Report from Dagstuhl Seminar 23302

Software Architecture and Machine Learning

Grace A. Lewis[*1], Henry Muccini[*2], Ipek Ozkaya[3], Karthik Vaidhyanathan[†4], Roland Weiss[*5], and Liming Zhu[*6]

1 **Carnegie Mellon Software Engineering Institute – Pittsburgh, US.** glewis@sei.cmu.edu
2 **University of L'Aquila, IT.** henry.muccini@univaq.it
3 **Carnegie Mellon Software Engineering Institute – Pittsburgh, US.** ozkaya@sei.cmu.edu
4 **IIIT Hyderabad, IN.** karthik.vaidhyanathan@iiit.ac.in
5 **ABB – Mannheim, DE.** roland.weiss@gmail.com
6 **Data61, CSIRO – Sydney, AU.** liming.zhu@data61.csiro.au

Abstract

This report documents the program and outcomes of Dagstuhl Seminar 23302, "Software Architecture and Machine Learning". We summarize the goals and format of the seminar, results from the breakout groups, key definitions relevant to machine learning-enabled systems that were discussed, and the research roadmap that emerged from the discussions during the seminar. The report also includes the abstracts of the talks presented at the seminar and summaries of open discussions.

Seminar July 23–28, 2023 – https://www.dagstuhl.de/23302
2012 ACM Subject Classification Software and its engineering → Software architectures; Computing methodologies → Machine learning; Software and its engineering → Extra-functional properties; Computing methodologies → Artificial intelligence; Software and its engineering
Keywords and phrases Architecting ML-enabled Systems, ML for Software Architecture, Software Architecture for ML, Machine Learning, Software Architecture, Software Engineering
Digital Object Identifier 10.4230/DagRep.13.7.166

1 Executive Summary

Grace A. Lewis (Carnegie Mellon Software Engineering Institute – Pittsburgh, US)
Henry Muccini (University of L'Aquila, IT)
Ipek Ozkaya (Carnegie Mellon Software Engineering Institute – Pittsburgh, US)
Karthik Vaidhyanathan (IIIT – Hyderabad, IN)
Roland Weiss (ABB – Mannheim, DE)
Liming Zhu (Data61, CSIRO – Sydney, AU)

The pervasive and distributed nature of many of today's software systems requires making complex design decisions to guarantee important system qualities such as performance, reliability, safety and security. The practices within the field of software architecture guide the design and development of software systems from its high-level blueprint down to their implementation and operations. While the fundamentals of software architecture practices

[*] Editor / Organizer
[†] Editorial Assistant / Collector

do not change, their execution evolves to address architecting with, and for, new system paradigms and emerging technologies. Incorporating machine learning (ML) elements into systems is advancing rapidly with the availability of more compute power, specialized infrastructure, and better and more efficient algorithms to process increasing amounts of data. This evolution has led many domains to leverage ML for automation, data analytics, decision support, and advanced user interfaces, among others. Experience published and shared by both software engineering research and practitioner communities shows that organizations struggle to move ML models and pilot projects into production. Reported software engineering challenges fall mostly outside data science expertise, which is focused on the development of ML models and not necessarily their production-readiness, and include areas such as testing, requirements management, software architecture, and configuration management. Among these, software architecture has a special role as it serves as an abstraction between requirements and implementation and drives the structure and behavior of systems.

Some of the reported challenges in developing ML-enabled systems, such as interface stability, data storage and access, and data transformation, are already addressed by existing software architecture techniques and practices. However, traditional software development and deployment practices are faced with and pose significant challenges when developing ML-enabled systems, which are systems that integrate ML components. These ML components include model training and updating components; model serving and inference components; infrastructure components to support data collection, processing, and servicing; and operations infrastructure, such as MLOps pipelines to support automated build and deployment. As these new ML-related components are introduced into systems, different architectural concerns and architecting challenges take higher priority, which include working with non-determinism, understanding and designing for new classes of dependencies, and co-architecting the system as well as the ML model development pipelines, among others. In addition, integration of ML components into software systems places greater emphasis on qualities that are not as common in non-ML systems, such as monitorability as a first-class citizen, designing for extensibility, and data-centricity.

This Dagstuhl Seminar culminated as a response to recognizing that software architecture research and practitioner communities have an opportunity, if not an obligation, to fill key software architecture principles and practices gaps that are particularly critical when incorporating ML components into software systems. The seminar focused on two key themes:

1. software architecture principles and practices for ML-enabled systems (SA4ML) and
2. application of ML techniques for improved architecting of software systems (ML4SA).

A key goal of the seminar was to enable technical exchange among otherwise scattered research and practitioner communities, such as software engineering, software architecture, self-adaptive systems, and machine learning around software architecture and ML (SA&ML). This Dagstuhl Seminar presented an opportunity to get these communities together to develop a common vocabulary and a coherent research agenda with a better understanding of problems faced in the industry. The goals of this seminar were to establish a common understanding of key concepts that are central to architecting ML-enabled systems, elicit challenges of meeting key quality attributes of ML-enabled systems, and build a research roadmap for future work to address these challenges. As such, this seminar marks an important milestone in accelerating research in SA4ML and ML4SA work by fostering better communication around key concepts and between diverse stakeholders. Hereafter, we will use the term SA&ML to include both SA4ML and ML4SA research directions.

Seminar Format

The seminar on software architecture and machine learning was structured to foster an interactive and productive idea exchange environment for all participants. Participants were initially asked to prepare a single-slide introduction about themselves and their work in software architecture and machine learning, which helped to establish a foundational understanding among attendees. Prior to the workshop, organizers developed an initial version of a key concepts map and a research challenges map using a grounded theory approach, which was represented visually on miro boards. During the seminar, participants delivered 10-minute lightning talks about their SA&ML work, which were instrumental in refining the research and concept maps with keywords and challenges mentioned by the speakers. At the end of the first day, organizers summarized the discussions and identified key emerging topics and research areas, leading to an initial set of key quality attributes critical to ML-enabled system development. The second day continued with discussions and a working group formation session, where participants prioritized five critical quality attributes for ML-enabled systems: Evolvability, Uncertainty and Observability, Trust and Trustworthiness, and Data Centricity. Four working groups were formed around these quality attributes, with appointed editors documenting the discussions. Each group was provided with a draft template to guide their discussions for consistency, and organizers rotated among groups to offer diverse insights. Plenary meetings on the second and third days allowed groups to share progress and discuss challenges, complemented by end-of-day meetings among organizers to consolidate learning and monitor progress. The seminar concluded on the fourth day with presentations from each group, open for feedback from other participants. This collaborative review process led to the integration of feedback into the final discussions, resulting in a well-defined set of challenges and research directions, marking the seminar's successful completion. In addition, this format ensured a thorough exploration of the seminar's themes and fostered active participation and collaborative learning among attendees.

2 Table of Contents

Executive Summary

Grace A. Lewis, Henry Muccini, Ipek Ozkaya, Karthik Vaidhyanathan, Roland Weiss, and Liming Zhu . . . 166

Overview of Talks

Fifty Shades of Uncertainty
Nelly Bencomo . . . 171

Software Architecture Modeling of Machine Learning Systems: Unsolved Challenge or Old Wine in New Bottles?
Justus Bogner . . . 172

AI: From Offline & Centralized to Online & Federated
Jan Bosch and Helena Holmström Olsson . . . 172

Predicting Software Performance with Divide-and-Learn
Tao Chen . . . 173

Analyzing Greenability of Software Architectures for AI Systems: The GAISSA project
Xavier Franch . . . 173

Why Organizations Fail to Implement AI
Benjamin Klöpper . . . 174

SPIRA Challenges and Lessons Learned on Architecting an Intelligent System for Respiratory Insufficiency Detection – Notes on Hands-on Education on SA4AI
Fabio Kon . . . 175

Software Architecture for Machine Learning Systems: Challenges, Practices, and Opportunities
Grace A. Lewis . . . 175

BeT (Behavior-enabled IoT)
Henry Muccini . . . 176

Software Architecting in the Era of AI and AI-Augmented Development Tools
Ipek Ozkaya . . . 176

Architecting Systems to Integrate Machine Learning
Lena Pons . . . 177

Machine Learning and Self-adaptation
Bradley Schmerl . . . 177

A Vision and Challenges about Intelligent and Trustworthy IoT Systems
Romina Spalazzese . . . 177

SA, ML and Patterns
Anastas Stoyanovsky . . . 178

Software Architecture Meets Machine Learning: A Tale of Convergence
Karthik Vaidhyanathan . . . 178

Generative AI at Fraunhofer and Research Roadmaps from the Software Architecture Community
Ingo Weber . . . 179

Building, Engineering & Operating Systems for Critical Infrastructure
Roland Weiss . . . 179

Challenges of Integrating ML Models in Safety-Relevant Architectures
Marc Zeller . . . 180

Software Architecture for Foundation Model-Based Systems
Liming Zhu . . . 180

Working groups

WG1: Architecting for Data Centricity
Jan Bosch, Benjamin Klöpper, Ipek Ozkaya, Lena Pons, and Christoph Schröer . . 181

WG2: Evolvability
Justus Bogner, Jan Bosch, Helena Holmström Olsson, Henry Muccini, Raghu Reddy, Anastas Stoyanovsky, Ingo Weber, and Liming Zhu . . . 182

WG3: Observability and Uncertainty
Nelly Bencomo, Xavier Franch, Fabio Kon, Ipek Ozkaya, Marie Platenius-Mohr, Bradley Schmerl, Roland Weiss, and Karthik Vaidhyanathan . . . 182

WG4: Architecting for Trust and Trustworthiness
Tao Chen, Thomas Kropf, Grace A. Lewis, Henry Muccini, Alex Serban, Romina Spalazzese, and Marc Zeller . . . 183

Open problems

Data Centricity . . . 184

Uncertainty and Observability . . . 185

Evolvability (and Adaptability) . . . 185

Trust and Trustworthiness . . . 186

High Priority Research Areas to Advance SA&ML . . . 186

Follow-up Work . . . 187

Participants . . . 188

3 Overview of Talks

The talks presented during the seminar included two industry keynotes. *Alex Serban from Siemens Healthineers* talked about *Software Engineering for Machine Learning: Past, Present and Future Conjectures*. The keynote explored the intricacies of machine learning, with a focus on the critical need for emphasizing robustness and trustworthiness in ML systems. It highlighted the challenges associated with integrating machine learning components into broader software architectures and underscored the importance of addressing uncertainty within these systems. The presentation delved into the development of engineering practices tailored to machine learning, emphasizing the need for adopting these practices to enhance agility, software quality, and team effectiveness. The keynote concluded by reflecting on the current and future trends in machine learning, particularly the role of language as a universal interface and the growing complexity of models, stressing the responsibility of software architects in crafting trustworthy and robust systems amidst emerging regulatory challenges.

Thomas Kropf from Robert Bosch in his keynote titled *Industrial AI – Real-World Applications, Challenges and Solution Approaches*, focused on the application and challenges of Industrial AI at Bosch, highlighting the journey from the establishment of the Bosch Center for AI to integrating AI across various products. It emphasized the distinct nature of Industrial AI, where quality, scaling, and algorithmic robustness are crucial, particularly in safety-critical applications, illustrated by examples such as Manufacturing Analytics and Automated Optical Inspection (AOI) systems. The keynote also covered significant challenges such as enterprise-level scalability, quality assurance amidst model drift, rapid AI evolution, computational constraints in embedded systems, and the necessity for extensive tool support for data management. The potential disruptive impact of Foundational Models on AI practices was also acknowledged. The keynote further described how these challenges influence software architectural choices, differentiating between cloud and low-power embedded AI solutions, and provided insights into Bosch's concrete strategies and architectures to address these issues, demonstrating the critical role of AI in industrial innovation and software architecture.

The examples of the state of the practice presented through these keynotes helped frame the remaining talks and discussions. What follows are the abstracts from the attendee talks.

3.1 Fifty Shades of Uncertainty

Nelly Bencomo (Durham University, GB)

There is growing uncertainty about the environment of software systems. Therefore, how the system should behave under different contexts cannot be fully predicted at design time. It is considerations such as these that have led to the development of self-adaptive systems (SAS), which can dynamically and autonomously reconfigure their behavior to respond to changing external conditions.

The scope of the talk is in the area of Requirements Engineering (RE) and the development of techniques to quantify uncertainty to improve decision-making. The explicit treatment of uncertainty by the running system improves its judgment to make decisions supported by

evaluating evidence found during runtime, possibly including the human-in-the-loop. I will also discuss how quantification of uncertainty can be used to improve requirements elicitation (using simulations, for example).

The talk will cover different approaches to quantifying uncertainty and its role in Human-Machine Teaming.

3.2 Software Architecture Modeling of Machine Learning Systems: Unsolved Challenge or Old Wine in New Bottles?

Justus Bogner (Universität Stuttgart, DE)

More and more software systems incorporate techniques from machine learning (ML) to support decision-making or automate information processing. Such ML systems still incorporate many traditional components but nonetheless require specialized practices in certain areas. From a software architecture perspective, one question is if existing practices and frameworks are suitable for the effective modeling and documentation of ML systems. While there are many methods for architecture documentation based on concepts like viewpoints, views, and modeling notations, it is still unclear how ML professionals can best apply these practices to model the architecture of ML systems. The data dependency and uncertainty of ML components, ML stakeholder diversity, plus new quality concerns like explainability or model observability might require new types of views and diagrams. While a few specialized approaches have been proposed, other software architecture publications use vastly different notations and diagrams to model ML systems.

In this short talk, I want to lay the foundation for a discussion about this topic. I will briefly talk about challenges in this space and present a few existing approaches. My goal is to discuss if these challenges are different from architecture modeling for traditional systems and if new approaches are needed. Ideally, the discussion might lead to research collaborations in this space to a) better understand challenges and b) create or adapt methods to overcome them.

3.3 AI: From Offline & Centralized to Online & Federated

Jan Bosch (Chalmers University of Technology – Göteborg, SE) and Helena Holmström Olsson (Malmö University, SE)

Digitalization is concerned with software, data, and AI as enabling technologies. It changes the company and business ecosystem in which it operates from transactional to continuous, i.e., XOps. This requires fundamental changes in how we work with technology from offline and centralized to online and federated. The talk includes examples from Software Center (www.software-center.se) companies, including automated experimentation, federated learning, reinforcement learning, semi-supervised learning, and related topics.

3.4 Predicting Software Performance with Divide-and-Learn

Tao Chen (University of Birmingham, GB)

Joint work of Tao Chen, Jingzhi Gong
Main reference Jingzhi Gong, Tao Chen: "Predicting Software Performance with Divide-and-Learn", in Proc. of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2023, San Francisco, CA, USA, December 3-9, 2023, pp. 858–870, ACM, 2023.
URL https://doi.org//10.1145/3611643.3616334

Predicting the performance of highly configurable software systems is the foundation for performance testing and quality assurance. To that end, recent work has been relying on machine/deep learning to model software performance. However, a crucial yet unaddressed challenge is how to cater for the sparsity inherited from the configuration landscape: the influence of configuration options (features) and the distribution of data samples are highly sparse. This talk is based on the paper in which we propose an approach based on the concept of "divide-and-learn", dubbed DaL. The basic idea is that to handle sample sparsity, we divide the samples from the configuration landscape into distant divisions, for each of which we build a regularized Deep Neural Network as the local model to deal with the feature sparsity. A newly given configuration would then be assigned to the right model of division for the final prediction.

Experiment results show that using strong domain knowledge to specialize AI performs significantly better in configuration performance learning.

3.5 Analyzing Greenability of Software Architectures for AI Systems: The GAISSA project

Xavier Franch (UPC Barcelona Tech, ES)

This presentation introduces the Spanish GAISSA project ("Towards Green AI-based Software Systems: An Architecture-centric Approach") run by the GESSI research group at UPC. The main hypothesis of GAISSA is that the impact of architectural decisions on environmental sustainability shall be understood, defined, reported, and managed in order to model, develop, and deploy green(er) AI-based systems. After presenting the objectives and vision, the presentation details the work done so far, which is basically a series of empirical studies on the effect of model and system architectures on environmental sustainability, exploring trade-offs with other attributes such as accuracy. Emerging challenges are the consolidation of the results of such studies, and how to boost the attention on sustainability in the community. In this respect, a result of the project is the so-called Energy Label, which categorizes the efficiency level (from A to E) of a trained model, defined in terms of several attributes. The concept has been proven using the HuggingFace repository. As future work, the talk mentions the link between requirements and architectures.

3.6 Why Organizations Fail to Implement AI

Benjamin Klöpper (Capgemini – Stuttgart, DE)

Many organizations, especially enterprises, use a data-lake-based infrastructure to prepare their data for machine learning modeling. Data-lake-based systems ingest raw data from a wide variety of data sources such as databases, legacy information systems, automation systems, and more. The processed data is intended to serve the needs of different users and use cases. A data-lake-based system consists of a complex technology stack that implements distributed data processing and data storage. Typical transformation steps include ingestion, cleansing, enrichment, transformation, and finally, serving.

Typically, a centralized team of data experts runs the infrastructure and implements the data processing. The thinking is centered around different data pipelines.

The monolithic architecture of the data lake and the centralized organization result in a siloed way of working that puts the data team in a difficult position, often leading to unsatisfactory machine learning results and limiting the data culture of the data team. The data team suffers from

- Operational data that is not fit for analytics, and the team lacks a deep understanding of the data
- Frequent and unexpected downstream changes that require the team to patch data pipelines in response
- Time spent searching for and identifying the right data for use cases
- Lack of understanding of business requirements

The resulting monolithic architecture and centralized team setup ignore common software engineering best practices, such as reasonably sized teams of 5-8 people with clear ownership, lack of mechanisms to offload cognitive load, minimizing hand-offs through loose coupling, and clear interfaces between components and teams.

One architectural approach that tries to facilitate the software engineering best practices are data products introduced by Dehgani. They ingest data only from a few source systems or other data products, encapsulate the data transformation and data storage, provide an easy-to-use business-oriented API, and contain a human-readable description and code for observability. Data products group elements of data transformation so that they can be owned by a software team and consumed as a service by other teams.

Essential to the adoption of a data product is a data platform that makes it easy for data product developers to follow the governance policies of the surrounding organization. A key goal of the data platform is developer experience (DevEx). There are several points to learn from a data-product-centric architecture:

- Design software and organizational architecture for ML-enabled systems together.
- Defined components or products should be independently deployable and a team of 5-8 developers should be sufficient to develop them.
- There needs to be some kind of platform to reduce the cognitive load.
- The platform should be built in an evolutionary fashion starting from the thinnest viable platform (TVP).

3.7 SPIRA Challenges and Lessons Learned on Architecting an Intelligent System for Respiratory Insufficiency Detection – Notes on Hands-on Education on SA4AI

Fabio Kon (University of Sao Paulo, BR)

Our group at the University of São Paulo has 23 years of experience teaching an advanced software development course called "Agile Software Development Lab." It is extremely popular with students and has been very successful in teaching advanced software development concepts with a 'learn by doing', project-based approach with real external customers. However, many recent projects have dealt with machine learning and AI-based systems, which sometimes significantly increase the complexity of projects. We are then currently remodeling the course to incorporate pedagogical aspects related to AI-based systems development.

In this short talk, I provide an experience report about the development of a complex system for respiratory insufficiency detection that has been developed at the University of São Paulo in a collaboration between data scientists, physicians, linguists, and computer science professors and students. We discuss the challenges and lessons learned in architecting and building the system.

3.8 Software Architecture for Machine Learning Systems: Challenges, Practices, and Opportunities

Grace A. Lewis (Carnegie Mellon Software Engineering Institute – Pittsburgh, US)

Developing software systems that contain machine learning (ML) components requires an end-to-end perspective that considers the unique life cycle of these components – from data acquisition to model training to model deployment and evolution. While there is an understanding that ML components, in the end, are software components, there are some characteristics of ML components that bring challenges to software architecture and design activities, such as data-dependent behavior, the need to detect and respond to drift over time, and timely capture of logs, metrics, user input and labeled data to inform retraining. In this presentation, I talk about some of these challenges and propose a set of practices to address these challenges, such as co-architecting, the recognition of monitorability as a driving quality attribute, and system-level architecture patterns and tactics. The presentation concludes with a list of opportunities to make progress in software architecture for ML systems, which includes adapting and growing the software architecture body of knowledge for application to ML systems, bringing the software architecture body of knowledge and discipline to model development activities, and positioning software architecture as a unifying activity for system stakeholders (model developers, software engineers, and operations).

3.9 BeT (Behavior-enabled IoT)

Henry Muccini (University of L'Aquila, IT)

Main reference Henry Muccini, Barbara Russo, Eugenio Zimeo: "The BET project: Behavior-enabled IoT", CoRR, Vol. abs/2307.13186, 2023.
URL https://doi.org//10.48550/ARXIV.2307.13186

BeT (Behavior-enabled IoT)[1] is a project that aims to provide a reference architecture, conceptual framework, and related techniques to design behavior-enabled IoT systems and applications. The presentation introduces the concept of the Internet of Behaviors (IoB) and analyzes the human-system bi-causal quality connection effects. BeT analyzes the emerging problem of understanding the mutual influence between QoS (Quality of Service) and QoE (Quality of Experience) in IoB systems, where dynamic changes in the system and surroundings can give rise to nontrivial unforeseen cause-effect mutual relations between system and human behaviors.

3.10 Software Architecting in the Era of AI and AI-Augmented Development Tools

Ipek Ozkaya (Carnegie Mellon Software Engineering Institute – Pittsburgh, US)

The boundary between SA4ML and ML4SA is diffusing as more general-purpose AI models are incorporated into systems. As more capable development tools enter the developer's ecosystem the role and responsibilities of the architect will shift [1]. As generative AI and foundation models increase in capabilities, they will help reveal the criticality of architecture knowledge and how it must guide development. As capabilities of large language models (LLM) evolve, however, there is a potential increasing gap between what tools supported by LLM can accomplish realizing narrowly scoped implementation tasks versus how they can do so while being cognizant of software architectural concerns [2]. This talk will emphasize some of the open questions and challenges that software architecture researchers need to address as LLM-supported development tools evolve rapidly. These include, but are not limited to, the following:

- What are the specific design and architectural concerns that need to be addressed when using generative AI?
- Can generative AI tools be used to improve the design and architecture of systems?
- Can these tools provide new features and capabilities that can be used to support the architectural design process?
- Could generative AI tools be used to generate design patterns and tactics, which could then be used to guide the development process?
- Can these tools accelerate the generation of alternative designs and their comparison?
- Can they be used to provide feedback on designs, such as identifying potential risks and issues?

References

1 Ipek Ozkaya: Application of Large Language Models to Software Engineering Tasks: Opportunities, Risks, and Implications. IEEE Softw. 40(3): 4-8 (2023)
2 Ipek Ozkaya. Can Architecture Knowledge Guide Software Development With Generative AI? IEEE Softw. 40(5): 4-8 (2023)

3.11 Architecting Systems to Integrate Machine Learning

Lena Pons (Carnegie Mellon Software Engineering Institute – Pittsburgh, US)

ML-enabled systems introduce additional concerns to systems, which introduce new complexity into the software architecture of such systems. Design considerations for integrating an ML component into a larger software system may be unfamiliar to software architects. Software engineers and data scientists need to communicate information that crosses domains, which may result in some considerations not being exposed during architecture and design conversations. We present a set of driving quality attributes for machine learning systems and how they align with conventional software engineering practice versus attributes where the integration of an ML component requires new patterns and tactics. We present some questions to assist architects in eliciting better information about design concerns.

3.12 Machine Learning and Self-adaptation

Bradley Schmerl (Carnegie Mellon University – Pittsburgh, US)

Architecture-based self-adaptation is a method for adding a control loop on top of systems to make changes and repairs at run time, with the software architecture as the basis for making decisions. But does this work for systems that have machine learning components in the architecture? We report on work that we are doing to investigate how to apply self-adaptation concepts to manage machine learning components. We report on the set of challenges related to the timeliness of observations, the propagation of uncertainty, and the lack of understood costs, benefits, and impacts of adaptation tactics, but also that the current concepts seem to apply. One of the key concepts in ML is explainability. We report on two aspects of explainability: 1) how we use planning models to support contrastive explanations, and 2) how standard techniques in ML, like principal component analysis, clustering, and decision tree learning, can be used to help explain choices that need to be made by architects, designers, etc. by focusing them on critical qualities/concerns and interactions among them.

3.13 A Vision and Challenges about Intelligent and Trustworthy IoT Systems

Romina Spalazzese (Malmö University, SE)

Today's systems are more and more software/hardware intensive, very large, heterogeneous, data-driven, dynamic, evolving, intelligent, and involve humans. These characteristics make their engineering and architecture challenging tasks. Additionally, different applications have different desired quality characteristics concerning, e.g., response time, power consumption, interoperability, privacy, and trust, which influence their design. Examples of such modern complex systems are ML-enabled Internet of Things (IoT) systems that have to deal with many challenges at the same time, both from a human and technical perspective.

An overarching question in our ongoing research project is "How should Intelligent and Trustworthy IoT systems be designed?" and some of the more specific questions are "How and when should AI (in particular ML) be used to realize such systems?" as well as "How could and when should edge computing (including hybrid edge-cloud processing) be used to realize such systems?". Towards identifying architectural approaches, patterns, and styles, as well as analysis models, metrics, and techniques for ML-enabled systems, I will discuss a vision and identify challenges related to an ML-enabled collaboration paradigm called IoT-Together.

3.14 SA, ML and Patterns

Anastas Stoyanovsky (Amazon – Pittsburgh, US)

The introduction of machine learning has not changed anything fundamental about architectural thinking or methods. What is new is the confluence of particular quality attributes, which effectively reduces the discussion to a matter of practical execution. Several high-level patterns, both architectural and organizational, are discussed as foundations for further inquiry. Industry experience points to challenges common to both industry and academia, leading to a call to action around education.

3.15 Software Architecture Meets Machine Learning: A Tale of Convergence

Karthik Vaidhyanathan (IIIT – Hyderabad, IN)

Main reference Henry Muccini, Karthik Vaidhyanathan: "Software Architecture for ML-based Systems: What Exists and What Lies Ahead", in Proc. of the 1st IEEE/ACM Workshop on AI Engineering – Software Engineering for AI, WAIN@ICSE 2021, Madrid, Spain, May 30-31, 2021, pp. 121–128, IEEE, 2021.
URL https://doi.org//10.1109/WAIN52551.2021.00026

Over the years, software systems have become increasingly complex due to their ever-increasing pervasive nature, resulting in various architecting challenges to ensure better performance, reliability, etc. On the other hand, machine learning (ML) has advanced rapidly with advancements in infrastructure, data availability, etc. However, the growing adoption of ML has given rise to challenges associated with development practices, deployments, ensuring data quality, etc., in addition to the challenges of a traditional software system. These challenges have resulted in the convergence of SA and ML, resulting in two broad research areas: i) ML4SA and ii) SA4ML.

This talk presents an overview of this convergence. The presentation then elaborates on the two research lines and the challenges the community must address going forward. The first part (ML4SA) focuses on using ML techniques to enable software systems to autonomously adapt and improve their architecture at run time to guarantee a better quality of service at run time and then using generative AI for architectural knowledge management for design-time aid for architects. The second part (SA4ML) elaborates on the challenges in architecting ML-enabled systems and further provides some insights on our recent work related to an agile methodology for ML-enabled systems [1] and self-adaptive architecture for ML-enabled systems [2].

References

1 K. Vaidhyanathan, A. Chandran, H. Muccini and R. Roy, "Agile4MLS—Leveraging Agile Practices for Developing Machine Learning-Enabled Systems: An Industrial Experience," in IEEE Software, vol. 39, no. 6, pp. 43-50, Nov.-Dec. 2022, doi: 10.1109/MS.2022.3195432.

2 K.Vaidhyanathan "Data-Driven Self-Adaptive Architecting Using Machine Learning," Ph.D. dissertation, Gran Sasso Science Institute, Italy, 2021, online at: http://hdl.handle.net/20.500.12571/15976

3.16 Generative AI at Fraunhofer and Research Roadmaps from the Software Architecture Community

Ingo Weber (TU München – Garching, DE)

Fraunhofer is the world's largest organization focused on application-oriented research. Several of its institutes are at the forefront of research in generative AI, but the recent advances make it a topic relevant for all industry sectors and hence potentially all Fraunhofer customers and institutes. Accordingly, Fraunhofer is broadening and deepening its AI endeavours. One early step was the introduction of fhGPT, an AI chatbot based on Azure OpenAI GPT models. Furthermore, from the ICSA-lite conference 2022, a book of high relevance for this Dagstuhl seminar emerged, which is currently in print: Patrizio Pelliccione, Rick Kazman, Ingo Weber, Anna Liu, editors. Software Architecture – Research Roadmaps from the Community. Springer, 2023.

3.17 Building, Engineering & Operating Systems for Critical Infrastructure

Roland Weiss (ABB – Mannheim, DE)

Control systems for process automation plants run business and mission critical infrastructure. They span various verticals, from chemical to power plants, from hydrogen generation to waste incineration. These applications require flawless execution 24 by 7, as malfunctions in the system can endanger people, the environment, or business targets. On the other hand, engineering such systems as well as operating them could benefit tremendously from the introduction of machine learning and artificial intelligence in general. In my talk, I explore the application of ML and AI in such systems.

In the engineering phase, generative AI has proven very promising in generating automation code (in IEC 61131 structured text). Based on a specification entered via a prompt interface, the code inside function blocks could be generated. Likewise, process graphics were translated into rules from older systems. Then, trained models validated the results and highlighted areas that deviated from the expected outcome. When running a process plant, operators have to monitor the system and, in the case of deviations or emergencies, interact with the system to either bring it to a safe state or to the desired stable state. Today, this requires very detailed domain know-how and extensive training. We are now

exploring supporting the operators with ML/AI-based assistance systems. For example, in case of deteriorating product quality in a chemical facility, the operator can pull up similar occurrences from the past and get recommendations from the assistance systems on how to handle the situation.

The key challenges we are facing when introducing AI/ML components into control systems are the following:

- The behavior of the system must be explainable. In regulated and safety-critical environments, the system providers need to be able to provide detailed RCAs in case of incidents.
- Training data is hard to get for various reasons. The collection of the data can't interfere with the process itself. Also, the incidents are typically rare, and operators aim to avoid them in the first place.
- Last but not least, these systems have extremely long lifetimes, thus, introducing ML means bringing them into legacy systems with traditional development processes.

We are looking forward to tackling these research challenges with Academia to contribute to the collection of best practices for building systems for critical infrastructure in order to make the world a safer and more sustainable place.

3.18 Challenges of Integrating ML Models in Safety-Relevant Architectures

Marc Zeller (Siemens – München, DE)

Traditional automation technologies alone are not sufficient to enable the fully automated operation of trains. However, Artificial Intelligence (AI) and Machine Learning (ML) offer great potential to realize the mandatory novel functions to replace the tasks of a human train driver, such as obstacle detection on the tracks. The problem, which still remains unresolved, is to find a practical way to link AI/ML techniques with the requirements and approval processes that are applied in the railway domain. The safe.trAIn project aims to lay the foundation for the safe use of AI/ML to achieve the driverless operation of rail vehicles and thus addresses this key technological challenge hindering the adoption of unmanned rail transport. The project goals are to develop guidelines and methods for the reliable engineering and safety assurance of ML in the railway domain. Therefore, the project investigates methods to reliably design ML models and to prove the trustworthiness of AI-based architectures, taking robustness, uncertainty, and transparency aspects of the AI/ML model into account.

3.19 Software Architecture for Foundation Model-Based Systems

Liming Zhu (Data61, CSIRO – Sydney, AU)

With the successful implementation of Large Language Models (LLMs) in chatbots like ChatGPT, there is growing attention on foundation models (FMs), which are anticipated to serve as core components in the development of future AI systems. Yet, systematic

exploration into the design of foundation model-based systems, particularly concerning risk management, trust, and trustworthiness, remains limited. In this talk, I propose the challenges and initial approaches in both architecting FM-based systems and how FMs have an impact on software engineering. I point to some initial directions, such as architecting as a process of understanding (rather than designing/building), designing guardrails (rather than quality attributes), and radical observability.

4 Working groups

The seminar provided an extensive and insightful exploration into the four key quality attributes that were prioritized by the attendees as critically important while architecting ML-enabled systems: *Data Centricity, Evolvability, Observability and Uncertainty, Trust and Trustworthiness*. The four working group (WG) discussions focusing on each quality attribute concern provided input for formulating the research agenda and key challenges to address. In this section, we describe the scope of each working group, and in the Seminar Overall Results section, we summarize the initial research roadmap and open challenges.

4.1 WG1: Architecting for Data Centricity

Jan Bosch (Chalmers University of Technology – Göteborg, SE), Benjamin Klöpper (Capgemini – Stuttgart, DE), Ipek Ozkaya (Carnegie Mellon Software Engineering Institute – Pittsburgh, US), Lena Pons (Carnegie Mellon Software Engineering Institute – Pittsburgh, US), and Christoph Schröer (Universität Oldenburg, DE)

The main purpose of the WG1 group on "Data Centricity" was to explore and address the pivotal role of architectural elements that extract, transform, load, store, and share data in the architectures of ML and analytics systems. The group focused on examining the entire data life cycle – from acquisition to transformation to runtime consumption – and understanding how data is integrated and managed by an ML-enabled system throughout this data life cycle.

The discussions emphasized that effective data management throughout the data life cycle is crucial for the success of ML-enabled systems. This includes recognizing the challenges associated with the complexities of data architectures. The WG1 group also identified the need for a distinct architectural view that focuses specifically on data, given its critical importance in ML-enabled systems. They acknowledged that while current software architecture practices and architectures of non-ML systems address data processing and storage, there is a need for a more focused approach to identify strategies for addressing data-centricity to cater to the unique requirements and challenges posed by ML systems.

In summary, the WG1 group's discussions emphasize the centrality of data in ML systems and the need for specialized and clearly communicated architectural approaches to manage and utilize data in these systems effectively.

4.2 WG2: Evolvability

Justus Bogner (Universität Stuttgart, DE), Jan Bosch (Chalmers University of Technology – Göteborg, SE), Helena Holmström Olsson (Malmö University, SE), Henry Muccini (University of L'Aquila, IT), Raghu Reddy (IIIT – Hyderabad, IN), Anastas Stoyanovsky (Amazon – Pittsburgh, US), Ingo Weber (TU München – Garching, DE), and Liming Zhu (Data61, CSIRO – Sydney, AU)

The primary objective of the WG2 group on "Evolvability" was to address the unique challenges in developing ML-enabled systems that can evolve efficiently in response to changing requirements and environmental factors. The group's primary focus was on understanding the dynamics of ML components within these systems, particularly how they interact with non-ML components and Machine Learning Operations (MLOps) infrastructures and practices.

A key finding of WG2 was the critical role of MLOps in ensuring the evolvability of ML components within ML-enabled systems. MLOps, supporting the entire life cycle of ML components from development to deployment and maintenance, emerged as a pivotal factor in managing and automating the continuous evolution of these systems. The group identified that the integration of MLOps within existing DevOps practices is not only essential but also challenging, given the unique characteristics of ML components.

Another significant aspect highlighted by WG2 was the concept of technical debt in ML systems. ML-enabled systems bring forth new classes of design and architectural challenges related to degradation in data or models as technical debt, which can impede the maintenance and evolution of ML-enabled systems. Addressing these forms of technical debt is crucial for sustaining the long-term viability and evolvability of these systems.

WG2 also emphasized the importance of the skill sets and roles of architects designing and maintaining ML-enabled systems. Architects need to possess a comprehensive understanding of aspects such as data engineering, model engineering, and ethical considerations to effectively manage the evolution of these complex systems.

In essence, the WG2 group underscores the complexities involved in creating evolvable ML-enabled systems and highlights the importance of MLOps, technical debt management, and specialized architectural knowledge in addressing these challenges.

4.3 WG3: Observability and Uncertainty

Nelly Bencomo (Durham University, GB), Xavier Franch (UPC Barcelona Tech, ES), Fabio Kon (University of Sao Paulo, BR), Ipek Ozkaya (Carnegie Mellon Software Engineering Institute – Pittsburgh, US), Marie Platenius-Mohr (ABB – Ladenburg, DE), Bradley Schmerl (Carnegie Mellon University – Pittsburgh, US), Roland Weiss (ABB – Mannheim, DE), and Karthik Vaidhyanathan (IIIT – Hyderabad, IN)

The working group on "Observability and Uncertainty" primarily focused on how uncertainty and observability were intertwined and how they impact the design, implementation, and operation of ML-enabled systems.

A key finding of the group was the often inherent non-deterministic and statistical nature of ML techniques, which introduces a fundamental level of uncertainty in these systems. This uncertainty is not just a challenge but a necessary aspect to consider in the architecture of ML-enabled systems. The group discussed two main ways to handle uncertainty: (1) reducing or eliminating uncertainty through strategies similar to using redundancy in reliability engineering and (2) managing uncertainty, which includes strategies for dealing with incomplete information and uncertain components.

The group's discussions underscored the close relationship between uncertainty and observability. Observability – the ability to infer the internal state of a system based on its output – is crucial for understanding and managing uncertainty in ML systems. The degree of uncertainty dictates the requirements for observability, impacting various other aspects of software development, including trustworthiness and evolvability.

In summary, WG3's outcomes emphasize the importance of treating uncertainty as a primary concern in the design and implementation of ML-enabled systems. In addition, the discussions highlight the need for effective observability mechanisms to identify and manage the inherent uncertainties of ML-enabled systems to ensure their reliability and effectiveness.

4.4 WG4: Architecting for Trust and Trustworthiness

Tao Chen (University of Birmingham, GB), Thomas Kropf (Robert Bosch GmbH – Renningen, DE), Grace A. Lewis (Carnegie Mellon Software Engineering Institute – Pittsburgh, US), Henry Muccini (University of L'Aquila, IT), Alex Serban (Siemens Healthineers, Erlangen, DE & University Transilvania of Brasov, RO), Romina Spalazzese (Malmö University, SE), and Marc Zeller (Siemens – München, DE)

The main purpose of WG4 was to explore the key aspects and challenges in designing trustworthy ML systems, such as in safety-critical and autonomous contexts.

The outputs of the discussions in this group emphasize the distinction between trust and trustworthiness and various aspects of these concepts in ML-enabled systems, including state-of-the-art trustworthy AI, the role of trust as an element of overall responsible AI practices and their relevance to ML-enabled system development, and the engineering of trustworthy ML-enabled systems. It also examines the importance of providing evidence of trustworthiness tailored to different stakeholders for calibrated trust, such as through certification labels and empirical studies, and discusses enabling the design of trustworthy ML-enabled systems through architectural models and toolkits.

A significant portion of the discussions were dedicated to addressing the challenges in evaluating the trustworthiness of ML-enabled systems and the role of human-in-the-loop in ensuring trustworthiness. The discussions in this group focused on the trust and trustworthiness of ML-enabled systems and had a close relationship with considerations that apply to AI systems in general, such as responsible AI practices. The discussions, in particular, emphasized the need for explainable AI as a key component in building trust in ML-enabled systems.

In summary, WG4's observations emphasize the critical need for defining trust and trustworthiness clearly in ML-enabled systems, exploring the complexities of achieving the several qualities essential in building trust and trustworthiness and developing and extending frameworks and strategies to guide the development of trustworthy applications with ML components.

5 Open problems

The SA&ML research roadmap, which is one of the key outcomes of this Dagstuhl Seminar, is heavily influenced by the high-priority quality attributes that were discussed in depth in the working groups and their relationships. Figure 1 provides an overview of the relationships between the quality attributes and the interplay that exists between them, with Data Centricity being a cross-cutting quality attribute. It also shows Explainability and Adaptability as two additional key quality attributes that contribute to these relationships. The following subsections provide a summary of how each attribute was scoped by the working groups, open problems in each area, and a list of high-priority research areas that need to be explored in the near term to advance the practice of SA&ML.

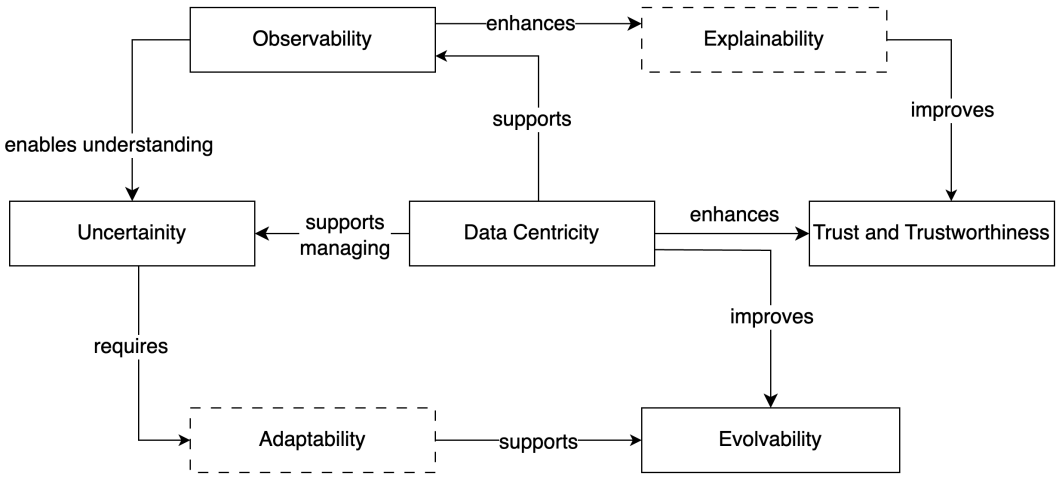

Figure 1 Relationships between the different quality attributes.

5.1 Data Centricity

Data-centricity emphasizes the significance of data quality and management for ML-enabled systems. High-quality, well-managed data is crucial for the development of reliable ML models and for maintaining the trustworthiness of the system.

Data-centricity also impacts evolvability and adaptability, as systems that can adapt to changes in data or requirements are more sustainable over time.

5.1.1 Challenges and Open Problems

The seminar looked beyond basic data integrity and security issues, delving into more complex and ML-relevant issues such as ensuring data relevance over time, managing large and diverse datasets, and maintaining compliance with evolving data privacy laws and standards. The future directions suggested include developing data views that reflect architecturally-significant concerns, understanding architectural implications of online vs offline learning, and data decoupling strategies.

5.2 Uncertainty and Observability

Uncertainty is a fundamental aspect of ML-enabled systems due to the often non-deterministic and statistical nature of ML models. Uncertainty can arise from the quality of training data, the model design, interactions between ML and other components, and the environment in which the system operates.

Observability involves the ability to monitor the internal states and outputs of the system to enable understanding and make informed decisions about its functioning, especially in the face of uncertainty. Observability helps system developers and operators understand and manage uncertainty.

Observability can also contribute to enhancing explainability, as more observable systems provide more information that can be used to explain inference results to users and for engineers to diagnose systems. Observability also contributes to evolvability as runtime data can be used to determine, for example, when a model needs to be retrained in response to drift.

5.2.1 Challenges and Open Problems

The major challenge identified by the working group participants lies in quantifying and managing uncertainty, which is inherent in ML-enabled systems due to their often non-deterministic nature. Another challenge is developing methods for effective observability that can aid in explaining the behavior of ML systems. Future research is required to enhance methods for quantifying uncertainty, properly propagating uncertainty throughout the life cycle and system, developing advanced observability techniques, and exploring the impact of uncertainty on trustworthiness and evolvability. There is also an emphasis on creating strategies for adaptive actions in response to observed properties, including uncertainties.

5.3 Evolvability (and Adaptability)

Evolvability was defined by the seminar participants as a system's ability to adapt to changes in expectations (i.e., requirements), while adaptability refers to the ability of the system to handle (or be modified to deal with) changes in its environment.. Evolvability, therefore, is mostly a design-time concern, whereas **adaptability** is both a design-time and runtime concern. These attributes are essential for designing ML-enabled systems, as these systems must continuously evolve and adapt to remain effective. Evolvability and adaptability are influenced by observability and uncertainty, and together, they contribute towards maintainability; by addressing these attributes proactively and intentionally, systems can be designed to evolve and adapt more effectively. Evolvability and adaptability are also closely linked to data centricity, as the system's ability to evolve and adapt is dependent on the quality and relevance of the data it uses and how its evolution and adaptation are influenced by changes in data.

5.3.1 Challenges and Open Problems

Participants unanimously agreed that the main challenge in evolvability and adaptability lies in integrating continuous development and deployment practices for ML components, such as MLOps, with software development and deployment of non-ML components. Addressing

the accruing technical debt and managing quality attributes of ML components are also highlighted as significant challenges. Further, participants suggest exploring effective integration of MLOps practices, identifying new evolvability and adaptability-specific technical debt items in ML-enabled systems, and conceptualizing and realizing architectural designs that support continuous change. Developing guidelines for managing the interaction between ML and non-ML components is also seen as a key future direction where more case studies and research need to be developed.

5.4 Trust and Trustworthiness

Trust is the subjective confidence stakeholders place in a system's quality. It is influenced by the system's trustworthiness.

Trustworthiness involves the system's objective ability to consistently perform according to quality expectations and ethical standards. It is bolstered by explainability, as stakeholders are more likely to trust a system they can understand. Trust and trustworthiness are impacted by data centricity, as the quality and management of data underpin the reliability and robustness of ML-enabled systems. Related, if the system provides better explainability, it fosters trust in the system. A system that is trustworthy (through evidence of achieving quality attributes such as security, reliability, and fairness) merits the trust placed in it by users.

5.4.1 Challenges and Open Problems

The participants elaborated on the challenges of balancing system transparency and explainability while complying with ethical standards and legal regulations in different domains. Proposed future research directions included the development of comprehensive frameworks for building trustworthy AI systems, tailoring trustworthiness evidence to different stakeholders for calibrated trust, exploring human-in-the-loop approaches to balance automation and human oversight, and creating dynamic processes for evolving trust and trustworthiness in line with technological and societal shifts.

5.5 High Priority Research Areas to Advance SA&ML

Several open research areas in SA&ML were identified as a result of the seminar discussions. High-priority research areas requiring near-term focus for advancing the state of the practice include the following:

- Architectural Design for Data-Centricity: Identifying and developing architectural approaches that effectively address the central role of data in ML-enabled systems, including data acquisition, processing, and management, as well as designing systems that can adapt to changes in data characteristics, need attention from researchers as well as practitioners in documenting their lessons learned. Existing architecture patterns, tactics, and quality attributes related to data architecting need to also be shared more consistently, and gaps need to be filled where applicable.
- Evolvability and adaptability of MLOps Architectures: Creating architectural designs that support the evolvability and adaptability of ML-enabled systems, particularly focusing on leveraging MLOps infrastructure, is not common practice. Challenge areas include addressing the lifecycle management of ML models and ensuring systems can evolve and adapt to new and changing requirements and environments.

- Uncertainty as a First-Class Concern: Integrating the management of uncertainty into the architectural design process for ML-enabled systems first requires identifying sources of uncertainty which can effectively be managed with architectural approaches. This involves developing methods for quantifying and mitigating uncertainty that can be leveraged by architecture practices and constructs.
- Observability in ML-enabled Systems: Existing approaches in system observability need to be enhanced to better observe and manage the behavior of ML components. This includes developing metrics and tools for monitoring and understanding ML components and ML-enabled system states.
- Trust, Trustworthiness, and Ethical Considerations: Building trustworthiness into the architecture of ML-enabled systems, including designing for ethical considerations, compliance with regulations, and ensuring transparency and explainability of AI decisions, is a growing need. Identifying what aspects of trustworthiness and ethical considerations can be handled architecturally and clearly communicating the remaining gaps is critical as research in this area progresses.
- Human-in-the-Loop AI Decision Making: Architecting systems that effectively incorporate human oversight and interaction, particularly in critical decision-making processes to ensure balanced autonomy and control in AI systems, will be an area of research that influences many quality attribute concerns, as well as how responsibilities are allocated to architectural elements, potentially resulting in new architecture patterns and tactics.

5.6 Follow-up Work

At the seminar, participants recognized that there is a lack of understanding and common definitions for many concepts related to ML-enabled systems and existing architecture practices and fundamentals. Furthermore, in addition to the set of key quality attributes discussed in detail and shared in this report, other quality attributes that require attention from the software architecture community were identified. Hence, a more detailed description of the concepts identified, extended discussions around the quality attributes discussed, as well as a more comprehensive explanation of the research roadmap, are planned for publication in appropriate venues.

Participants

- Nelly Bencomo
Durham University, GB
- Justus Bogner
Universität Stuttgart, DE
- Jan Bosch
Chalmers University of Technology – Göteborg, SE
- Tao Chen
University of Birmingham, GB
- Xavier Franch
UPC Barcelona Tech, ES
- Helena Holmström Olsson
Malmö University, SE
- Benjamin Klöpper
Capgemini – Stuttgart, DE
- Fabio Kon
University of Sao Paulo, BR
- Thomas Kropf
Robert Bosch GmbH – Renningen, DE
- Grace A. Lewis
Carnegie Mellon University – Pittsburgh, US
- Henry Muccini
University of L'Aquila, IT
- Ipek Ozkaya
Carnegie Mellon University – Pittsburgh, US
- Marie Platenius-Mohr
ABB – Ladenburg, DE
- Lena Pons
Carnegie Mellon University – Pittsburgh, US
- Raghu Reddy
IIIT – Hyderabad, IN
- Bradley Schmerl
Carnegie Mellon University – Pittsburgh, US
- Christoph Schröer
Universität Oldenburg, DE
- Alex Serban
Siemens Healthineers, Erlangen, DE & University Transilvania of Brasov, RO
- Romina Spalazzese
Malmö University, SE
- Anastas Stoyanovsky
Amazon – Pittsburgh, US
- Karthik Vaidhyanathan
IIIT – Hyderabad, IN
- Ingo Weber
TU München – Garching, DE
- Roland Weiss
ABB – Mannheim, DE
- Marc Zeller
Siemens – München, DE
- Liming Zhu
Data61, CSIRO – Sydney, AU