what the objectives are. Your focus is more on the implications of organizational objectives on technical choices. You take an overall system view. You are building models of the problem and solution space, exploring alternative approaches, preparing documents and explaining the architecture to sponsors and stakeholders.

The personal characteristics really essential to success in this domain are a high tolerance for ambiguity and a lot of skill working consistently at an abstract level. We know of at least one case where an otherwise qualified junior architect did not get the senior architect position because of his need for clear and unambiguous objectives.

What You Know: Technology Expertise

Level 1	Level 2	Level 3	Level 4
Technology Specialist: indepth knowledge of specific technologies and how to apply them. Applies the technology in day-to-day work.	Technology Generalist: has a broad understanding of an array of technologies and their use. Prototypes and experiments with technologies. Is well-respected by developers for technology insight and technical ability.	Technology Watch: actively scans for technologies that offer new opportunities to differentiate. Technology Watchdog: assesses the relative merits of technology maturity against business need. Is a credible technical expert, having had broad and deep experience applying a number of different technologies. Does not necessarily work at the code level in day-to-day work, but works closely enough with those that do to understand key elements of critical technologies.	Technical thought-leader: invents new technologies, applications of technology, or ways to synthesize technologies and domain understanding to forge new market opportunities. Has excelled as a developer, applying different technologies in a variety of projects in relevant domains. Does not work at the code level in day-to-day work. Draws on past technology experience to quickly integrate the key principles and issues with new technologies.

What You Know: Systems Experience

Has broad and rich experience building systems in the domain, with good insight into various aspects of the systems as well as the perspectives of users, managers, developers and other stakeholders of these systems.

Level 1	Level 2	Level 3	Level 4
Contributes to the development of a system.	Has had the experience of owning some significant aspect of the product/application design and implementation. Has an in-depth understanding of the product/application domain.	Has had the experience of architecting a product/application, or managing a product development project. Has broad experience, having worked on different aspects of the system or other systems. Has a good understanding of the products or services of the business unit.	Has had experience creating more than one architecture in a complex organizational and technical setting. Has broad experience and can see from multiple perspectives, having worked in various roles on multiple development and architecting projects. Has a good understanding of the products or services of the enterprise.

What You Do: Create and Document Architecture

Architects create architectures. This includes articulating the architectural vision, conceptualizing and experimenting with alternative architectural approaches, creating models and component and interface specification documents, and validating the architecture against requirements and assumptions.

Level 1	Level 2	Level 3	Level 4
Owns the creation of a component, module or chunk of the system. Understands that component in detail, and its context in the architecture. Takes a component-centric viewpoint, optimizing the component within the constraints of its agreed interfaces and service-level agreements.	Leads the creation of an architecture for a product/application. Creates a "big-picture" view of the product design (conceptual architecture). Decomposes the system into components or modules, and specifies the components and their interfaces in precise, unambiguous and actionable terms. Designs mechanisms to address cross-cutting system concerns. Sets technical priorities for the architecture. Makes trade-offs to accomplish the system-wide properties (cross-cutting concerns) of the product. Prepares architecture documents and presentations. Clearly describes the architecture, including rationale for decisions and implications of decisions. Takes a system viewpoint, optimizing the architecture across the components of the system.	Leads the creation of an architecture for multiple products or applications in a product line/family or solution set. Creates a big-picture design view of a set of products within a family of products (portfolio). Makes trade-offs across the product family, recognizing that local products may have to accept "good enough" in order to optimize for the family as a whole. Designs the architecture to accomplish cross-system objectives such as reuse, integration and consistency. Prepares architecture documents and technical papers explaining and motivating the approaches to be taken in the architectures of multiple systems. Takes a product family or system-of-systems viewpoint, optimizing the architecture across multiple applications or products.	Leads the creation of architectural strategy for the enterprise, making architectural decisions that have impact across the company. Negotiates and sets priorities across product families (portfolios). Defines architecture principles, styles and standards for systems across the enterprise, and creates architectural mechanisms to address concerns that have broad impact (e.g., system integration across the enterprise). Prepares and documents the enterprise architecture vision and strategy, as well as key approaches to broad architectural concerns impacting various areas of the company. Takes an enterprise viewpoint, optimizing the architectural strategy across various families or portfolios of applications or products in the enterprise.

What You Are: Abstractionist

Architecture provides an overall view of a complex system. Architects thus need to be good at working at an abstract level, and they need to be able to set technical direction in the context of much uncertainty and ambiguity.

Level 1	Level 2	Level 3	Level 4
Deals effectively with well-defined problems. Needs clear objectives. Works effectively at a concrete level.	Deals effectively with problems that are less well-defined, often with unclear or conflicting objectives. Is comfortable creating and dealing with system abstractions.	Is tolerant of significant ambiguity and poorly defined objectives. Is able to move forward in the face of uncertainty, recognizing that backtracking may be necessary. Is comfortable working at a high level of abstraction, but is able to work at the detailed technical level as necessary.	Is tolerant of high degrees of ambiguity, and helps to create the vision and strategy that will resolve uncertainy and set direction. Is very good at working at an abstract level, and at creating abstractions that clarify and contribute to system integrity across the enterpise.

What You Are: Intelligent and Quick

Architects cannot work on every part of the system and know every detail, so they need to be very intelligent and quick to grasp technical issues. They must have built and maintain respect for their technical ability, so that they are credible in the technical and management communities. They also need to have the intellectual horsepower to come up with good solutions to system-level problems.

Level 1	Level 2	Level 3	Level 4
Is able to apply known solutions in novel ways. Learns new solution techniques for known problems easily.	Is well-respected for technical skill and ability to resolve technical issues. Picks up new technical skills with relative ease. Is creative and investigative but practical.	Has credibility with technical experts because he/she is quick to grasp the key issues. Is able to develop sound strategies to solve technical problems and address cross-cutting concerns.	Highly respected internally and externally as a sharp technical thinker who quickly grasps technical implications. Is innovative and able to make technical leaps, finding novel solutions.

Self-Assessment Questions

- Can I deal with ambiguity and uncertainty, or do I have a strong preference for well-defined, clear objectives that I can deliver on?
- Can I juggle multiple, conflicting objectives?
- Do I have the experience to see the whole system, and make tradeoffs across the system?
- Do I stand out as someone who excels at technical problem solving at the system level?
- Do I understand which concerns impact multiple areas of the system, and am I able to propose good solutions to them? Do others see value in the solutions I propose?

Learning on the Job

- Identify someone who has more experience than you, and is respected for their system design prowess. Talk with them about how they approach system design, and watch what they do. Go to them for advice on specific system problems, to get a better solution, and so that you can learn from the way they solve system problems. Ask them to go into their solution process as well as the solution.
- Develop a network of peers, and share knowledge, insights, and approaches with them.
- Identify external experts, and follow their writing, hold discussions groups to share the work and the insights from it, bring in the expert for consulting.

Training

General guidelines

- Identify good conferences for technology scanning and tracking
- Identify experts and host lunch-time talks, training classes, etc.

Specific suggestions

- Software Architecture Workshops from Bredemeyer Consulting. See http://www.bredemeyer.com/training.htm

Recommended Reading

- Bass, L., P. Clements, and R. Kazman. *Software Architecture in Practice*. Prentice-Hall 1998.
- Booch, G., I. Jacobson and J. Rumbaugh, *The Unified Modeling Language User Guide.* Addison-Wesley, 1999, pp. 219-241.
- Bosch, Jan, *Design and Use of Software Architectures: Adopting and Evolving a Product-line Approach*, Addison-Wesley, 2000.
- Buschmann, Frank, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, *Pattern-Oriented Architecture: A System of Patterns*. Addison-Wesley, 1996.
- Cheesman, John and John Daniels, *UML Components*, 2000.
- Clements, Paul, R. Kazman, M. Klein, *Evaluating Software Architectures: Methods and Case Studies*, Addison-Wesley, 2001.
- Dikel, David, D. Kane, and J. Wilson, *Software Architecture: Organizational Principles and Patterns*, Prentice-Hall, 2001.
- Douglass, Bruce Powel, *Doing Hard Time: Developing Real-Time Systems with UML, Objects, Frameworks and Patterns*. Addison-Wesley, 1999. Ch. 8, pp. 367-419.
- Gamma, Erich, Richard Helm, Ralph Johnson and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.
- Hofmeister, Christine, R. Nord, and D. Soni, *Applied Software Architecture*, Addison-Wesley 2000.
- Kruchten, Philippe, "The 4+1 View Model of Software Architecture", *IEEE Software*, pp. 42-50. November 1995.

- Mark W. Maier and Eberhardt Rechtin, Eberhardt, *The Art of Systems Architecting*, 2nd edition, CRC Press, 2000.
- Malan, Ruth and Dana Bredemeyer, "Software Architecture: Central Concerns, Key Decisions", published on the *Resources for Software Architects* website at http://www.bredemeyer.com/pdf_files/ArchitectureDefinition.PDF, May 2002.
- Malan, Ruth and Dana Bredemeyer, "Less is More with Minimalist Architecture", *IEEE IT Professional*, Sept/Oct 2002.
- Malveau, Raphael, and Thomas Mowbray, *Software Architect Bootcamp*, Prentice-Hall, 2000.
- Rechtin, E. *Systems Architecting: Creating and Building Complex Systems*. Prentice-Hall, 1991.
- Schmidt, Douglas, M. Stal, H. Rohnert and F. Buschmann, *Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects*. Wiley, 2000.