Jahresbericht 2020

Prof. Dr. Holger Giese
Fachgebiet Systemanalyse und Modellierung

Hasso-Plattner-Institut für
Digital Engineering gGmbH

Campus Griebnitzsee
Universität Potsdam

HPI Hasso Plattner Institut
Digital Engineering · Universität Potsdam

Jahresbericht / Annual Report 2020

Fachgebiet Systemanalyse und Modellierung
Hasso-Plattner-Institut für Digital Engineering
Universität Potsdam

Fachgebiet *Systemanalyse und Modellierung*
Hasso-Plattner-Institut für Digital Engineering gGmbH
Universität Potsdam
Prof.-Dr.-Helmert-Str. 2-3, D-14482 Potsdam
Leitung: Prof. Dr. Holger Giese

`http://www.hpi.de/giese`

Inhaltsverzeichnis / Table of Contents

1 Personelle Zusammensetzung / Staff

Leiter des Fachgebiets / Head
Prof. Dr. Holger Giese

Sekretariat / Secretary
Kerstin Miers

Senior Researcher
Dr. Leen Lambers

Postdocs
Dr. Maria Maximova
Dr. Sven Schneider

Wissenschaftliche Mitarbeiter / Research Assistants
Matthias Barkowsky, M.Sc.
Dipl.-Inform. Joachim Hänsel (03.2014 - 06.2020)
Lucas Sakizloglou, M.Sc.
Christian Zöllner, M.Sc.

PhD-Stipendiaten / Scholarship Holders
Christian Adriano, M.Sc.
Dipl.-Wirtsch.Inf. Thomas Brand
Sona Ghahremani, M.Sc.
Dipl.-Inform. Joachim Hänsel (01.08.2020 - 30.11.2020)
Iqra Zafar, M.S
He Xu, M.Sc.

Extern
Dr. Dominique Blouin
Dr. Soumyadip Bandyopadhyay

Studentische Hilfskräfte / Student Assistants

Maximilian Böther	Henrik Guhl	Jan-Eric Hellenberg
Leander Masopust	Felix Roth	Ciara Schirmer
Maximilian Schulze	Fabian Schwenker	Simon Wietheger

2 Lehrveranstaltungen / Courses

2.1 Vorlesungen / Lectures

Sommersemester / Summer term 2020

- Software Language Engineering
- Requirement Engineering

Wintersemester / Winter term 2020/2021

- Probabilistic Models: Modeling, Learning, and Analysis
- Software Analysieren, Testen und Verifizieren

2.2 Übungen/Projekte / Exercises/Projects

Sommersemester / Summer term 2020

- Software Language Engineering
- Requirement Engineering

Wintersemester / Winter term 2020/2021

- Probabilistic Models: Modeling, Learning, and Analysis
- Software Testen, Analysieren und Verifizieren

2.3 Seminare / Seminars

Sommersemester / Summer term 2020

- Graph Transformaton for Software Engineers
- Online Learning for Self-Healing and Self-Optimization
- Simulation
- Software using machine learning: how to test and verify?

Wintersemester / Winter term 2020/2021

- Graph Neural Networks
- Multi-Agent Systems
- Queries for Large-Scale Graphs

3 Betreuung von Studierenden und Dissertationen / Supervised Students and Dissertations

3.1 Betreuung von Bachelorprojekten / Supervised Bachelor projects

3.1.1 Human-in-the-loop-Simulation von MANV-Lagen als Übung für den Bevölkerungsschutz

Zeitraum / Project Period: ab 10/2020 bis 07/2021

Kooperationspartner / Project Partners:

- Malteser Hilfsdienst e.V. Berlin ↗website
- Akademie für Krisenmanagement, Notfallplanung und Zivilschutz (AKNZ) des Bundesamts für Bevölkerungsschutz und Kathastrophenhilfe (BBK) ↗website
- Berliner Feuerwehr- und Rettungsdienstakademie (BFRA) ↗website

Motivation Ein Massenanfall von Verletzten (MANV) ist ein Großschadensereignis, bei dem die personellen oder materiellen Ressourcen nicht zur adäquaten Behandlung von allen Verletzten ausreichen, sodass auf besondere Verfahren der Katastrophenmedizin zurückgegriffen werden muss. Aufgrund der seltenen Einsätze und dem daraus resultierenden Erfahrungsdefizit ist das regelmäßige Training solcher Situationen aber für das kompetente Handeln aller Einsatzkräfte essentiell. Speziell für ehrenamtliche Einsatzkräfte im Bevölkerungsschutz sind möglichst einfache und kostengünstige Trainingsmöglicheiten daher besonders wichtig.

Problem Das Training erfolgt üblicherweise durch das (sehr aufwändige und teure) Nachstellen von MANV-Lagen mit Verletztendarstellerinnen. Es gibt zudem einige Simulationssysteme für Trainingszwecke. Eines der bekanntesten ist die dynamische Patientensimulation (dPS) von der AKNZ, welche die Patientenversorgung bei MANV-Lagen mit laminierten Karten und Aufklebern simuliert. Allerdings sind auch Übungen mit der dPS weiterhin mit vergleichsweise hohen Materialkosten und Durchführungsaufwand sowie einer formatbedingte Übungskünstlichkeit verbunden.

Lösung Das Bachelorprojektteam entwickelt eine digitale Variante der dPS, mit der MANV-LAgen einfach und kostengünstig trainierbar werden sollen. Auch die Übungskünstlichkeit soll reduziert werden. Dabei wird eine Serverkomponente entwickelt sowie eine App, die auf den Endgeräten der Übungsteilnehmer laufen kann. Dank der Kooperation mit der AKNZ kann das Team dabei die originalen Datensätze der dPS nutzen, dank der beiden Projektpartner in Berlin steht für das Projekt ein großer Pool an potentiellen Testbenutzern zur Verfügung.

Ansprechpartner / Contact: Christian Zöllner, Matthias Barkowsky.

↗website

3.2 Betreuung von Dissertationen / Supervised PhD theses

3.2.1 Laufende Dissertationen / Running PhD theses

Christian Adriano: Causal models of Software Fault Understanding with an Application to Scalable Source Code Inspection Tasks for Software Bugs

Matthias Barkowsky: Modular and Incremental Global Model Management

Thomas Brand: Generic Adaptive Monitoring with Architectural Runtime Models

Sona Ghahremani: Architecture-centric Self-adaptation

Joachim Hänsel: Testing for Self-Adaptive Software Systems

Lucas Sakizloglou: Encoding, Querying, and Monitoring the History of Runtime Models

He Xu: Run-time Verification and Validation for Self-adaptive System

Iqra Zafar: Failure Propagation in Microservice Architecture

Christian Zöllner: Modeling and Simulation of Collaborations

4 Bearbeitete Forschungsthemen / Research Topics

4.1 Causal models of Software Fault Understanding with an Application to Scale Source Code Inspection Tasks for Software Bugs

Software programmers spend from 20% to 40% of their time searching for the causes of software failures. To alleviate that, debugging techniques were developed to reduce the search space from the entire program execution to a list of suspicious program statements. However, these debugging techniques assume "perfect fault understanding", i.e., that the programmer will always recognize the software fault among the list of suspicious program statements. Since inaccurate fault understanding inevitably happens, this causes programmers to waste time generating invalid bug fixes, which in turn undermine the trust on the debugging techniques.
I investigated fault understanding in the context of code inspection tasks that are focused on a few lines of code at a time. These tasks are typical of inspecting codes for bugs during debugging. However, these tasks are mostly invisible, as they take a few seconds or minutes, reasonably self-contained, and leave little to no traces in logs or versioning systems. Hence, the nature of these small tasks pose a challenge to investigate them.
My approach was to investigate ways to capture performance attributes of these tasks, while at the same time allowing to scale to hundreds of tasks. I designed an experimentation platform that allows for: (1) recruitment and qualification of programmers, (2) automatic generation of tasks from a set of template questions, and (3) incremental distribution of tasks based on the outcome of previous tasks.
I evaluated my approach through a series of experiments with real software failures from popular Open Source Software Projects. Our preliminary results are promising in a sense that (1) different groups of programmers (subcrowds) were able to correctly identify the cause of the software failures within a few lines of code, (2) the speed and cost were reduced by incrementally deciding which tasks to allocate and to whom, and (3) as part of the tasks, programmers provided explanations that contributed positively to suggest bug fixes.
These results opened a new research problem: how to select a minimal set of tasks that maximize bug finding precision?. This is a difficult problem because requires us to decide at any given moment which program statements to inspect by whom and how many programmers. The approach was to partition this problem in three sub-problems: a causal model, an aggregation model, and a task sequencing model. The causal model explains the accuracy of task outcomes based on the programmers' coding skill and the tasks attributes (duration, perceived difficulty). The aggregation model consists of mechanisms (majority and cardinal voting) that summarize the competing opinions about the bug location. The task sequencing model combines the Bayes update procedure (by learning from previous tasks) with the expected utility of each available task (extracted from the causal model). This way I incrementally updated the knowledge about which minimal set of new tasks, if executed, would maximize the chances of precisely locating a given software bug.

Ansprechpartner / Contact: Christian Adriano

4.2 Modular and Incremental Global Model Management

The development of complex software systems involves the creation and maintenance of a multitude of models describing various aspects such as the architecture, behavior and requirements of the software. In model-driven development, models are assigned an important role in the development process and are subject to both manual and automated activities. Since these models may cover overlapping parts of the system under development, the execution of such activities has to be coordinated properly by global model management in order to avoid inconsistencies in the system's description.
Because of the heterogeneity and growing size of the involved models, global model management poses several challenges. We want to address these issues by employing the concept of megamodels to document and execute the interplay between models. In particular, we are studying the concepts required to achieve a solution which can cope with incremental changes to existing models on the one hand and allows a modular introduction of additional models and activities on the other hand. To that end, we are working on extending the triple-graph-grammar approach for model synchronization by an efficient change propagation between models while keeping track of their version history in a compact manner. This is supported by our work on optimizing the execution of graph queries over large models via a combination of existing static and dynamic techniques for graph pattern matching and a decomposition into simpler subqueries.

Ansprechpartner / Contact: Matthias Barkowsky

4.3 Adaptive Monitoring with an Architecture Runtime Model

Information about the operation and usage of a system can be useful for different purposes, e.g., deciding about the adaption of the system configuration or the evolution of its underlying software to fit changing requirements. Monitoring to obtain this information can comprise the following tasks: gathering, aggregating, checking, transporting, storing, and accessing data. Over time those tasks consume significant amounts of resources, such as compute power, bandwidth, and storage. However, much data is produced and processed for nothing as it is not considered by downstream analysis routines. Either the current set of routines does not consider the monitored system properties at all (static analysis sufficient) or the relevance of properties depends on the values of other properties and the corresponding conditions are not fulfilled at present (requires dynamic analysis). Wasting resources for unnecessary monitoring is at least problematic from an environmental standpoint and in systems with constrained resources, e.g., when relying on battery driven monitoring sensors.
We support detecting phenomena in the current system state at runtime in an automated fashion. We presume that mechanisms exist which can be triggered to analyze the current values of monitored properties. A property value is either primitive like number or structural like a set of references. Thus, the searched phenomena can also be related to structural aspects of the system. The monitoring of individual properties can be activated and deactivated where the activation can consume a considerable amount of time.
We argue that it is possible to significantly reduce the amount of unnecessarily monitored properties by observing the interaction of the phenomena detection mechanisms with an interface for accessing property values. As such an interface we employ an architecture runtime model which represents relevant parts of the running system based on monitoring.

Our approach is agnostic to the phenomena detection mechanisms, e.g., simple Java programs, a query specification interpreter, or RETE nets. It is also agnostic regarding the purpose for which the monitoring and phenomena detection are performed as well as to the meta model of the runtime model. This makes the approach flexible to be used in different scenarios where the relevance of properties and their values depends on the values provided for other properties.

Ansprechpartner / Contact: Thomas Brand

4.4 Architecture-centric Self-adaptation

Architecture-based self-adaptive systems abstract the observed behavior of the running system into features of an architectural model, this makes it possible for the adaptation engine to reason about the changes that should be made to a system using variety of existing architectural analysis techniques. There are various ways how self-adaptation following the MAPE-K feedback loop and in particular the analyzing and planning phases of the loop can be realized. Rule-based approaches prescribe the adaptation to be executed if the system or environment satisfy certain conditions and result in scalable solutions, however, with often only satisfying adaptation decisions. In contrast, utility-driven approaches determine optimal adaptation decisions by using an often costly optimization step, which typically does not scale well for larger problems.
We propose a rule-based and utility-driven approach that achieves the beneficial properties of each of these directions such that the adaptation decisions are optimal while the computation remains scalable as an expensive optimization step can be avoided. The approach can be used for the architecture-based self-healing of large software systems. In our approach, we model the dynamic architecture of the self-adaptive system as a graph. Natural state of the system as well as the abstract syntax of the runtime models of the software are depicted via an annotated graph. We apply architectural utility functions in which any possible architectural configuration of the system is mapped to a scalar value.
We define the utility for large dynamic architectures of such systems based on patterns capturing issues the self-healing must address and we use pattern-based adaptation rules to resolve the issues. Defining the utility as well as the adaptation rules in a pattern-based manner allows us to compute the impact of each rule application on the overall utility and realize an incremental and efficient utility-driven self-adaptation. We target both self-healing and self-optimization in architectural manner.Achieving optimal adaptation decisions on-line within a reasonable time is an important challenge of self-adaptive software systems that is addressed.

Ansprechpartner / Contact: Sona Ghahremani

4.5 Testing for Self-Adaptive Software Systems

Self-adaptive software systems are equipped with feedback loops to adapt autonomously to changes of the software or environment. In established fields, such as embedded software, sophisticated approaches have been developed to systematically study feedback loops early during the development. In order to cover the particularities of feedback, techniques like one-way and in-the-loop simulation and testing have been included. However, related approaches for systematic testing of feeback loops in self-adaptive software system do not exist.

We propose a systematic testing approach based on architectural runtime models for self-adaptive software systems. The aim is to exploit architectural runtime models for testing early in the development phase, since they are usually available, even before the different activities of a feedback loop are realised or even designed. Furthermore we research testing of self-adaptive software systems at runtime in order to benefit from knowledge about the changed environment which is not available at design time.

Ansprechpartner / Contact: Joachim Hänsel

4.6 Encoding, Querying, and Monitoring the History of Runtime Models

A (structural) Runtime Model provides a snapshot of the state of a system at runtime. Runtime Models are typically employed in the context of Self-adaptive Systems (SAS), where a feedback loop adapts the system behavior at runtime in response to external or internal stimuli. Capturing the history of Runtime Models, i.e., snapshots of past states, may be useful for a number of aims relevant to SAS such as the detection of recurrent stimuli and loop reactions or postmortem analysis. However, the topic of history in Runtime Models has only recently received attention and, consequently, there is a lack of sophisticated technology. We present an approach (and the foundations thereof) that aims at efficiently handling history of Runtime Models. In detail, we define an encoding for Runtime Models which may allow for a memory-efficient history representation. Moreover, we define model queries that express requirements on the evolution of model fragments and present an incremental query execution scheme for such queries. Finally, we present a tool which enables monitoring history and, thus, allow for the utilization of history in self-adaptation scenarios.

Ansprechpartner / Contact: Lucas Sakizloglou

4.7 Run-time Verification and Validation for Self-adaptive System

The software is now the backbone of human activity. Software systems play important roles in industrial facilities, automobile, and aircraft etc.
In self-adaptive systems, the software has to deal with the rapidly changing environment conditions and the failures of its own system. How to guarantee the functional and non-functional requirements of the system during and after the adaptation process is a crucial problem.
Verification and Validation theory is wildly adopted in the whole cycle of software system development. Expanding these techniques into run-time verification and validation for self-adaptive systems is a great challenge. Run-time V&V can ensure, during or after the adaptation, system's requirements and its core qualities will not be compromised, and at the same time, the goals of adaptation process will be satisfied. Run-time V&V methods and tools are critical for the success of autonomous, autonomic, smart, self-adaptive and self-managing systems.
There are three parts in my research topic: First, to investigate the formal methods and their use at run-time, especially run-time model checking. Second, to implement the research on system modeling and requirements/properties specification methods. Third, to integrate the run-time verification and self-adaptive system and to find out a general structure for providing assurances for the self-adaptive system in its whole life cycle.

Ansprechpartner / Contact: He Xu

4.8 Failure Propagation in Microservice Architecture

In large-scale microservice architecture, reliability issues can be caused by a variety of reasons, such as restricted host resources, unavailable hardware or software, unstable networks, etc. To keep the system running reliably, cloud operators often deploy large-scale performance checkpoints and metric collection mechanisms. These monitoring mechanisms can help us initially verify that applications and services are running healthily. However, since the applications run on a complex architecture consisting of various microservices, it is a challenging task to reveal the mechanism of anomaly propagation from massive monitoring metrics, and to pinpoint the root cause of the failure. In conclusion, traditional anomaly diagnosis methods are usually based on key performance indicator (KPI) thresholds.System administrators set the KPI monitoring threshold manually according to their domain knowledge for early warning. However, due to the very large number of services in the MSA application and the complex dependencies between services, it is difficult for system administrators to detect anomalies by setting reasonable KPI monitoring thresholds, let alone diagnose root causes in fine granularity. So, Automated root cause analysis reduces the overall dependency on expert knowledge and provide solution to failure propagation in Microservice Architecture.

Ansprechpartner / Contact: Iqra Zafar

4.9 Modeling and Simulation of Collaborations

In future large-scale cyber-physical systems, the interconnection of autonomous systems via complex software and networking will result in massive systems of systems where a huge number of heterogenous systems collaborate and act together. In this research topic, we address the challenge of modeling relevant aspects of such systems of systems. Given the high demand for safety assurances for cyber-physical systems, the thorough analysis of systems and their models is obligatory. Besides verification and validation, we propose simulation as a means to identify and resolve potential safety risks and gain further insights into how the modeled systems act and collaborate in a large systems of systems context.

Ansprechpartner / Contact: Christian Zöllner

5 Drittmittelprojekte / Third-Party funded Projects

5.1 DFG – Quantitative Analyse von service-orientierten Echtzeitsystemen mit Strukturdynamik (QUANTUM)

Gefördert / Funded: ab 01/2015
Drittmittelgeber / Funding organisation: DFG

Ziel von QUANTUM ist die Entwicklung neuer quantitativer Modelle und quantitativer Analysetechniken für service-orientierte Echtzeitsysteme, welche die nötigen Kombinationen aus probabilistischem Verhalten, Echtzeitverhalten und Strukturdynamik bieten, die besondere Relevanz im Bereich von service-orientierten Echtzeitsystemen haben. Obwohl bereits limitierte Kombinationen aus probabilistischem Verhalten, Echtzeitverhalten und Strukturdynamik existieren, und auch substantielle Fortschritte bezüglich ihrer Analyse in den letzten Jahren gemacht wurden, fehlt noch immer eine komplette Kombination, welche alle geforderten Aspekte in einem Modell vereint.
Im Projekt ist deshalb geplant, die existierenden Modelle von zeitbehafteten Graphtransformationssystemen und probabilistischen Graphtransformationssystemen zu kombinieren und zu erweitern und passende Analysemöglichkeiten durch Integrieren von existierenden Werkzeugen bereitzustellen, welche die quantitative Analyse einer größeren Klasse von Systemen und ihrer Eigenschaften erlaubt als es durch die bisher existierenden Modelle möglich ist. Neben dem neuen formalen Modell, welches alle relevanten Aspekte abdeckt, wird eine probabilistische zeitbehaftete Spezifikationslogik, eine auf dem formalen Modell basierende, abstrakte QUANTUM-Modellierungssprache, welche durch Erweitern des SoaML UML-Profils direkt die Beschreibung von service-orientierten Echtzeitsystemen ermöglicht, sowie eine verwandte visuelle Spezifikationssprache für QUANTUM-Modelle entwickelt, um die Modellierungskonzepte und Analysetechniken für ein breiteres Publikum nutzbar zu machen.

Ansprechpartner / Contact: Holger Giese, Maria Maximova. ↗website

5.2 DFG – Korrekte Modelltransformationen (KorMoran III) – 2. Fortsetzungsprojekt

Gefördert / Funded: ab 11/2017
Drittmittelgeber / Funding organisation: DFG

Eingebettete Systeme sind heutzutage allgegenwärtig. Durch immer größer werdende Rechenleistungen und Vernetzung der Systeme sind diese in der Lage, immer komplexere Aufgaben zu erfüllen. Um diese Komplexität beherrschen zu können ist es notwendig, standardisierte und bewährte Methoden der Softwareentwicklung anzuwenden. Dazu zählt die modellgetriebene Entwicklung (MDE), die den Entwickler vom Design der abstrakten Anwendungsfälle bis zum konkreten, ausführbaren Code begleitet. Auch die abstrakten Modelle können sehr komplex werden. Eine Technik zur Reduzierung der Komplexität ist das Refactoring – Modelltransformationen, die äquivalentes Verhalten bei Ausführung der Modelle garantieren. In besonders sicherheitskriti-

schen Bereichen, beispielsweise in der Automobil-, Luftfahrt- und Schienenverkehrsindustrie, spielt darüber hinaus formale Verifikation eine große Rolle. In diesen Industriezweigen wird zum MDE überwiegend MATLAB/Simulink eingesetzt.
Das DFG-Projekt KorMoran widmet sich daher dem Problem der Verifikation von Modelltransformationen, konkret dem formalen Beweis der Verhaltensäquivalenz von Quell- und Zielmodell. In den Vorgängerprojekten KorMoran I und II wurden sowohl Transformationen für zeit-diskrete und wert-diskrete Transitionsmodelle als auch für zeit-diskrete, zeit-kontinuierliche und wert-kontinuierliche Datenflussmodelle untersucht. In KorMoran III sollen nun zunächst die Untersuchungen zu zeit-diskreten und zeit-kontinuierlichen Datenflussmodellen fortgesetzt werden. Insbesondere ist eine Erweiterung der Verifikationsmethoden geplant, um hybride Systeme zu unterstützen – Modelle, in denen sowohl zeit-diskrete als auch zeit-kontinuierliche Anteile gemischt vorkommen.
Bezüglich der Verifikation von Transformationen für Transitionsmodelle sollen Erweiterungen bestehender Techniken bezüglich der Ausdrucksmächtigkeit und Anwendbarkeit entwickelt werden. Konkret sollen die existierenden Verifikationstechniken erweitert werden, um neben Bisimulation und Simulation auch schwache Bisimulation und Simulation zu unterstützen. Zusätzlich sollen auch Methoden für Transformationen der Transitionsmodelle mit wert-kontinuierlichen Signalen entwickelt werden.
KorMoran III ist ein Kooperationsprojekt zwischen der Technischen Universität Berlin und dem Hasso-Plattner-Institut, wobei erstere den Fokus auf den Bereich der Datenflussmodelle legt, während Modelltransformationen für Transitionsmodelle am Hasso-Plattner-Institut betrachtet werden. Gemeinsamer Teil des Projekts wird die Kombination entwickelter Techniken sein, um Anwendbarkeit der in Datenflussmodelle eingebetteten Transitionsmodelle zu untersuchen und zu erreichen.

Ansprechpartner / Contact: Holger Giese, Sven Schneider, Leen Lambers, Soumyadip Bandyopadhyay. ↗ website

5.3 DFG – Modulares und inkrementelles globales Modell-Management (miGMM)

Gefördert / Funded: ab 07/2018
Drittmittelgeber / Funding organisation: DFG

Die Entwicklung komplexer Systeme mit Hilfe einer Vielzahl von Modellen benötigt ein globales Modell-Management, das sicherstellt, dass neben den Arbeiten auf einzelnen Modellen auch das Wechselspiel zwischen den Modellen geeignet verwaltet wird. Solch eine Verwaltung muss dabei die Integration der Modellierungssprachen, die Koordination der Aktivitäten auf Basis der Modelle sowie die Verwaltung der Modelle und all der Aktivitäten auf diesen abdecken. Es existiert zwar eine Reihe von Ansätzen, die Teile dieses Problem zu adressieren versuchen; ein fundiertes Verständnis der Bedürfnisse und Herausforderungen fehlt jedoch bisher. Darüber hinaus skalieren die meisten Lösungen nicht für die heutzutage durchaus vorkommenden sehr großen Modelle und sie unterstützen auch keine Modularität. Diese Einschränkung gilt sowohl für die Konstruktion als auch Ausführung der Modelle und der entsprechenden Aktivitäten. Im Projekt "modulares und inkrementelles Globales Modell-Management" (miGMM) wollen wir deswegen die Herausforderung des globales Modell-Management angehen, indem wir einen Ansatz für Mega-Modelle mit Integra-

tionslinks, Integrationssichten, Nachverfolgbarkeitslinks, Modellkonsistenz und Modelloperationen entwickeln und dabei insbesondere die notwendigen Konzepte für eine inkrementelle und modulare Lösung erforschen.

Ansprechpartner / Contact: Holger Giese, Matthias Barkowsky.

6 Forschungskooperationen / Research Cooperations

6.1 Projektpartner aus der Wissenschaft / Project Partners from Research Institutions

COST Action IC1404 Multi-Paradigm Modelling for Cyber-Physical Systems (MPM4CPS)
Hans Vangheluwe, University of Antwerp (Belgium) and McGill University, Montréal (Canada)
Vasco Amaral, NOVA-LINCS FCT, Universidade Nova de Lisboa (Portugal)

DFG-Projekt KorMoran III
Jürgen Dingel (Queen's University)
Sabine Glesner (Technische Universität Berlin)

Scalable Model Management
Etienne Borde (Télécom ParisTech, Université Paris-Saclay)
Dalila Tamzalit (LS2N, Université de Nantes)

Modeling and Analysis of Ecological Systems
Cédric Gaucherel (INRA — AMAP Laboratory, Montpellier)
Christelle Hély (ISEM, Université de Montpellier, CNRS, IRD, EPHE, Montpellier)
Boris Flotterer (ISEM, Université de Montpellier, CNRS, IRD, EPHE, Montpellier)

6.2 Externe Kooperationspartner bei Publikationen / External Partners in Publications

Prof. Dr. Holger Giese hat in 2020 mit folgenden externen Kooperationspartnern gemeinsame Publikationen veröffentlicht: Thomas Vogel.

Dr. Leen Lambers hat in 2020 mit folgenden externen Kooperationspartnern gemeinsame Publikationen veröffentlicht: Fernando Orejas, Jens Weber.

Sona Ghahremani hat in 2020 mit folgenden externen Kooperationspartnern gemeinsame Publikationen veröffentlicht: Thomas Vogel, Mirko D'Angelo, Simos Gerasimou, Johannes Grohmann, Ingrid Nunes, Evangelos Pournaras, Sven Tomforde.

7 Publikationen / Publications

7.1 Zeitschriftenartikel / Journal Articles

[A1] Matthias Barkowsky and Holger Giese. Hybrid search plan generation for generalized graph pattern matching. *Journal of Logical and Algebraic Methods in Programming*, 114:100563, 2020.

[A2] Sona Ghahremani and Holger Giese. Evaluation of Self-Healing Systems: An Analysis of the State-of-the-Art and Required Improvements. *Computers*, 9(1), 2020.

[A3] Sona Ghahremani, Holger Giese, and Thomas Vogel. Improving Scalability and Reward of Utility-Driven Self-Healing for Large Dynamic Architectures. *ACM Trans. Auton. Adapt. Syst.*, 14(3), February 2020.

7.2 Begutachtete Konferenz- und Workshopartikel / Peer-Reviewed Conference and Workshop Papers

[K1] Mirko D'Angelo, Sona Ghahremani, Simos Gerasimou, Johannes Grohmann, Ingrid Nunes, Sven Tomforde, and Evangelos Pournaras. Learning to Learn in Collective Adaptive Systems: Mining Design Patterns for Data-driven Reasoning. In *2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, pages 121–126, August 2020.

[K2] Holger Giese, Leen Lambers, and Christian Zöllner. From Classic to Agile: Experiences from More than a Decade of Project-Based Modeling Education. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, MODELS '20, New York, NY, USA, 2020. Association for Computing Machinery.

[K3] Joachim Hänsel, Christian M. Adriano, Johannes Dyck, and Holger Giese. Collective risk minimization via a bayesian model for statistical software testing. In Shinichi Honiden, Elisabetta Di Nitto, and Radu Calinescu, editors, *SEAMS '20: IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, Seoul, Republic of Korea 29 June - 3 July, 2020*, pages 45–56. ACM, 2020.

[K4] Leen Lambers. How to Teach Software Testing? Experiences with A Sandwich Approach. In *13th IEEE International Conference on Software Testing, Verification and Validation Workshops, ICSTW 2020, Porto, Portugal, October 24-28, 2020*, pages 425–428. IEEE Digital Library, 2020.

[K5] Leen Lambers, Sven Schneider, and Marcel Weisgut. Model-Based Testing of Read Only Graph Queries. In *13th IEEE International Conference on Software Testing, Verification and Validation Workshops, ICSTW 2020, Porto, Portugal, October 24-28, 2020*, pages 24–34. IEEE, 2020.

[K6] Lucas Sakizloglou, Sona Ghahremani, Matthias Barkowsky, and Holger Giese. A scalable querying scheme for memory-efficient runtime models with history. In Eugene Syriani, Houari A. Sahraoui, Juan de Lara, and Silvia Abrah ao, editors, *MoDELS '20: ACM/IEEE 23rd International Conference on Model Driven Engineering Languages and Systems, Virtual Event, Canada, 18-23 October 2020*, pages 175–186. ACM, 2020.

[K7] Lucas Sakizloglou, Sona Ghahremani, Thomas Brand, Matthias Barkowsky, and Holger Giese. Towards Highly Scalable Runtime Models with History. In *15th IEEE/ACM International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS@ICSE 2020, Seoul South Korea, October, 2020*. IEEE Computer Society, 2020.

[K8] Sven Schneider, Johannes Dyck, and Holger Giese. Formal Verification of Invariants for Attributed Graph Transformation Systems Based on Nested Attributed Graph Conditions. In Fabio Gadducci and Timo Kehrer, editors, *Graph Transformation - 13th International Conference, ICGT 2020 Held as Part of STAF 2020, Bergen, Norway, June 25-26, 2020, Proceedings*, volume 12150 of *Lecture Notes in Computer Science*, pages 257–275. Springer, 2020.

[K9] Sven Schneider, Lucas Sakizloglou, Maria Maximova, and Holger Giese. Optimistic and Pessimistic On-the-fly Analysis for Metric Temporal Graph Logic. In Fabio Gadducci and Timo Kehrer, editors, *Graph Transformation - 13th International Conference, ICGT 2020 Held as Part of STAF 2020, Bergen, Norway, June 25-26, 2020, Proceedings*, volume 12150 of *Lecture Notes in Computer Science*, pages 276–294. Springer, 2020.

[K10] Christian Zöllner, Matthias Barkowsky, Maria Maximova, Melanie Schneider, and Holger Giese. A Simulator for Probabilistic Timed Graph Transformation Systems with Complex Large-Scale Topologies. In Fabio Gadducci and Timo Kehrer, editors, *Graph Transformation - 13th International Conference, ICGT 2020 Held as Part of STAF 2020, Bergen, Norway, June 25-26, 2020, Proceedings*, volume 12150 of *Lecture Notes in Computer Science*, pages 325–334. Springer, 2020.

8 Vorträge / Talks

8.1 Vorträge auf Konferenzen und Workshops / Talks at Conferences and Workshops

Dr. Leen Lambers

October 2020 *Model-Based Testing of Read-Only Graph Queries*. 16th Workshop on Advances in Model-Based Testing, online event, university of Porto, Portugal, October 24, 2020.

October 2020 *How to Teach Software Testing: Experiences with a Sandwich Approach*. 1st International Software Testing Education Workshop, online event, university of Porto, Portugal, October 28, 2020.

June 2020 *Initial Conflicts for Transformation Rules with Nested Application Conditions*. 13th International Conference on Graph Transformation, online event, June 25, 2020.

Dr. Sven Schneider

June 2020 *Optimistic and Pessimistic On-the-fly Analysis for Metric Temporal Graph Logic*. ICGT '20, Virtual, June 25, 2020.

June 2020 *Formal Verification of Invariants for Attributed Graph Transformation Systems Based on Nested Attributed Graph Conditions*. ICGT '20, Virtual, June 25, 2020.

Christian Adriano

June 2020 *Bayesian Causal Inference Models of Software Fault Understanding with an Application to Sequential Decision Models for Optimal Code Inspection Task Allocation*. DagStuhl Seminar, DFG Research Schools Meeting, Schloss Dagstuhl, Germany, June 04, 2020.

October 2020 *Causal and Sequential Decision Models of Software Fault Understanding*. Seminar, Research School Retreat, October 27, 2020.

Lucas Sakizloglou

July 2020 *Towards Highly Scalable Runtime Models with History*. SEAMS@ICSE '20, Virtual, July 02, 2020.

October 2020 *A Scalable Querying Scheme for Memory-efficient Runtime Models with History*. MoDELS '20, Virtual, October 23, 2020.

Christian Zöllner

June 2020 *A Simulator for Probabilistic Timed Graph Transformation Systems with Complex Large-Scale Topologies*. ICGT '20, Virtual, June 26, 2020.

October 2020 *From Classic to Agile: Experiences from More than a Decade of Project-Based Modeling Education*. EduSym@MoDELS '20, Virtual, October 20, 2020.

9 Web-Portale und -Services / Web-Portals and Services

9.1 Self-adaptive.org

Das Online-Angebot `http://www.self-adaptive.org` dient als Übersichtsseite für das jährliche Symposium *Software Engineering for Adaptive and Self-Managing Systems* (SEAMS) im Rahmen der *International Conference on Software Engineering* (ICSE). Auf der Webseite sind alle Call for Papers für aktuelle und vergangene SEAMS Symposien, eine umfassende themenspezifische Bibliographie, Informationen zu weiterführenden Veranstaltungen wie den Dagstuhl Seminaren 08031 und 10431 sowie eine Liste von Wissenschaftlern, die auf dem Gebiet forschen, zu finden.

9.2 MDELab.org

Mit dem Online-Angebot `http://www.mdelab.org` informieren wir über Forschungsarbeiten unseres Fachgebiets im Bereich des *Model-Driven Engineering* (MDE). Dabei liegt der Schwerpunkt auf Werkzeuge unter anderem für die modellgetriebene Softwareentwicklung, die an unserem Fachgebiet entwickelt werden und die zum Download bereitstehen.

9.3 CPSLab.org

Mit dem Online-Angebot `http://www.cpslab.org` informieren wir über Aktivitäten im Kontext unseres Labors im Bereich *Cyber-Physical-Systems*. Inhalte beziehen sich auf vergangene, aktuelle als auch geplante Forschungsarbeiten. Weiterhin werden ausgewählte Projekte, welche im Kontext der Lehre umgesetzt wurden, repräsentiert.

10 Mitgliedschaften, Programmkomitees und Gutachtertätigkeiten / Memberships, Committee and Reviewing Activities

10.1 Mitgliedschaften / Memberships

Prof. Dr. Holger Giese

- Mitglied der Association for Computing Machinery (ACM)
- Mitglied der folgenden Special Interest Groups: SIGSOFT, SIGBED, SIGPLAN
- Mitglied der IEEE (Valued IEEE Member, Member since 1994)
- Mitglied der IEEE Computer Society
- Mitglied der folgenden Technical Councils: TCSE, TCDP, TCRTS, TFAAS
- Mitglied der IEEE Systems, Man, and Cybernetics Society
- Mitglied der Gesellschaft für Informatik e.V. (GI)
- Mitglied der folgenden Fachgebiete und Fachgruppen: ST, TAV, OOSE, ASE, PN, SPECS, FOMSESS
- Mitglied des Deutschen Hochschulverbandes (DHV)

Christian Adriano

- Mitglied der Association for Computing Machinery (ACM)
- Mitglied der IEEE (IEEE Member, Member since 2008)
- Mitglied der IEEE Computer Society

Sona Ghahremani

- Mitglied der Association for Computing Machinery (ACM)

10.2 Mitarbeit in Programmkomitees / Activities in Program Committees

Prof. Dr. Holger Giese

- 1st International Conference on Autonomic Computing and Self-Organizing Systems - ACSOS
  Virtual, August 17-21, 2020, ↗ website
- 13th International Conference on Graph Transformation - ICGT)
  Virtual, June 25-26, 2020, ↗ website
- Modellierung
  Virtual, February 19-21, 2020, ↗ website
- 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems - SEAMS
  Virtual, June 29 -July 3, 2020, ↗ website

Dr. Leen Lambers

- 13th International Conference on Graph Transformation
  online event, June 25-26, 2020, ↗ website
- 11th International Workshop on Graph Computation Models
  Eindhoven, The Netherlands, June 24, 2020, ↗ website
- EduSymp@Models 2020
  online event, Montreal Canada, October 20, 2020, ↗ website
- Models20-Demo Tools and Demonstrations
  online event, Montreal Canada, October 21-23, 2020, ↗ website
- 17th Workshop on Model-Driven Engineering, Verification and Validation
  online event, Montreal Canada, October 20, 2020, ↗ website
- 14th Workshop on Models and Evolution
  online event, Montreal Canada, October 16, 2020, ↗ website

Sona Ghahremani

- Artifact Track at 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems - SEAMS.
  Virtual, June 29 -July 3, 2020, ↗ website
- Poster and Demo Track at 1th International Conference on Autonomic Computing and Self-Organizing Systems - ACSOS
  Virtual, August 17-21, 2020, ↗ website
- Workshop On Self-aware Computing at ACSOS - SEAC
  Virtual, August 17 2020, ↗ website

Christian Adriano

- Research Track at European Conference In Machine Learning
  Virtual, Sept. 14-18, 2020, ↗ website

10.3 Organisationstätigkeiten / Organizational Activities

Prof. Dr. Holger Giese

- Co-Chair for Tutorials at Modellierung
  Virtual, February 19-21, 2020, ↗ website

Sona Ghahremani

- Session Chair for Testing, Analysis, Reasoning, and Monitoring at SEAMS
  2020, Virtual, ↗ website

10.4 Gutachtertätigkeiten / Reviewing Activities

10.4.1 Forschungsprojekte / Research Projects

Prof. Dr. Holger Giese

- Deutsche Forschungsemeinschaft (DFG)
- Österreichische Forschungsförderungsgesellschaft (FFG)

10.4.2 Zeitschriften und Magazine / Journals

Prof. Dr. Holger Giese

- IEEE Transactions on Software Engineering (TSE)
- Springer Journal of Software and Systems Modeling (SoSyM)

Dr. Leen Lambers

- Information and Computation Journal, Elsevier
- Compositionality Journal

Sona Ghahremani

- Future Generation Computer Systems - FGCS 2020

Dr. Sven Schneider

- International Journal on Software Tools for Technology Transfer, Springer