# Evaluating the Software Architecture Competence of Organizations

**Len Bass, Paul Clements, Rick Kazman, Mark Klein**
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pa 15213, USA
{ljb,clements,kazman,mk}@sei.cmu.edu

## Abstract

*An organization is architecturally competent if it has the ability to acquire, use and sustain the skills and knowledge necessary to carry out architecture-related practices that lead to systems that serve the organization's business goals. This paper presents some principles of architecture competence, based on four models that aid in explaining, measuring, and improving the architecture competence of an individual or an organization with respect to these principles, The principles are based on a set of fundamental beliefs about software architecture.*

.

## 1. Introduction

When *architects* within an organization carry out architecture-related practices well, this increases the likelihood that the organization produces high-quality architectures that are aligned with its business goals. We call these architects "competent". However, such practices may be highly dependent on the specific, idiosyncratic skills and personality of the individual architect. When an *organization* is competent (by our definition), it achieves good results consistently and predictably. We believe that equipping architects with the best tools and methods will contribute to the goal of producing high-quality architectures. But more importantly we believe that improving *organizational factors* is critical to successfully adopting those tools and methods across the enterprise, and hence the key to achieving the full promise of architectures throughout the organization.

We are creating an assessment instrument for gauging organizational architectural competence (as distinct from individual competence), so that we can assess the degree to which organizations foster architectural competence, and so that we can guide organizations to become more competent. This involves evaluating not just the individuals in an organization but also the organizational practices that support architecture based development.

We adopt the following definition of architecture competence of an organization:

> *DEFINITION: The architecture competence of an organization is the ability of that organization to grow, use and sustain the skills and knowledge necessary to effectively carry out architecture-centric practices at the individual, team, and organizational levels to produce high-quality architectures that lead to systems aligned with the organization's business goals.*

In this paper we will discuss our approach to assessing the architectural competence of organizations. Our work is informed by four models that can applied to competence. These four models are:

a) *Duties, Skills, and Knowledge Model*: In [Clements 07], we first reported on a broad survey that we conducted which resulted in a categorization of Duties, Skills, and Knowledge that are useful for architects.
b) *Human Performance Theory*: Our definition of competence includes the ability to reach stakeholder-specific, situation-dependent goals. This perspective led us to adopt the Human Performance Theory of Gilbert [Gilbert 78] which emphasizes the value of an output – in this case the value of an architecture – as a measure of competence.
c) *Organizational Coordination Theory*: Software systems are developed from an architecture by teams of people coordinating. We therefore are using an aspect of Organizational Coordination Theory. In particular, we are using the concept of "congruence" or the fit between an organization's design and its ability to carry out a task [Burton 98].
d) *Organizational Learning Theory*: Organizations learn, apply, and sustain skills. Organizational Learning Theory provides a theoretical background for the learning, application, and sustainment of architecture skills [Argote 99, Argote & Todorova 07].

## 2. Fundamental Beliefs and Principles

Our approach to assessing an organization's architectural competence is based on the following three fundamental beliefs. These beliefs are difficult to empirically validate and yet they are the underpinnings of the interest in software architecture as a field, and have been observed in various forms by multiple authors.

1. The software architecture of a system is the fundamental artifact that guides development. The belief that software architecture is fundamental in the development process is the justification for the study of software architecture as a discipline.

0-7695-3092-3/08 $25.00 © 2008 IEEE
DOI 10.1109/WICSA.2008.12



IEEE computer society

2. The design of the software architecture is based on a set of *architecturally significant requirements*. Architecturally significant requirements are a (usually small) subset of the explicitly generated requirements for a system and many of them may be implicit rather than explicit. They include run time quality attribute requirements, development time quality attribute requirements, and requirements based on the context of development such as the definition of required components.

3. The software architecture and the resulting system are developed by people within an organizational and business context. Both an organization and an architecture have structure and the alignment of these structures has a large impact, either on the design of the architecture or the design of the organization. This was first observed in 1968 by Melvin Conway [Conway 68].

We now present a set of principles on which our assessment instrument will be based. Each principle is an assertion about an artifact or its derivation that, when rigorously attended to, will improve the architectural competence of the organization. That is, an organization that adheres to a principle will be more architecturally competent than one that does not.

We can not claim that our set of principles is complete, only that it represents a useful beginning, and that each principle is based on a foundation from one or more of our beliefs and underlying models. We organize these principles around four key artifacts in an architecturally aware development process: the business goals, the requirements, the software architecture, and the implementation.

### Business Goals

*Architectural knowledge should be utilized when developing the business goals for a system*. If the software architecture is the artifact that determines whether the business goals for a development will be met, it is important that the business goals be feasible from an architectural perspective.

### Requirements

*Architecturally sensitive requirements should be explicitly identified.* If the design of the architecture is based on the architecturally sensitive requirements, then they should be made explicit so that they can be reviewed and traced back to the business goals so that design choices can be explicitly linked to them.

### Architecture

*The architecture should satisfy the architecturally significant requirements.* This is the most basic duty of an architect.

*The architecture should reflect evolutionary changes in requirements.* Since the development is happening in an organizational context, changes and evolution are inevitable. These need to be accommodated within the architecture throughout the system's life.

*The development team(s) within an organization should have access to architectural knowledge as necessary.* If the architecture is central to the development effort, then the documentation and availability of architectural knowledge is critical to the development of the architecture.

### Implementation

*The implementation should conform to the architecture*. If the architecture is the fundamental artifact that supports the business goals for a system development effort, then if the implementation does not conform to the architecture it is likely that the implementation will not support the business goals.

## 7. Building an Assessment Instrument

Clearly, an organization whose development efforts consistently satisfy the enumerated principles should be deemed more architecturally competent than one that does not. But such principles are too vague for accurate assessment. An assessment instrument must delve deeper than the above-mentioned principles, to determine whether each principle is satisfied, to what level it is satisfied, and how the various principles should be weighted to achieve an overall assessment that reflects an organization's competence. In practice, organizations vary widely on their overall competence and on their competence in individual areas. For this reason a multi-dimensional view of competence must be captured in much the same way that the continuous CMMI (Capability Maturity Model Integration [CMMI 02]) views process maturity as a set of maturity levels based on specific process areas, rather than as a single "stage" or level for the entire organization .

### Assessment Outcomes

We envision at least three sets of outcomes from assessments of architectural competence:

- There are outcomes relevant to an acquisition organization: such an organization can use an assessment of architecture competence to assess a contractor in much the same way that contractors are scrutinized with respect to their CMM level, or to choose among competing bids. All other things being equal, an acquiring organization would prefer a contractor with a higher level of architectural competence since this typically means fewer downstream problems and rework [Boehm 07]. An acquisition organization might assess the contractors directly, or hire a third party to do the assessment.

- There are outcomes relevant to service organizations: such organizations might be motivated to maintain, measure, and advertise their architectural competence as a means of attracting and retaining customers. In such a case they would typically rely on outside organizations to assess their level of competence in an objective fashion.
- Finally there are outcomes that are relevant to product builders: these organizations would be doubly motivated to assess, monitor (and, over time, increase) their level of architectural competence as it would 1) aid in advertising the quality of their products and 2) aid in their internal productivity and predictability. In fact, in these ways their motivations are aligned with those of service organizations.

### Creating an Instrument

To create the assessment instrument we have taken both top-down and bottom-up approaches. Top-down, we have generated questions from knowledge of where architecture is situated in the software and system development life cycle. For example, we know that architectures are critically influenced by quality attribute requirements, and so questions in the instrument must probe the extent to which the architect elicits, captures, and analyzes such requirements. Bottom-up, we have worked from the specific duties, skills, and knowledge in the Duties, Skills, Knowledge model, and from the components of the organizational learning, organizational coordination, and human performance models. In the bottom-up approach, we examine each category in the various models and generate questions that realize each component. Overlap is not a problem; it helps to validate and refine the questions in the instrument.

Whenever we pose a question in the assessment instrument, we will ask accompanying questions. For example:

- What is your supporting evidence?
- How sustainable is your answer over time, over different systems, and over different architects?

With this in mind, we now turn to a sample set of questions drawn from the four models. For each model we will describe the source of the question, the question itself, and sub-questions/probing questions that follow from the main question.

#### *Questions based on Duties*

Duty: Creating an architecture

Question: How do you create an architecture?

- How do you ensure that the architecture is aligned with the business goals?
- What is the input into the architecture creation process? What inputs are provided to the architect(s)?
- How does the architect validate the information provided? What does the architect do in case the input is insufficient/inadequate?

Duty: Architecture evaluation and analysis

Question: How do you evaluate and analyze an architecture?

- Are evaluations part of the normal software development life-cycle or are they done when problems are encountered?
- Is the evaluation incremental or "big bang"? How is the timing determined?
- Does the evaluation include an explicit activity relating architecture to business goals?
- What are the inputs to the evaluation? How are they validated?
- What are the outputs from an evaluation? How are the outputs of the evaluation used? Are the outputs differentiated according to impact or importance? How are the outputs validated? To whom are the various outputs communicated?

#### *Questions based on Knowledge*

Knowledge: Architecture Concepts

Question: How does your organization ensure that its architects have adequate architectural knowledge?

- How are architects trained in general knowledge of architecture?
- How do architects learn about architectural frameworks, patterns, styles, standards, documentation notations, architecture description languages?
- How do architects learn about new or emerging architectural technologies (e.g. model-driven architecture)?
- How do architects learn about analysis and evaluation techniques and methods?
- How do architects learn quality attribute-specific knowledge, such as techniques for analyzing and managing reliability, performance, maintainability, security?
- How are architects tested to ensure that their level of knowledge is adequate, and remains adequate, for the tasks that they face?

#### *Questions based on the Organizational Coordination Model*

Question: How is the architecture designed with distribution of work to teams in mind?

- How available or broadly shared is the architecture to various teams?

- How do you manage the evolution of architecture during development?
- Is the work assigned to the teams before or after the architecture is defined, and/or with due consideration of the architectural structure?

### *Questions based on the Human Performance Technology Model*

Question: Do you track how much the architecture effort costs, and how it impacts overall project cost and schedule?

Question: Do you track the "worth" of the architecture, the benefits: E.g., stakeholder satisfaction, quality, repeat business, bug reports, etc.

### *Questions based on the Organizational Learning Model*

Question: How do you capture and share experiences, lessons learned, technological decisions, techniques and methods, and knowledge about available tooling?

- Do you use any knowledge management tools?
- Is architectural knowledge embedded in your processes?
- Where is the information about "who knows what" captured and how is this information maintained?
- How complete and up to date is your architecture documentation? How widely disseminated is it?

What we have presented here is a very small selection of questions motivated by the four models that we are using to understand architectural competence in more detail. But in just this small sample, we already see several interesting results. First models inspire questions. Given that there is a duty called "Documentation", we expect to see one or more questions related to documentation arising from the differing perspectives of the four models.

But more than this straightforward mapping from the models to the questions, we can see that the intersection of the models and the combination of bottom-up and top-down works to both flesh out and to validate the questions. For example we not only get questions on documentation coming from the duties, but we also expect to get documentation questions coming from the organizational learning model since, in that model, we are concerned with how information is collected and shared and this naturally leads to a question concerning documentation.

There is still a considerable amount of structuring required to turn a set of questions into a document that will guide a competence assessment interview. But the foundations for an assessment instrument have been laid, as we have shown, by delving into our four models of competence. We are now turning our attention to fleshing out the questions and to empirical validation of the results.

## 10. Acknowledgments

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

## 11. References

[Argote 99] Argote, L. *Organizational learning: Creating, retaining and transferring knowledge*. Norwell, MA: Kluwer, 1999.

[Argote & Todorova 07] Argote, L., & Todorova, G. "Organizational learning". In G.P. Hodgkinson and J.K. Ford (Eds.), *International Review of Industrial and Organizational Psychology*, Vo. 22, pp. 193-235, 2007, Chichester, England: John Wiley &Sons, Ltd.

[Boehm 07] Boehm, B., Valerdi, R., Honour, E. 2007. "The ROI of Systems Engineering: Some Quantitative Results". In *Proceedings of INCOSE 2007*.

[Burton 98] Burton, R.M. and Obel, B. *Strategic Organizational Diagnosis and Design*. Kluwer Academic Publishers, Norwell, MA, 1998.

[Clements 07] Clements, P., Kazman, R., Klein, M., Devesh, D., Reddy, S., Verma, P., "The Duties, Skills and Knowledge of Software Architects", *Proceedings of WICSA 2007*.

[CMMI 02] *Capability Maturity Model Integration (CMMISM), Version 1.1*, Carnegie Mellon University Software Engineering Institute Technical Report CMU/SEI-2002-TR-012, 2002.

[Conway 68] Conway, M. "How do Committees Invent?", *Datamation*, April 1968.

[Gilbert 78] Gilbert, T., *Human Competence: Engineering Worthy Performance*, HRD Press, 1978.