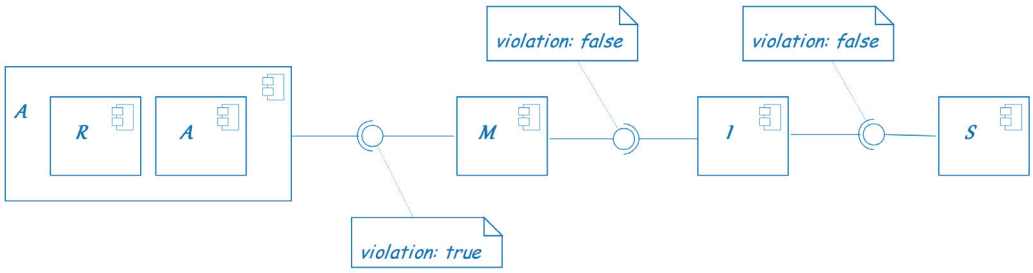
SE | RWTH AACHEN UNIVERSITY

Ana-Maria-Cristina Nicolaescu

Behavior-Based Architecture Conformance Checking



Aachener Informatik-Berichte,
Software Engineering

Band 37

Hrsg: Prof. Dr. rer. nat. Bernhard Rumpe
Prof. Dr. rer. nat. Horst Lichter

Behavior-Based Architecture Conformance Checking

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades einer Doktorin der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Ana-Maria-Cristina Nicolaescu, M.Sc.
aus Bukarest, Rumänien

Berichter: Universitätsprofessor Dr. rer. nat. Horst Lichter
Universitätsprofessor Dr. rer. nat. Claus Lewerentz

Tag der mündlichen Prüfung: 13. September 2018

Aachener Informatik-Berichte, Software Engineering

herausgegeben von
Prof. Dr. rer. nat. Bernhard Rumpe
Software Engineering
RWTH Aachen University

Band 37

Ana-Maria-Cristina Nicolaescu
RWTH Aachen University

Behavior-Based Architecture Conformance Checking

Shaker Verlag
Aachen 2018

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Zugl.: D 82 (Diss. RWTH Aachen University, 2018)

Printed in Germany.

ISBN 978-3-8440-6415-5
ISSN 1869-9170

Shaker Verlag GmbH • P.O. BOX 101818 • D-52018 Aachen
Phone: 0049/2407/9596-0 • Telefax: 0049/2407/9596-9
Internet: www.shaker.de • e-mail: info@shaker.de

Acknowledgement

Looking back to my PhD years, I become overwhelmed with happiness: it has been an incredible journey and a privilege.

First of all, I would like to express my most sincere gratitude to my doctoral father and mentor, Prof. Dr. rer. nat. Horst Lichter, who has enabled me to conduct my doctoral studies in the Research Group for Software Construction. I couldn't have concluded this work without his continuous support, careful guidance and precious advice. Throughout the years, I had the honor to know him not only as the absolute professional and perfectionist, but also as a person that became for me a constant source of inspiration and a model.

At the same time, I would like to thank Prof. Dr. rer. nat. Claus Lewerentz for having accepted to overtake the second supervision and having guided me with useful comments and improvement suggestions in the last phase of my PhD.

Dorin Moraru, my math teacher, is one of the persons that hugely impacted my life. I learned from him what true passion for one's work is. He has constantly pushed his students to become the better versions of themselves, to never settle with less and to constantly learn something new.

My grandparents Anica and Gheorghe gave me wings to fly. Grandma Anica, always very loving and gentle, made one particular issue very clear: she had big dreams for me and was not willing to accept anything less than perfect. Grandpa Gheorghe made everything humanly possible to awake in me the joy of learning and the thirst for knowledge. I wish I could show him my PhD thesis and dance with him our "dance of joy" that we invented in my childhood. They will be both forever dearly missed.

My mother is the person that supported me the most through all these years. She is my number one advocate, my model and the lighthouse towards which I turn to when I don't know further. Thank you for always having been there for me! I couldn't have finished my work without your unconditional help and love.

My father constantly reminded me that knowledge trumps possessions and my grandparents Gicu and Jeni always believed in my abilities.

Georgeta's ambition and perfectionism astonished and inspired me. Her personal example, belief in me, motherly affection contributed significantly to my becoming. I think of her every day with love and respect.

Petre has always supported and encouraged me very affectionately. Cici, Dinu, Irina, Iulian, Diana and Robert constantly cheered for me and stood by my side, ready to listen and offer good advice.

My friends Ioana, Anna, Vicktor, Marina, Miguel, Göckhan, Zied, Dejan, Alexandra, Yiwei, Sten, Dominik, Peter and Adeline made my days cheerful and also invested time to listen and give me good advice when help was needed. Ralf Klamma slowly shifted from my first chef to a great friend and a constant source of inspiration. Cornelia, Carl and Ruth helped us settle in Germany and have offered constant guidance. Last but not least, Marion, Ina and Nele made Aachen feel like home and surrounded us with love and care.

Furthermore, I would like to express my deepest gratitude to all my colleagues from the research group: Simona, Veit, Andreas, Matthias, Firdaus, Andrej, Andy, Simon, Konrad and Peter. It was a pleasure to work with you and constantly learn from you! I am very grateful for

your support and friendship!

I would like to thank my long-term industrial cooperation partner Generali Deutschland Informatik Services for having financed my PhD, involved me in real-world projects and evaluated my results. Furthermore, the company Kisters helped me to extend the scope of the evaluation and contributed significantly to the overall quality of my work.

All of my internship, master and bachelor students contributed to the completion of my PhD and I am thankful to each and every one of them. However, the happiest time of my PhD is associated with my small "ARAMIS Core Team": Artjom Göringer, Dung Le, Jan Thomas, Peter Alexander and Alexandru Sabau, who have contributed with valuable ideas and actively developed the ARAMIS Toolbox.

To end with, none of what happened in the last 13 years would have been possible without my best friend, soul-mate and husband, Petru, who accompanied me in so many adventures and happy moments and always found the right words and actions to help me out when life was not running as expected. The long discussions with him always brought me forward and gave me new insights. Petru, it is a privilege to spend my life with you, thank you for everything you are!

Last, I would like to thank our little son, Erik George. His arrival in our lives brought us tremendous happiness! Despite generously contributing to our overall sleep deprivation in the last PhD phases, Erik has showed us what life really is about and completed us as persons. My dear child, you are still so young, yet so inspiring with your pure joy and fierce ambition to achieve your goals!

Thank you!

Abstract

In this dissertation we present ARAMIS: a concept and corresponding tool support for behavior-based architecture conformance checking of software systems.

In the past years several approaches for static-based architecture conformance checking were proposed. These pose important limitations when considering modern systems, typically composed of several interacting processes. Ever since the advent of object orientation and due to the shift from monolith architectures to componentized ones, the complexity of software systems has moved from structure to behavior. This is typically out of the scope of static-based conformance approaches, which face an impossibility in assessing if the system under analysis is behaving as foreseen by its architects.

ARAMIS is our solution to alleviating the above-mentioned problem. First, the intended architecture description of the system under analysis is expressed using an ARAMIS-specific meta-model. This encompasses the architecture units constituting the system and the communication rules governing these. To increase acceptance, model-engineering techniques are also proposed to enable the reuse of intended architecture descriptions elaborated using different meta-models than that of ARAMIS. Next, interactions are extracted during the system's execution using third party monitoring tools. Given that a holistic analysis of the behavior of a system is impossible in general, we proposed several indicators to assess whether the captured interactions represent an adequate basis for checking the conformance of the system as a whole. The interactions are consequently elevated to depict communication between the units defined in the system's intended architecture description and validated to check their conformance to the formulated communication rules. The results constitute a description of the implemented architecture of the system, characterized by its drift from the intended one. This can subsequently be explored using several mechanisms such as user-defined architecture views and perspectives or dedicated visualizations. Last but not least, processes for guiding the activities involved in behavior-based conformance checking were developed and described.

The ARAMIS concept and the developed toolbox were evaluated in three case studies, the last two being conducted in industrial settings. The results were very positive. The evaluation proved that the ARAMIS approach can be utilized by organizations when attempting to understand and evaluate the current state of implemented architectures and as a starting point for future evolution.

Kurzfassung

In dieser Arbeit stellen wir ARAMIS vor: einen neuen, innovativen Ansatz und eine dazu gehörende Werkezugumgebung, um auf Basis von Laufzeitinformationen verhaltensbasiert die Konformität von Softwarearchitekturen zu überprüfen.

In den letzten Jahren wurden einige Ansätze für eine statisch-basierte Architekturkonformitätsüberprüfung von Softwaresystemen vorgeschlagen. Diese haben jedoch erhebliche Nachteile, insbesondere dann, wenn damit moderne, technologisch heterogene und verteilte Softwaresystemen analysiert werden sollen. Etablierte Technologien wie die Objektorientierung aber auch der Übergang von monolithischen zu komponentenbasierten Systemen haben die Komplexität von Softwaresystemen von deren Struktur zum Systemverhalten verschoben. Eine Prüfung der Architekturkonformität solcher Systeme kann häufig nicht mit statisch-basierten Verfahren durchgeführt werden, da diese nicht immer feststellen können, ob sich ein System so verhält, wie dies von der Architektur vorgesehen wurde. In dieser Arbeit stellen wir ARAMIS vor, einen neuen Ansatz zur Prüfung der Architekturkonformität, der dieses Problem löst.

In einem ersten Schritt wird dabei die vorgesehene Architekturbeschreibung des zu analysierenden Softwaresystems mittels eines ARAMIS-spezifischen Metamodells erfasst. Diese Beschreibung besteht im Wesentlichen aus den Architektureinheiten des Systems und ihren Kommunikationsregeln. Um die Akzeptanz des Ansatzes zu erhöhen, werden darüber hinaus Techniken des Modell-Engineerings genutzt. Dabei wird das Ziel verfolgt, schon existierende Architekturbeschreibungen verarbeiten zu können, die nicht dem ARAMIS-Metamodell entsprechen. Im Anschluss daran werden die Interaktionen innerhalb einer ausgeführten Software aufgezeichnet, wobei vorhandene Monitoring-Systeme verwendet werden. Im Gegensatz zu den statisch-basierten Ansätzen, ist eine vollständige Analyse eines nicht trivialen Softwaresystems jedoch unmöglich. Um dem entgegenzuwirken, werden eine Reihe von Indikatoren eingeführt, die Hinweise bezüglich der Angemessenheit des analysierten Verhaltens, im Verhältnis zu einer systemweiten Konformitätsüberprüfung, liefern. Die Interaktionen innerhalb des Systems werden im nächsten Schritt analysiert. Dabei wird die Kommunikation den definierten Architektureinheiten zugewiesen und anschließend gegen die definierten Kommunikationsregeln geprüft. Dieses Ergebnis liefert die implementierte Architektur des Softwaresystems, die insbesondere auch die Abweichungen von der vorgesehenen Architekturbeschreibung definiert. Die implementierte Architektur kann vielfältig analysiert werden, so können zum Beispiel benutzerdefinierte Architektursichten und Perspektiven definiert oder dedizierte Visualisierungen genutzt werden. Abschließend werden Prozesse vorgeschlagen, die einen Leitfaden für eine verhaltensbasierte Architekturkonformitätsüberprüfung darstellen.

Der ARAMIS-Ansatz wurde durch drei Fallstudien evaluiert, zwei davon im industriellen Kontext. Die Ergebnisse sind sehr positiv. Die Evaluierungen haben gezeigt, dass ARAMIS effektiv genutzt werden kann, um eine implementierte Architektur im Bezug zu ihrer vorgesehenen Architekturbeschreibung zu verstehen und zu evaluieren. Dies ist ein wichtiger Ausgangspunkt für eine gezielte Softwareevolution.

Contents