

ISERN 2008 Software Architecture (SA) Session

Chair: G. Cantone

Facilitators: M. AliBabar, J. Carver, D. Cruzes, D. Falessi, O. Pastor, F. Shull, and G. H. Travassos

Monday 6.10.2008 14:00 – 15:00

Reading Material

Applying Empirical Software Engineering to Software Architecture

by M. AliBabar, G. Cantone, J. Carver, D. Cruzes, D. Falessi, O. Pastor, F. Shull, G. H. Travassos

***Abstract.** In this note, we first highlight some open issues that concern software architecture and its relationships with empirical studies. Subsequently, based on the time allotted to, we define a main goal and a couple of additional goals for this session. Finally, depending on the number of expected participants we provide organizational insights for the ISERN 2008 Software Architecture Session.*

1. Software Architecture Issues

1.1 Basic Issues

These issues concern some points, including:

1. What is Software Architecture?
2. Is Software Architecture relevant, and why?
3. Is Software Architecture actually designed on scientific basis or just made on the basis of the individual architect experience? (Q: Is this point suitable for ISERN participants?)
4. Is it worth to develop scientific research about Software Architecture?
5. What evaluation models are applied nowadays to software architecture development technologies?

Concerning the abovementioned points, let us present some preliminary considerations.

Software architecture has emerged [1] as an important [2] field of software engineering for managing the realm of large-system development and maintenance [3]. The main intent of software architecture is to provide intellectual control over a sophisticated system enormous complexity [4].

There are several definitions of software architecture [5]; one of the most used is the set of significant decisions about the organization of a software system: selection of the structural elements and their interfaces by which a system is composed, behavior as specified in collaborations among those elements, composition of these structural and behavioral elements into larger subsystem, architectural style that guides this organization. Software architecture also

involves usage, functionality, performance, resilience, reuse, comprehensibility, economic and technology constraints, tradeoffs, and aesthetic concerns [6].

There are several metrics regarding design quality (e.g. [7,8,9,10]); however, there is a difference between architecture and design [11]: the former artifact is part of the latter. The word software architecture is now used everywhere, reflecting a growing concern and attention. In fact, architecting is the activities of conceiving, defining, documenting, maintaining, improving, and certifying proper implementation of an architecture [12]. Thus, many design decisions are left unbound by the architecture and are happily left to the discretion and good judgment of downstream designers and implementers [3]. Each software system has a software architecture [13]; it can be implicit, i.e. accidental, or explicit i.e. documented and specifically designed to fulfill predefined business objective or non functional requirements. A discussion regarding state of the art of available software architecture design method can be found in [14], while commonalities and variability of software architecture design methods are extensively described in [15].

However, why we should care about software architecture? Software architecture is developed during the early phases of the development process; it hugely constraints or facilitates the achievement of specific functional requirements, non-functional requirements, and business goals [2]. Hence, reviewing the software architecture represents a valid means to check the conformance of the system and to reveal early any potentially missed objective [16]. There are different ways to evaluate a software architecture; such an evaluation activity can differ in: the technique adopted [17], the input given [18] to, and in the scenarios [19] adopted for, such an evaluation technique. Moreover, the level of “goodness” [20] of an architecture heavily depends on the amount of knowledge that the architects have; this problem is known as “bounded rationality” [21]. Software architecture is an artifact that the software development lifecycle delivers very early; this implies that software architecture decisions are made often based on unstable and quite vague system requirements. Hence, software architecture goodness depends on the existent level of knowledge that architects have about both the problem space (i.e. requirements understanding/changes), and the solution space (design solution characteristics): such a level is really difficult to measure, hence hard to replicate and control.

In case of maintenance operation, since software architecture is key to manage large software applications, it may be the case to spend effort in documenting the underlying software architecture [3,22] even if we have to recover it [23]. Grady Booch in [24] proposes a sound discussion regarding the economic value of focusing on software architecture concerns. Moreover, the OMG’s Model Driven Architecture [25] is becoming a standard de facto for the development of large scale projects.

1.2 Issues Related to Software Architecture Measurement

The main problem here concerns our current knowledge about empirical relational domain, i.e., if it is enough to develop the right measurement models. Related points include:

1. How do software architects actually evaluate software architecture development technologies that they use? (Q: Is this point suitable for ISERN participants?)
2. To what extend software architects and project managers rely on existing metrics? (Q: Is this point suitable for ISERN participants?)
3. Would we provide [GQM-based] (new & better) measurement models for goodness of software architecture?
4. Does it make sense, and how to evaluate an intermediate artifact (like software architecture) rather than, or without analyzing also, the final artifact (whole system)?
5. Which metric help give insight into end-product qualities like maintainability, performance, reusability?

6. What evidence do we actually have for thinking that these metrics provide such insight?

1.3 Issues Concerning Software Architecture Empirical Studies

The main issue here concerns if and in what extent it is feasible the construction of valid empirical studies on software architecture. Related points include:

1. How a study of software architecture is different from a study on any other software engineering topic?
2. How to empirically assess the usability and usefulness of software architecture technologies (e.g., Architectural description languages) within industrial settings and considering cost bounds?
3. Could we use empirical researchers as reviewers of architecture, as developed/maintained in architectural experiments, when the activity of architecture review usually requires considerable experience in the field?
4. Does it make any sense to use students (e.g. [26]) as subjects in architectural experiments when experienced people usually play in the role of architects?
5. Does it make sense to use toy objects in architectural experiments? If not, how to conduct software architecture empirical investigation in a controlled environment?

1.4 Issues Concerning Software Architecture Influential Factors

The main issue here concerns if and in what extent we are able to describe, hence to keep in control, software architecture influential factors. In particular, how to comprehensively describe things influencing software architecture, which include (but are not limited to): Evaluation technique, Evaluation input, Evaluation scenarios, Amount of knowledge (on solution space and problem space). Some of the consequential points are:

1. How to keep software architecture influential factors in control during empirical studies (constant variables, blocks, etc.)?
2. How to replicate studies on software architecture? Is it different from replicating studies in other domains?

1.5 Issues Concerning Software Architecture Study and Teaching

The main issue here concerns if, and in what extent, the way to teach software architecture influences the answers to some of the abovementioned points. Some of the consequential points are:

1. How do we teach software architecture around the different countries?
2. What the impact of such differences on the results?

1.6 Cross-cutting concerns

Some of the related points are:

1. Is there any cultural misalignment among the architecture and the empirical community? If yes, how to deal with it?
2. Are the actual software architecture challenges similar to ones experienced/resolved in other domains?

2. Goal of the ISERN Session

Our plan is to address the goal of gathering information regarding software architecture from the point of view of the ISERN members in the context of an ISERN session by focusing on (1) How a study of software architecture is similar to/different from a study on any other software engineering topic, (2) Measurement models for goodness of software architecture, and possibly (3) How we teach software architecture around the different countries. In case, further discussion groups could be started.

The result should be the basis of a report to go back to ISERN members.

3. Points to Discuss at the ISERN 2008 Meeting

As already mentioned, our idea is to focus discussion on the following bullets:

Question 1 from issue/Section 1.3, i.e.:

- ***Issues Concerning Software Architecture Empirical Studies***
 - *How a study of software architecture is different from a study on any other Software engineering topic?*

The participant members of ISERN should try to provide a discussion as evidence-based as possible. Maybe if we could discuss some of the problem that people have had in the past running experiments on software architecture, we could bring our experience from previous work in the discussion. Maybe the discussion could also afford some other questions from Section 1.3.

Maybe the discussion (and the final report) could focus on problem specific to this topic, based on past attempt to study it, and share/brainstorm some solutions that future researchers could reuse.

Question 3 from issue/Section 1.2:

- ***Issues Concerning Software Architecture Measurement***
 - *Would we provide [GQM-based] (new & better) measurement models for goodness of software architecture?*

Question 1 from issue/Section 1.4.

- ***Issues Concerning Software Architecture Study and Teaching***
 - *How do we teach software architecture around the different countries?*

Finally, further goals/questions chosen among those shown above could be included, so starting further discussion groups, based on the interest of the participants.

4. Structuring the ISERN 2008 Software Architecture Session

Based on the assumed goals and related selected questions, the SA Session should realistically come to have three discussion groups at least.

We evaluate as a convenient organization having one or more facilitators in each discussion group, and the session chair attending all groups, each in turn for a quantum of time.

The tasks of leading the SA Sessions could be conveniently assigned as in the following:

- Issues Related to Software Architecture Measurement: Daniela, Forrest [and Jeff];
- Issues Related to Software Architecture Empirical Studies: Ali, Davide [and Jeff];
- Issue Related to Software Architecture Study and Teaching; Oscar, Guilherme, [and Jeff]

In order to arrange a 10 minutes report for ISERN participants as a whole, 10 minutes should be given to session leaders, one volunteer per session, for meeting the session chair and arranging the presentation report. Hence, SA discussion group meetings should last around 40 minutes.

5. Following the ISERN 2008 Software Architecture Session: Prospective Work

The abovementioned report - as based on the Software architecture ISERN session and directed to ISERN members - should be the basis of a paper or position paper statement to realize subsequently by meetings to put in place through e-mail and/or Skype. Eventually, these e-meetings should address deriving guidelines for how to conduct software architecture empirical research.

We should define a strategy to afford the other aforementioned points.

6. References

- [1] M. Shaw, and P. Clements, "The Golden Age of Software Architecture," *IEEE Software*, vol. 23, no. 2, pp. 31-39, 2006.
- [2] G. Booch, "The Irrelevance of Architecture," *IEEE Software*, vol. 24, no. 3, pp. 10-11, 2007.
- [3] P. Clements, F. Bachmann, L. Bass *et al.*, *Documenting Software Architectures: Views and Beyond*, Boston: Addison-Wesley, 2002.
- [4] P. Kruchten, H. Obbink, and J. Stafford, "The past, present and future for software architecture," *IEEE Software*, vol. 23, no. 2, pp. 2-10, March-April, 2006.
- [5] SEI, "Published Software Architecture Definitions," http://www.sei.cmu.edu/architecture/published_definitions.html, 2007].
- [6] P. Kruchten, *The Rational Unified Process: An Introduction* 3rd ed.: Addison-Wesley Professional, 2003.

- [7] D. Rombach, "A Controlled Experiment on the Impact of Software Structure on Maintainability," *IEEE Trans. Softw. Eng.*, vol. 13, no. 3, pp. 344-354, 1987.
- [8] L. Briand, J. Wust, J. Daly, and V. Porter, "A Comprehensive Empirical Validation of Design Measures for Object-Oriented Systems," in Proceedings of the 5th International Symposium on Software Metrics, 1998.
- [9] V. Basili, L. Briand, and W. Melo, "A Validation of Object-Oriented Design Metrics as Quality Indicators," *IEEE Trans. Softw. Eng.*, vol. 22, no. 10, pp. 751-761, 1996.
- [10] G. Succi, W. Pedrycz, S. Djokic, P. Zuliani, and B. Russo, "An Empirical Exploration of the Distributions of the Chidamber and Kemerer Object-Oriented Metrics Suite," *Empirical Softw. Engg.*, vol. 10, no. 1, pp. 81-104, 2005.
- [11] A. H. Eden, and R. Kazman, "Architecture, design, implementation," in Proceedings of the 25th International Conference on Software Engineering, Portland, Oregon, 2003.
- [12] IEEE, *IEEE std 1471:2000--Recommended practice for architectural description of software intensive systems*, Los Alamitos, CA: IEEE, 2000.
- [13] G. Booch, "The Accidental Architecture," *IEEE Software*, vol. 23, no. 3, pp. 9-11, 2006.
- [14] D. Falessi, G. Cantone, and P. Kruchten, "Do Architecture Design Methods Meet Architects' Needs?," in Proceedings of the 6th Working IEEE/IFIP Conference on Software Architecture, Mumbai, India, 2007.
- [15] C. Hofmeister, P. Kruchten, R. L. Nord, H. Obbink, A. Ran, and P. America, "A general model of software architecture design derived from five industrial approaches," *Journal of Systems and Software*, vol. 80, no. 1, pp. 106-126, 2007.
- [16] J. F. Maranzano, S. A. Rozsypal, G. H. Zimmerman, G. W. Warnken, P. E. Wirth, and D. M. Weiss, "Architecture Reviews: Practice and Experience," *IEEE Software*, vol. 22, no. 2, pp. 34-43, march/april, 2005.
- [17] M. A. Babar, L. Zhu, and R. Jeffery, "A framework for classifying and comparing software architecture evaluation methods," in Proceedings of the Australian Software Engineering Conference. , 2004.
- [18] H. Obbink, P. Kruchten, W. Kozaczynski *et al.*, *Report on Software Architecture Review and Assessment (SARA), Version 1.0*. At <http://philippe.kruchten.com/architecture/SARAv1.pdf>, 2002.
- [19] R. Kazman, S. J. Carriere, and S. G. Woods, "Toward a discipline of scenario-based architectural engineering," *Annals of Software Engineering*, vol. 9, no. 1-4, pp. 5-33, 2000.
- [20] G. Booch, "Goodness of Fit," *IEEE Software*, vol. 23, no. 6, pp. 14-15, Nov/Dec, 2006.
- [21] H. Simon, *The Sciences of the Artificial* 3ed., Cambridge, Mass.: The MIT Press, 1996.
- [22] N. Rozanski, and E. Woods, *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*, Boston: Addison-Wesley, 2005.
- [23] L. Ding, and N. Medvidovic, "Focus: A Light-Weight, Incremental Approach to Software Architecture Recovery and Evolution," in Proceedings of the Working IEEE/IFIP Conference on Software Architecture (WICSA'01) - Volume 00, 2001.
- [24] G. Booch, "The Economics of Architecture-First," *IEEE Software*, vol. 24, no. 5, pp. 18-20, 2007.
- [25] O. Pastor, and J. Molina, *Model-Driven Architecture in Practice: A Software Production Environment Based on Conceptual Modeling*: Springer-Verlag New York, Inc., 2007.
- [26] B. Williams, and J. Carver, "Characterizing Software Architecture Changes: An Initial Study," in Proceedings of the First International Symposium on Empirical Software Engineering and Measurement, 2007.