

Empirical research in software architecture: opportunities, challenges, and approaches

Muhammad Ali Babar · Patricia Lago ·
Arie Van Deursen

© The Author(s) 2011. This article is published with open access at Springerlink.com

1 Introduction

Software architecture has become an important area of research and practice since the late 1980s (Shaw and Clements 2006). The term “software architecture” started gaining acceptance by the software engineering community in early 1990s but the foundations of this field were laid by the seminal work of Edsger Dijkstra, David Parnas, and others between 1960s and 1980s (Clements 2000). The increasing size and complexity of software systems and demand for high quality are some of the most important factors that have driven the increased interest in this sub-discipline of software engineering. It has generally been realized that a high-level design description can play an important role in successfully understanding and managing large and complex software systems (Clements et al. 2002; Lung and Kalaichelvan 2000). The high-level design decisions regarding the software architecture of a system are not only the hardest and most expensive to change but also play a fundamental role in setting the boundaries for the required quality attributes such as maintainability, reliability, usability, performance, and flexibility of a system (Bass et al. 2003; Clement and Northrop 1996).

As a result of increasing realization of the important role of software architecture in large-scale software development and evolution projects, the software architecture community has developed several methods, techniques, and tools to support the software

M. A. Babar (✉)
IT University of Copenhagen, Copenhagen, Denmark
e-mail: malibaba@itu.dk

P. Lago
VU University Amsterdam, Amsterdam, the Netherlands

P. Lago
e-mail: p.lago@cs.vu.nl

A. Van Deursen
Delft University of Technology, Delft, the Netherlands

A. Van Deursen
e-mail: Arie.vanDeursen@tudelft.nl

architecture process. However, excluding a few exceptions, there has been little effort to systemically gathering, rigorously analysing, and widely disseminating empirical evidence to support the claimed benefits and capabilities of specific methods, techniques, and tools developed for supporting software architecture (Falessi et al. 2010). What is usually presented as evaluation is either an anecdotal claim by a technology developer based on a study with a small case or the testimonial from an industrial evangelist who is willing to vouch the efficiency and effectiveness of a particular method or tool after applying it to some of his/her projects. This situation gives credence to the claims that there is a dearth of literature reporting high quality empirical research for evaluating software architecture technologies. However, there is growing demand for systematically gathered evidence rather than anecdotes or rhetoric to promote the use of a particular method or tool that purports to support any of the software engineering activity (Dyba et al. 2005; Oates 2004).

Hence, there is a vital need for gathering and disseminating empirical evidence to help researchers to assess current research and identify promising future research areas, and practitioners to choose appropriate methods and techniques for supporting the software architecture process. Given this kind of the state of the art in terms of empirical evaluation of software architecture technologies (i.e., processes, methods, and tools), we assert that one of the main research goals of the software architecture community should be to systematically design, rigorously execute, and diligently report high quality empirical studies assessing different aspects of software architecture technologies using different research approaches and data generation methods and following the principles of evidence-based paradigm (Dyba et al. 2005). Such an effort should leverage the approaches from both positivist and interpretivist research disciplines to provide a solid form of evidence in support of the claims made in favour or against a particular technology (Falessi et al. 2010). Some of the main research methods for this kind of research can be controlled experiments, case studies, surveys (i.e., interviews and questionnaires), ethnographically inspired field studies, expert opinion, and systematic literature reviews (Montesi and Lago 2008).

This special issue aims at increasing the recognition of the importance and value of empirical research as an objective and structured means of assembling and analysing the available data in order to identify and answer the most significant research questions about the effectiveness and efficiencies of different technologies being proposed and/or developed to support the process of designing, evaluating, implementing, and evolving software architectures of large scale systems as well as the architectural artefacts. For this special issue, we have selected four papers which have been briefly introduced in the following paragraphs.

Trosky B. Callo Arias, Pieter van der Spek and Paris Avgeriou in “A practice-Driven Systematic Review of Dependency Analysis Solutions” report a systematic literature review on dependency analysis solutions. This work combines problems and theories emerging from industrial practice with an empirical research method typically applied to academic research, systematic literature review that is aimed at supporting evidence-based decision making by software development practitioners. Thanks to this combination the contribution of this article is for both practitioners and researchers. They can take it as a reference to learn about dependency analysis, match their own practice to the presented results, and build similar overviews of other techniques and methods for other domains or types of systems.

In their article titled “From Monolithic to Component-based Performance Evaluation of Software Architectures—A Series of Experiments Analysing Accuracy and Effort”, Anne Martens, Heiko Kozirolek, Lutz Prechelt and Ralf Reussner report on a series of three experiments (with different levels of control) on architectural performance evaluation methods and the related applicability, level of accuracy, and effort spent. While the

experiments were carried out in academic setting, the authors discuss industrial relevance of the results, directions for future empirical research in the field as well as identify some interesting research questions to be further investigated, and make some insightful suggestions for setting up experiments.

Michel Wermelinger, Yijun Yu, Angela Lozano and Andrea Capiluppi in “Assessing Architectural Evolution: a Case Study” propose an historical perspective on the evolution of a large, well-known open source software project, the Eclipse SDK. In this case study, the authors investigate if well-established software evolution laws hold, and investigate if architectural evolution practices can be isolated from this long-lived project.

Zude Li, Nazim H. Madhavji, Syed Shariyar Murtaza, Mechelle Gittens, Andriy V. Miransky, David Godwin and Enzo Cialini in “Characteristics of Multiple-Component Defects and Architectural Hotspots: A Large System Case Study” address the crucial problem of managing defects in large software systems. The authors carried out a case study on a very large, commercial, legacy software system representing six releases over seventeen years. Results provide qualitative and quantitative evidence of the crucial role played by architectural hotspots in effectively identifying and correcting architectural defects.

These four articles in this special issue provide only some examples of applying empirical research methods in the software architecture field. We hope that many more will appear in the future, in either academic research or industrial practice, and possibly in joint academic-industrial efforts.

Acknowledgement We are thankful to all the authors who responded to our call for papers by submitting their work. We are also thankful to Prof. Lionel Briand, Editor-in-Chief of Empirical Software Engineering Journal, for accepting our proposal and supporting us during the whole process of preparing this special issue. Most importantly, we are greatly indebted to our colleagues who generously allocated their precious time and effort for helping us during the review process of selecting the papers accepted in this special issue.

Open Access This article is distributed under the terms of the Creative Commons Attribution Noncommercial License which permits any noncommercial use, distribution, and reproduction in any medium, provided the original author(s) and source are credited.

References

- Bass L, Clements P, Kazman R (2003) Software architecture in practice. Addison-Wesley
- Clement P, Northrop L (1996) Software architecture: An executive overview. CMU/SEI-96-TR-003, CMU/SEI-96-TR-003, SEI, Carnegie Mellon University
- Clements P, Kazman R, Klein M (2002) Evaluating software architectures: Methods and case studies. Addison-Wesley
- Clements PC (2000) Constructing superior software. Macmillan Technical, USA
- Dyba T, Kitchenham B, Jorgensen M (2005) Evidence-based software engineering for practitioners. IEEE Softw 22(1):58–65
- Falessi D, Ali Babar M, Cantone G, Kruchten P (2010) Applying empirical software engineering to software architecture: challenges and lessons learned. Empir Softw Eng 15(3):250–276
- Lung C-H, Kalaichelvan K (2000) An approach to quantitative software architecture sensitivity analysis. Int J Softw Eng Knowl Eng 10(1):97–114
- Montesi M, Lago P (2008) Software engineering article types: an analysis of the literature. J Syst Softw 81(10):1694–1714
- Oates BJ (2004) Widening the scope of evidence gathering in software engineering. In: Proceedings of the 11th Annual International Workshop on Software Technology and Engineering Practice, pp
- Shaw M, Clements P (2006) The golden age of software architecture. IEEE Softw 23(2):31–39



Muhammad Ali Babar is an Associate Professor at IT University of Copenhagen, Denmark. Previously, he worked as a researcher for different research institutes in Australia and Ireland. He has authored/co-authored more than 100 peer-reviewed research papers in journals, conferences, and workshops. He co-edited a book, software architecture knowledge management: theory and practice. Dr. Ali Babar has been guest editor of several special issues/sections of IEEE Software, JSS, ESEJ, SoSyM, IST, and REJ. Apart from being on the program committees of several international conferences such as WICSA/ECSA, ESEM, SPLC, ICGSE, and ICSP for several years, Dr. Ali Babar is also chairing the program committees of several conferences such as WICSA/ECSA 2012, ICGSE 2011, ECSA 2010, and PROFES 2010. He is a member of the steering committees of WICSA, ECSA and ICGSE. He has presented tutorials in the areas of software architecture and empirical approaches at various well known international conferences. Prior to joining R&D field, he worked as a software engineer and an IT consultant for several years in Australia. His current research interests include software architecture, cloud computing, global software development and evidence-based software engineering. He obtained a PhD in computer science and engineering from University of New South Wales, Australia.



Patricia Lago is associate professor at the VU University Amsterdam, The Netherlands. Her research interests are in software- and service oriented architecture, architectural knowledge management, green IT and sustainable software engineering. Lago has a PhD in Control and Computer Engineering from Politecnico di Torino. She co-edited various special issues and co-organized workshops on topics related to service-oriented and software architecture, architectural knowledge management, and green IT. She has been Program Chair of the IEEE/IFIP Working Conference on Software Architecture, and is member of IEEE and ACM, and of the International Federation for Information Processing Working Group 2.10 on Software Architecture. She is Area Editor Service Orientation for the Elsevier Journal of Systems and Software. She published over 100 peer-reviewed articles in international journals, conference proceedings and books, and co-edited the books published by Springer on Software Architecture Knowledge Management (2009) and on Relating Software Requirements and Architectures (2011).



Arie van Deursen is a full professor at Delft University of Technology, where he is heading the Software Engineering Research Group. He obtained his PhD degree from the University of Amsterdam in 1994. His research interests include software architecture, collaborative software engineering, empirical software engineering, end user programming, and software testing. He was general chair of the Working Conference on Mining Software Repositories (MSR'11) and will be program chair of the International Conference on Program Comprehension (ICPC'12).