

Quantum Software Engineering

Shaukat Ali^{*1}, Johanna Barzen^{*2}, Andrea Delgado^{*3},
Hausi A. Müller^{*4}, and Juan Manuel Murillo^{*5}

1 Simula Research Laboratory – Oslo, NO. shaukat@simula.no

2 Institute of Architecture of Application Systems (IAAS) – Stuttgart, DE.
johanna.barzen@iaas.uni-stuttgart.de

3 Oak Ridge National Laboratory (ORNL) – US. delgadoa@ornl.gov

4 University of Victoria, CA. hausi@uvic.ca

5 University of Extremadura – Cáceres, ES. juanmamu@unex.es

Abstract

The Dagstuhl Seminar 24512 on “Quantum Software Engineering” was held from December 15 to 20, 2024. It brought together 26 participants from industry and academia from 13 different countries, including senior and junior researchers as well as practitioners in the field of Quantum Software Engineering. The aim of the seminar was to advance software engineering methods and tools for the engineering of hybrid quantum systems by promoting personal interaction and open discussion among researchers who are already working in this emerging area of knowledge. The first day of the seminar was devoted to the topic “When software engineering meets quantum mechanics”, while the second day focused on “Quantum software engineering and its challenges.” During both days, 16 invited presentations were given. The rest of the seminar was organized into three working groups to address the topics “Quantum Software Design, Modelling and Architecturing”, “Adaptive Hybrid Quantum Systems”, and “Quantum Software Quality Assurance”. The seminar was a very fruitful experience for all participants both in terms of scientific outcomes and in terms of the personal relationships that were generated to jointly address future experiences.

Seminar December 15–20, 2024 – <https://www.dagstuhl.de/24512>

2012 ACM Subject Classification Computing methodologies; Theory of computation → Models of computation; Computer systems organization → Quantum computing; Theory of computation → Quantum information theory; Software and its engineering; Theory of computation

Keywords and phrases adaptive hybrid quantum systems, Linear algebra, Quantum algorithms, Quantum architecture, Quantum circuit compilation, Quantum computing, Quantum machine learning, Quantum modeling, Quantum networking, Quantum optimization, Quantum runtime systems, Quantum simulation, quantum software design, Quantum software development, Quantum software engineering, quantum software quality assurance, Quantum software stack and platforms

Digital Object Identifier 10.4230/DagRep.14.12.63

* Editor / Organizer



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 4.0 International license

Quantum Software Engineering, *Dagstuhl Reports*, Vol. 14, Issue 12, pp. 63–84

Editors: Shaukat Ali, Johanna Barzen, Andrea Delgado, Hausi A. Müller, and Juan Manuel Murillo



Dagstuhl Reports

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Executive Summary

Shaukat Ali (Simula Research Laboratory – Oslo, NO)

Johanna Barzen (Institute of Architecture of Application Systems (IAAS) – Stuttgart, DE)

Andrea Delgado (Oak Ridge National Laboratory (ORNL) – US)

Hausi A. Müller (University of Victoria – CA)

Juan Manuel Murillo (University of Extremadura – Cáceres, ES)

License  Creative Commons BY 4.0 International license

© Shaukat Ali, Johanna Barzen, Andrea Delgado, Hausi A. Müller, and Juan Manuel Murillo

Under the umbrella of quantum information theory, quantum algorithms have been proposed that solve polynomial-time problems for which no classical algorithm solves in that order of complexity. The entry into the NISQ era with the development of the first quantum computers and simulators makes testing such algorithms on real computers possible. This generates new expectations and promotes growing interest in Quantum Computing. Researchers are proposing solutions in the quantum domain to optimize existing algorithmic approaches regarding the number of qubits or circuit depth, propose solutions to manage decoherence and quantum errors, or propose new algorithms for unexplored problems. All these efforts, coupled with the fact that numerous media reports on quantum computing achievements appear, are inviting researchers and practitioners from industry, academia and government to investigate the utility of quantum computing to explore the possibilities it can offer. Some countries have launched specific programs encouraging industry and academia to examine together the applicability in different sectors, the current limitations of quantum technologies and their expected evolution over time. The objective is to prepare for the future uptake of quantum computing in industry in many application domains.

All of the above is evidence that quantum software is already a reality. It is commonly accepted that future software will be hybrid, integrating both classical and quantum components because each quantum algorithm requires classical preprocessing or postprocessing. With this, interest is also beginning to appear from researchers in the field of software engineering. The question arises whether the software engineering body of knowledge acquired over the last decades is applicable in this new computing era. The first forums focusing on the field of Quantum Software Engineering (QSE) have already appeared. Examples are Q-SE at ICSE, QSW at IEEE Services or Q-SET at IEEE QCE. Many interesting works have been presented in these forums on approaching certain software engineering practices now in the context of quantum computing. Topics such as building hybrid software architectures, orchestrating quantum systems, building models that enable automatically generating quantum software, and creating quantum software with better quality attributes are being addressed in these forums.

Although all these works cover interesting specific aspects that different research teams have addressed, there is a need for a deeper reflection on the nature and scope of QSE. This reflection should address questions such as the differences between classical and quantum software engineering, about which features of quantum software can be addressed with the known classical techniques, which features cannot be addressed with classical techniques and why, which classical techniques should be extended or modified to address quantum software development and which new software engineering techniques should be developed to address quantum software from an engineering perspective.

As the area of QSE can be very large, the Dagstuhl Seminar on “Quantum Software Engineering” focused on three main work topics described below.

Quantum Software Design, Modelling, and Architecturing

Each software development methodology embodies its own design artefacts. The question arises as to which design artefacts of the methodologies suit quantum parts in hybrid systems. We may not have such artifacts; we need new ones or known artifacts with new features. In the same way, it could be asked if the conventional classical methodologies are adequate for the development of hybrid systems or if it is necessary to define or adapt our known methodologies. In particular, Model-driven engineering (MDE) has been one of the areas of research that has attracted significant interest over the years and has affected many changes in software engineering. It is now necessary to explore how its principles are adapted to capture the quantum nature of hybrid systems. Some of the contributions that could emerge are new design techniques to address hybrid software development, differentiation and integration of classical and quantum parts during the design, new design objects suitable to the characteristics of quantum computing, specific domain languages for quantum systems, platform-specific models for hybrid systems, reverse engineering of legacy systems for building new hybrid systems or model transformation for hybrid systems. Finally, how to compose hybrid systems from simpler classical and quantum parts is one of the aspects that we should most rapidly address to facilitate the construction of large-scale computational systems involving quantum computation. There are interesting aspects to be addressed here, such as the orchestration of quantum programs, techniques to deal adequately with the classical preprocessing and postprocessing needed by quantum algorithms, and techniques for the interaction between quantum and classical code that do not avoid the benefits provided by service-oriented computing.

Adaptive Hybrid Quantum Systems

Quantum systems combine classical computers with quantum processors to realize hybrid solutions to practical computing problems, including optimization, machine learning and nature simulation. Classical computers are involved in every layer of the quantum stack, from high-level algorithm design to low-level qubit control. The success of quantum computing relies on the seamless integration of classical and quantum components. Optimizing performance requires considering the interactions between different layers of the quantum stack. Thus, the quantum stack demonstrates the inherent hybrid nature of quantum systems.

Quantum system adaptivity is equally essential to the success of quantum computing. Adaptive quantum systems dynamically optimize the use of quantum and classical resources during computation at different levels of the quantum stack. Adaptive systems rely on various technologies to achieve self-* properties, such as self-healing, self-optimizing, self-configuring, or self-adjusting behavior. Key technologies to adapt different levels of the quantum stack include quantum runtimes, parameterized circuits, mid-circuit measurements, dynamic circuits, and circuit weaving to achieve real-time quantum error correction, dynamic calibration of qubit properties, noise mitigation, parameter adjustments in variational algorithms, qubit entanglement routing, or adaptive circuit decomposition. These highly dynamic quantum systems will significantly benefit from software engineering technologies such as feedback loops, models at runtime, assurance at runtime, autonomic systems, digital twins, software-defined infrastructure, and compute & storage clouds.

The rapid proliferation of distributed quantum computing (DQC) amplifies the need for adaptive quantum system technologies to help optimize entanglement distribution across quantum chips, manage network dynamics & latency, and problem decomposition across the network. Adaptive technologies are the key to unlocking the full potential of DQC and enabling the development of large-scale, fault-tolerant, and resilient quantum computers.

Quantum Software Quality Assurance

The construction of software from an engineering perspective always demands that the product obtained has adequate quality attributes. Building a product that is not profitable due to its final characteristics or the cost necessary for its construction is not a valid solution. Quantum software cannot suffer from the lack of these quality attributes. So, it is necessary to guarantee attributes such as reusability or maintainability. How to promote these attributes during the process of building new quantum software is still a task that needs to be addressed. Related to the task of guaranteeing that software meets the required quality attributes is the discipline of software testing. Software testing groups together a set of practices aimed at examining software artefacts and their operation to guarantee that the software has adequate quality attributes and behaves adequately with respect to what it was conceived for. In classic software, many types of testing can be practised, such as white-box, black-box, grey-box, dynamic, or static. It is necessary to determine whether all these testing types are applicable during quantum software development. Also, when dealing with quantum software, it is necessary to study the need for different testing, at least at two levels. On the one hand, testing the classical software that generates the quantum circuit requires further investigation, where classical testing techniques can be applied. On the other hand, testing quantum circuits requires studying new testing techniques. Related to the above, formal methods for ensuring correctness and checking quantum software are also engaging in this topic.

Seminar agenda and development

The seminar agenda started on the first day with a session dedicated to personal presentations, where attendees described their focus area and the related problems that interest them. Next, the rest of the day was devoted to the topic “When Software Engineering meets Quantum Mechanics.” Five interesting talks were given on this topic.

The talks continued on the second day to address “Quantum Software Engineering and its Challenges.” Eleven talks were given on this topic.

From the third day onwards, the seminar focused on discipline-specific group discussions. To build groups of interest for all attendees, the theme of the groups and dynamics were discussed on the morning of the third day.

During each day, coffee breaks were scheduled with sufficient time for interaction and free discussion among the participants. In addition, at the end of the first, second and fourth days, we orchestrated sessions for plenary discussion on topics of relevance to the participants. During the seminar, a board was available for interested attendees or groups of attendees to write down the topics they would like to address during those highly interactive plenary sessions.

During the off-schedule time, the participants enjoyed much social interaction, including long singing sessions accompanied by the excellent guitar player Prof. Hausi A. Müller. This singing was not unrelated to the discipline of quantum computing. Far from it, a song written by Prof. Müller was rehearsed and performed multiple times by the attendees in the renowned Dagstuhl music room and on stage at the closing session on Friday morning. The lyrics of this song are also an outcome of the seminar and are therefore included as a panel discussion at the end of this report.

2 Table of Contents

Executive Summary

<i>Shaukat Ali, Johanna Barzen, Andrea Delgado, Hausi A. Müller, and Juan Manuel Murillo</i>	64
--	----

Overview of Talks

Quantum Software Testing <i>Paolo Arcaini</i>	69
Distributing Quantum Computations <i>Antonio Brogi</i>	70
Quantum Computing, Consciousness, and Software Engineering <i>Schahram Dustdar</i>	71
KetGPT – Dataset Augmentation of Quantum Circuits using Transformers <i>Sebastian Feld</i>	71
Quantum Software Ecosystem Design <i>Michael Felderer</i>	72
Quantum Services: Load balancing and scheduling to improve usage of QPUs <i>José Manuel García Alonso</i>	73
Quantum Software = Quantum minus Software? <i>Wolfgang Maurer</i>	73
Advancing Quantum Software for Software Engineering Tasks: Current Limits and Future Possibilities <i>Andriy Miranskyy</i>	74
Quantum Software Consortium: Exploring Distributed Quantum Software Solutions <i>Hausi A. Müller</i>	74
Classiq: Electronic design automation synthesis methods for quantum software <i>Yehuda Naveh</i>	75
A software product line approach for developing hybrid quantum-classical software <i>Ricardo Pérez-Castillo</i>	75
Generation of Fixed Margin Binary Matrices using Quantum Annealing <i>Rui Abreu</i>	76
Towards Reliable Quantum Computing Architecture <i>Shinobu Saito</i>	76
Towards View-Based Development of Quantum Software <i>Ina Schaefer and Wolfgang Maurer</i>	77
On HPCQC Software Stacks Needs <i>Laura Schulz</i>	77
Scalable approaches to constrained combinatorial optimization <i>Ulrike Stege</i>	78
Design Automation for Quantum Computing <i>Robert Wille</i>	78

Multi-Objective Optimization for Quantum Programs <i>Manuel Wimmer</i>	78
Working groups	
Report of the working group on “Quality assurance for quantum programs” <i>Paolo Arcaini, Shaukat Ali, José Campos, Ignacio García Rodríguez de Guzmán, Andriy Miransky, Anila Mjeda, and Lei Zhang</i>	79
Adaptive Hybrid, Distributed Quantum Systems <i>Antonio Brogi, Shahram Dustdar, Michael Felderer, Hausi A. Müller, Juan Manuel Murillo, Ricardo Pérez-Castillo, Antonio Ruiz Cortés, Shinobu Saito, and Ulrike Stege</i>	81
Software Abstraction in Quantum Computing <i>Sebastian Feld, Michael Falkental, Michael Felderer, José Manuel García Alonso, Wolfgang Maurer, Yehuda Naveh, Ina Schaefer, Laura Schulz, Robert Wille, and Manuel Wimmer</i>	82
Panel discussions	
A Quantum Computing Song: Qubit Entanglement <i>Hausi A. Müller</i>	83
Participants	84

3 Overview of Talks

3.1 Quantum Software Testing

Paolo Arcaini (*National Institute of Informatics – Tokyo, JP*)

License © Creative Commons BY 4.0 International license
© Paolo Arcaini

Joint work of Paolo Arcaini, Shaukat Ali, Eñaut Mendiluze, Asmar Muqeet, Xinyi Wang, Tongxuan Yu, Tao Yue

As for classical software, also quantum programs need to be tested in order to assess whether they behave as expected. *Quantum Software Testing* is a novel research area that develops testing techniques tailored for quantum programs. The talk provides an overview of a series of works we proposed in this area.

In testing, *coverage criteria* specify what needs to be covered by a test suite in order to have some correctness guarantees. In [1, 6], we proposed some criteria related to the inputs and the outputs of quantum programs. The criteria proposed in [1, 6] may require several tests to be achieved; for example, input coverage requires to cover all the inputs. In order to obtain more compact (but still effective) test suites, in [4, 8] we proposed a combinatorial testing approach for quantum programs that only requires to cover specific combinations of the inputs.

Another classical way to assess the quality of a test suite is *mutation testing*. In [2], we proposed a mutation testing tool for quantum programs that provides mutation operators that target the different types of gates of a quantum circuit. If some mutants are not killed, one should augment the test suite with tests killing these mutants. In [9], we proposed a search-based approach that is able to generate the minimum number of tests that kill as many mutants as possible.

Another goal in testing is to find tests that expose failures of the program. In [5, 7], we proposed a search-based approach that searches different failing tests for the program under test.

Existing testing approaches for quantum programs suffer from some limitations, such as requiring a full program specification, and being incompatible with error mitigation methods. To address these issues, in [3], we proposed a new definition of test cases based on *Pauli strings*, and that takes advantage of the possibility to aggregate Pauli strings in *Pauli families*.

References

- 1 Shaukat Ali, Paolo Arcaini, Xinyi Wang, and Tao Yue. Assessing the effectiveness of input and output coverage criteria for testing quantum programs. In *2021 IEEE 14th International Conference on Software Testing, Validation and Verification (ICST)*, pages 13–23, 2021.
- 2 Eñaut Mendiluze, Shaukat Ali, Paolo Arcaini, and Tao Yue. Muskit: A mutation analysis tool for quantum software testing. In *Proceedings of the 36th IEEE/ACM International Conference on Automated Software Engineering, ASE '21*, pages 1266–1270. IEEE Press, 2022.
- 3 Asmar Muqeet, Shaukat Ali, and Paolo Arcaini. Quantum program testing through commuting Pauli strings on IBM's quantum computers. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering, ASE '24*, pages 2130–2141, New York, NY, USA, 2024. Association for Computing Machinery.
- 4 Xinyi Wang, Paolo Arcaini, Tao Yue, and Shaukat Ali. Application of combinatorial testing to quantum programs. In *2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)*, pages 179–188, 2021.

- 5 Xinyi Wang, Paolo Arcaini, Tao Yue, and Shaukat Ali. Generating failing test suites for quantum programs with search. In *Search-Based Software Engineering: 13th International Symposium, SSBSE 2021, Bari, Italy, October 11–12, 2021, Proceedings*, pages 9–25, Berlin, Heidelberg, 2021. Springer-Verlag.
- 6 Xinyi Wang, Paolo Arcaini, Tao Yue, and Shaukat Ali. Quito: A coverage-guided test generator for quantum programs. In *Proceedings of the 36th IEEE/ACM International Conference on Automated Software Engineering, ASE '21*, pages 1237–1241. IEEE Press, 2022.
- 7 Xinyi Wang, Paolo Arcaini, Tao Yue, and Shaukat Ali. QuSBT: Search-based testing of quantum programs. In *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings, ICSE '22*, pages 173–177, New York, NY, USA, 2022. Association for Computing Machinery.
- 8 Xinyi Wang, Paolo Arcaini, Tao Yue, and Shaukat Ali. QuCAT: A combinatorial testing tool for quantum software. In *Proceedings of the 38th IEEE/ACM International Conference on Automated Software Engineering, ASE '23*, pages 2066–2069. IEEE Press, 2024.
- 9 Xinyi Wang, Tongxuan Yu, Paolo Arcaini, Tao Yue, and Shaukat Ali. Mutation-based test generation for quantum programs with multi-objective search. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '22*, pages 1345–1353, New York, NY, USA, 2022. Association for Computing Machinery.

3.2 Distributing Quantum Computations

Antonio Brogi (University of Pisa, IT)

License © Creative Commons BY 4.0 International license
© Antonio Brogi

Joint work of Antonio Brogi, Giuseppe Bisicchia, Alessandro Bocci, Giuseppe Clemente, Massimo D’Elia, José García-Alonso, Juan Manuel Murillo

Main reference Giuseppe Bisicchia, José García-Alonso, Juan Manuel Murillo, Antonio Brogi: “Distributing Quantum Computations, by Shots”, in Proc. of the Service-Oriented Computing – 21st International Conference, ICSOC 2023, Rome, Italy, November 28 – December 1, 2023, Proceedings, Part I, Lecture Notes in Computer Science, Vol. 14419, pp. 363–377, Springer, 2023.

URL https://doi.org/10.1007/978-3-031-48421-6_25

Main reference Giuseppe Bisicchia, Giuseppe Clemente, José García-Alonso, Juan Manuel Murillo Rodríguez, Massimo D’Elia, Antonio Brogi: “Distributing Quantum Computations, Shot-wise”, CoRR, Vol. abs/2411.16530, 2024.

URL <https://doi.org/10.48550/ARXIV.2411.16530>

Main reference Giuseppe Bisicchia, Alessandro Bocci, José García-Alonso, Juan Manuel Murillo, Antonio Brogi: “Cut&Shoot: Cutting & Distributing Quantum Circuits Across Multiple NISQ Computers”, in Proc. of the IEEE International Conference on Quantum Computing and Engineering, QCE 2024, Montreal, QC, Canada, September 15-20, 2024, pp. 187–192, IEEE, 2024.

URL <https://doi.org/10.1109/QCE60285.2024.10276>

I presented the work that we have been carrying on to develop a methodology for distributing the shots of a quantum computation across multiple, heterogeneous Quantum Processing Units.

The proposed methodology features different policies for shot distribution as well as a calibration technique to pre-evaluate the accuracy and reliability of target QPUs.

I discussed the potential advantages of such a methodology both quantitatively (average/maximal error and deviation) and qualitatively (robustness to noise and resilience to QPU failures).

Finally, I briefly pointed to our most recent work, devoted to exploring the potential advantages of combining circuit-cutting and shot-wise distribution techniques.

3.3 Quantum Computing, Consciousness, and Software Engineering

Schahram Dustdar (TU Wien, AT)

License © Creative Commons BY 4.0 International license
© Schahram Dustdar

This talk has 3 parts: Part 1 explores 9 foundational challenges for applying software engineering principles to quantum computing. Part 2 discusses Federico Faggin’s OTP Theory in the light of principled challenges for Quantum Software. Part 3 develops several baselines for utilizing OTP theory as an inspiration for the emerging field of Quantum Software.

3.4 KetGPT – Dataset Augmentation of Quantum Circuits using Transformers

Sebastian Feld (Delft University of Technology, NL)

License © Creative Commons BY 4.0 International license
© Sebastian Feld
Joint work of Boran Apak, Medina Bandic, Aritra Sarkar, Sebastian Feld
Main reference Boran Apak, Medina Bandic, Aritra Sarkar, Sebastian Feld: “KetGPT – Dataset Augmentation of Quantum Circuits Using Transformers”, in Proc. of the Computational Science – ICCS 2024 – 24th International Conference, Malaga, Spain, July 2-4, 2024, Proceedings, Part VI, Lecture Notes in Computer Science, Vol. 14837, pp. 235–251, Springer, 2024.
URL https://doi.org/10.1007/978-3-031-63778-0_17

Quantum algorithms, represented as quantum circuits, can be used as benchmarks for assessing the performance of quantum systems [1]. Existing datasets, widely utilized in the field, suffer from limitations in size and versatility, leading researchers to employ randomly generated circuits. Random circuits are, however, not representative benchmarks as they lack the inherent properties of real quantum algorithms for which the quantum systems are manufactured [2]. This shortage of “useful” quantum benchmarks poses a challenge to advancing the development and comparison of quantum compilers and hardware, as also highlighted by recent studies on full-stack quantum computing systems [3]. Our goal is to enhance existing quantum circuit datasets by generating what we refer to as “realistic-looking” circuits by employing the Transformer machine learning architecture [4]. For this purpose, we introduce KetGPT [5], a tool that generates synthetic circuits in OpenQASM language [6], whose structure is based on quantum circuits derived from existing quantum algorithms and follows the typical patterns of human-written algorithm-based code (e.g., order of gates and qubits). Our three-fold verification process – comprising manual inspection and Qiskit framework execution, transformer-based classification, and structural analysis – demonstrates the efficacy of KetGPT in producing large amounts of additional circuits that closely align with algorithm-based structures.

References

- 1 Bandic, M., Almudever, C. & Feld, S. Interaction graph-based characterization of quantum benchmarks for improving quantum circuit mapping techniques. *Quantum Machine Intelligence*. **5**, 40 (2023)
- 2 Bandic, M., Henaff, P., Ovide, A., Escofet, P., Rached, S., Rodrigo, S., Someren, H., Abadal, S., Alarcón, E., Almudever, C. & Others: Profiling quantum circuits for their efficient execution on single-and multi-core architectures. *Quantum Science And Technology*. **10**, 015060 (2025)

- 3 Bandic, M., Feld, S. Almudever, C.: Full-stack quantum computing systems in the NISQ era: algorithm-driven and hardware-aware compilation techniques. *2022 Design, Automation & Test In Europe Conference & Exhibition (DATE)*. pp. 1-6 (2022)
- 4 Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, Ł. & Polosukhin, I. Attention is all you need. *Advances In Neural Information Processing Systems*. **30** (2017)
- 5 Apak, B., Bandic, M., Sarkar, A. & Feld, S. Ketgpt–dataset augmentation of quantum circuits using transformers. *International Conference On Computational Science*. pp. 235-251 (2024)
- 6 Cross, A., Bishop, L., Smolin, J. & Gambetta, J. Open quantum assembly language. *ArXiv Preprint ArXiv:1707.03429*. (2017)

3.5 Quantum Software Ecosystem Design

Michael Felderer (DLR – Köln, DE)

License  Creative Commons BY 4.0 International license
© Michael Felderer

The rapid advancements in quantum computing necessitate a rigorous approach to the construction of a corresponding software ecosystem that combines knowledge and expertise from quantum computing, software engineering and computer science. The German Aerospace Center (DLR) is currently developing a quantum software ecosystem integrating software and hardware from various academic and industrial stakeholders. This talk gives an overview of these activities, shares experiences and highlight different research perspectives in the context of quantum software ecosystem design. It discusses considerations essential for building a quantum software ecosystem that makes quantum computing available for scientific and industrial problem-solving. At the heart of this discourse is the concept of hardware–software co-design, which creates a bidirectional feedback loop from the application layer at the top of the software stack down to the hardware. This approach begins with compilers and low-level software that are specifically designed to align with the unique specifications and constraints of the quantum processor, proceeds with algorithms developed with a clear understanding of underlying hardware and computational model features, and extends to applications that effectively leverage the capabilities to achieve a quantum advantage. For this purpose, a suitable benchmarking framework must be developed and integrated at all levels of the software stack. We discuss the ecosystem from two critical perspectives: the conceptual view, focusing on theoretical foundations, and the software infrastructure view, addressing practical implementations around real quantum devices necessary for a functional ecosystem. The integration of these two perspectives raises novel research challenges for the quantum software engineering community that are highlighted. Following this approach ensures that the focus is toward promising applications with optimized algorithm–circuit synergy, while ensuring a user-friendly design, an effective data management, an overall orchestration strategy, suitable benchmarking for quantum readiness as well as a clean software and services architecture.

3.6 Quantum Services: Load balancing and scheduling to improve usage of QPUs

José Manuel García Alonso (*University of Extremadura – Cáceres, ES*)

License  Creative Commons BY 4.0 International license
© José Manuel García Alonso

In this talk we explored different approaches to improve the usage of QPUs inspired by classical service-oriented computing techniques.

First, we discussed a Quantum Task Manager [1] that integrates load balancing and resource allocation for efficient quantum task execution across multiple quantum providers. This solution aims to simplify the resource selection process for developers, making it easier for them to take full advantage of the resources offered by different quantum service providers. The Task Manager implementation adheres to the REST API architectural pattern, and it dynamically allocates tasks based on factors like the load of quantum resources, the number of pending tasks, and the availability of quantum machines. Its main goal is to improve resource availability and performance, delivering a higher quality of service for quantum task execution.

And then, the QCRAFT Scheduler [2] was reviewed. This tool reduces waiting times and execution costs while making greater use of quantum computer resources. To do that, circuit from different users are scheduled into combined circuits that can be executed together. This strategy provides a substantial improvement for the developers in terms of reducing queueing times and costs. Furthermore, the noise suffered by the execution of the combined circuits does not notably affect the obtained results. . These works underscore the potential of service-oriented computing methodologies and tools to improve resource utilization in quantum computing, facilitating the way for more efficient and accessible quantum software development.

References

- 1 Alvarado-Valiente, J., Romero-Alvarez, J., Moguel, E., Garcia-Alonso, J. & Murillo, J. Orchestration for quantum services: The power of load balancing across multiple service providers. *Sci. Comput. Program.* **237** pp. 103139 (2024), <https://doi.org/10.1016/j.scico.2024.103139>
- 2 Alvarado-Valiente, J., Romero-Alvarez, J., Casco-Seco, J., Moguel, E., Garcia-Alonso, J. & Murillo, J. Circuit Scheduling Policies on Current QPUs: QCRAFT Scheduler. *Service-Oriented Computing – 22nd International Conference, ICSOC 2024, Tunis, Tunisia, December 3-6, 2024, Proceedings, Part II*. **15405** pp. 195-209 (2024), <https://doi.org/10.1007/978-981-96-0808-9>

3.7 Quantum Software = Quantum minus Software?

Wolfgang Mauerer (*Ostbayerische Technische Hochschule – Regensburg, DE*)

License  Creative Commons BY 4.0 International license
© Wolfgang Mauerer

We cannot think about quantum software without acknowledging that we know only a handful of quantum algorithms – and these are almost trivial from a software engineering point of view. Based on empirical analyses of open source quantum software, it is known that little of “quantum software” is actually “quantum”, and core quantum aspects are

interleaved with large amounts of classical code. However, this is not necessarily a bad thing if we slightly rethink what “software” means in this context: I argue that by properly distinguishing between formal representations of (computational or problem) models and the underlying physical process that performs the actual work, it can be seen that transformation processes and the choice of appropriate model representations are at the core of quantum computation, and should receive appropriate attention as “software”. This leads to important unsolved questions that must be addressed by future research: Which models do perform useful computational tasks, how can we leverage their computational power, and how can we devise and automate the required model transformations from representations that have proved useful for traditional computation.

3.8 Advancing Quantum Software for Software Engineering Tasks: Current Limits and Future Possibilities

Andriy Miranskyy (Toronto Metropolitan University, CA)

License  Creative Commons BY 4.0 International license
© Andriy Miranskyy

When it comes to applying software engineering principles to programs for quantum computers, the path forward is well-defined, with a clear vision for the next five to ten years. In contrast, using quantum computers to address software engineering problems presents a more uncertain and complex trajectory. This talk explores the challenges we face in this area and highlight some open questions, offering insights that may be of interest to the broader community.

3.9 Quantum Software Consortium: Exploring Distributed Quantum Software Solutions

Hausi A. Müller (University of Victoria, CA)

License  Creative Commons BY 4.0 International license
© Hausi A. Müller
URL <https://www.qscc.ca/>

The Canadian Quantum Software Consortium (QSC) research team, with funding from a five-year NSERC Alliance Quantum Consortium Grant, explores distributed quantum solutions for Canada.

The QSC team will pioneer distributed quantum computing software methodologies by adapting known quantum algorithms, developing new distributed quantum algorithms, investigating quantum software engineering methods, and establishing communication programming abstractions to connect disparate quantum processors. The team will also develop software for several socially and economically relevant distributed quantum applications, including practical chemistry, drug discovery, quantum materials, and quantum machine learning.

Distributed quantum computing (DQC) is a rapidly evolving field with its own unique challenges and opportunities. DQC represents a significant step in evolving hybrid quantum-classical computing. While hybrid systems tightly couple classical and quantum processors, DQC systems provide classical and quantum network services to connect and entangle logical qubits across quantum chips. DQC offers ways to build scalable platforms beyond current limitations for exploring quantum architectures, networks, algorithms, software, and applications.

Experimental platforms for distributed quantum computing are critical in demonstrating quantum utility at scale. We are developing a software framework to automate the deployment of DQC experiments, thereby accelerating the proliferation of distributed quantum algorithms and solutions.

3.10 Classiq: Electronic design automation synthesis methods for quantum software

Yehuda Naveh (Classiq Technologies – Tel Aviv, IL)

License © Creative Commons BY 4.0 International license

© Yehuda Naveh

Main reference Tomer Goldfriend, Israel Reichental, Amir Naveh, Lior Gazit, Nadav Yoran, Ravid Alon, Shmuel Ur, Shahak Lahav, Eyal Cornfeld, Avi Elazari, Peleg Emanuel, Dor Harpaz, Tal Michaeli, Nati Erez, Lior Preminger, Roman Shapira, Erik Michael Garcell, Or Samimi, Sara Kisch, Gil Hallel, Gilad Kishony, Vincent van Wingerden, Nathaniel A. Rosenbloom, Ori Opher, Matan Vax, Ariel Smoler, Tamuz Danzig, Eden Schirman, Guy Sella, Ron Cohen, Roi Garfunkel, Tali Cohn, Hanan Rosemarin, Ron Hass, Klem Jankiewicz, Karam Gharra, Ori Roth, Barak Azar, Shahaf Asban, Natalia Linkov, Dror Segman, Ohad Sahar, Niv Davidson, Nir Minerbi, Yehuda Naveh: “Design and synthesis of scalable quantum programs”, 2025.

URL <https://arxiv.org/abs/2412.07372>

I will claim that the right way to create complex, scalable quantum programs is through electronic design automation (EDA) synthesis methods. I will present Classiq’s mature synthesis engine and related aspects. I will show results backing this claim, in particular orders of magnitude improvements in program parameters compared to common compilation techniques for quantum software. The talk is based on <https://arxiv.org/abs/2412.07372>

3.11 A software product line approach for developing hybrid quantum-classical software

Ricardo Pérez-Castillo (University of Castilla-La Mancha, ES)

License © Creative Commons BY 4.0 International license

© Ricardo Pérez-Castillo

Joint work of Samuel Sepúlveda, Ricardo Pérez-Castillo, Mario Piattini

Main reference Samuel Sepúlveda, Ricardo Pérez-Castillo, Mario Piattini: “A software product line approach for developing hybrid software systems”, *Inf. Softw. Technol.*, Vol. 178, p. 107625, 2025.

URL <https://doi.org/10.1016/J.INFSOF.2024.107625>

This talk introduces a Software Product Line (SPL) approach for developing hybrid quantum-classical software systems, addressing the challenges of variability, complexity, and reuse in Quantum Software Engineering (QSE). As quantum computing (QC) increasingly integrates with classical software, systematic engineering techniques are essential to manage this hybrid paradigm. The proposed approach leverages feature models (FMs) to structure software variability, enabling modular, reusable components that adapt to evolving quantum and classical computing landscapes. The methodology follows a multi-level abstraction process, encompassing computing-independent, platform-independent, and platform-specific models to facilitate domain-driven and technology-agnostic software development. The preliminary findings highlight SPLs as a powerful mechanism for automating hybrid software development, enhancing reuse, improving testing efficiency, and managing scalability. However, challenges remain in tool support, runtime variability (Dynamic SPLs), and quantum-software-hardware co-design. This approach establishes the foundation for Quantum Software Product Lines (QSPLs), providing a structured framework to advance QSE and address the complexities of hybrid quantum applications.

3.12 Generation of Fixed Margin Binary Matrices using Quantum Annealing

Rui Abreu (University of Porto, PT)

License © Creative Commons BY 4.0 International license
© Rui Abreu

Joint work of Rui Abreu, Alexandre Bergerault, Daniel Fortunato

Main reference Alexandre Bergerault, Daniel Fortunato, Rui Abreu: “Generation of Fixed Margin Binary Matrices Using Quantum Annealing”, in Proc. of the IEEE International Conference on Quantum Computing and Engineering, QCE 2024, Montreal, QC, Canada, September 15-20, 2024, pp. 193–198, IEEE, 2024.

URL <https://doi.org/10.1109/QCE60285.2024.10277>

Fixed-margin binary matrices widely used in fields like ecology and software engineering serve as essential tools for tasks such as hypothesis testing and fault localization. Generating such matrices with fixed row and column sums is critical for null model analysis but computationally challenging. This talk presents a novel approach using quantum annealing formulating the problem as a Quadratic Unconstrained Binary Optimization (QUBO) task. We share experimental results from the D-Wave Advantage quantum computer and Fixstars Amplify Annealing Engine showcasing the potential of quantum technologies to solve these problems efficiently.

This talk is supported by the [Bergerault].

References

- 1 Bergerault, Alexandre, Daniel Fortunato, and Rui Abreu. “Generation of Fixed Margin Binary Matrices Using Quantum Annealing.” In 2024 IEEE International Conference on Quantum Computing and Engineering (QCE), vol. 2, pp. 193-198. IEEE, 2024.

3.13 Towards Reliable Quantum Computing Architecture

Shinobu Saito (NTT – Tokyo, JP)

License © Creative Commons BY 4.0 International license
© Shinobu Saito

While quantum computers have attracted much attention, dealing with computational errors due to noise caused by the external environment is a significant challenge. The current state of quantum computers is unstable, therefore. Users remain concerned about incorporating them into their businesses. We propose a new system architecture for reliable quantum computing. It has two features: redundancy and self-adaptation. By integrating classical software engineering techniques (i.e., software redundancy and self-adaptive techniques), the proposed architecture contributes to improving the reliability of the output of the quantum software systems. It consists of three layers: 1) Quantum computing application, 2) MAPE-QK (Monitor, Analysis, Plan, Execute, and Quantum Knowledge), and 3) NVQS (N-version Quantum Software Systems). While MAPE-QK is a quantum software managing system, NVQS is a managed quantum software system. In NVQS, multiple quantum systems that perform the same calculation are run independently to improve the reliability of the quantum system output. On the other hand, the MAPE-QK system monitors and analyzes the execution status of multiple quantum systems. Then, the quantum systems are automatically reconfigured to maintain and improve reliability. Through this architecture, we aim to create a society where users can operate quantum applications safely and securely.

3.14 Towards View-Based Development of Quantum Software

Ina Schaefer (KIT – Karlsruher Institut für Technologie, DE), Wolfgang Mauerer (Ostbayerische Technische Hochschule – Regensburg, DE)

License © Creative Commons BY 4.0 International license
 © Ina Schaefer and Wolfgang Mauerer
Joint work of Ina Schaefer, Wolfgang Mauerer, Joshua Ammermann
Main reference Joshua Ammermann, Wolfgang Mauerer, Ina Schaefer: “Towards View-based Development of Quantum Software”, in Proc. of the 54. Jahrestagung der Gesellschaft für Informatik, INFORMATIK 2024 – Lock in or log out? Wie digitale Souveränität gelingt, Wiesbaden, Germany, September 24-26, 2024, LNI, Vol. P-352, pp. 551–554, Gesellschaft für Informatik, Bonn, 2024.
URL https://doi.org/10.18420/INF2024_43

Quantum computing is an interdisciplinary field that relies on the expertise of many different stakeholders. The views of various stakeholders on the subject of quantum computing may differ, thereby complicating communication. To address this, we propose a view-based quantum development approach based on a Single Underlying Model (SUM) and a supporting quantum Integrated Development Environment (IDE). Realizing this View-based Quantum Development approach with a supporting IDE will require solving the following research challenges: The conceptual challenge is the design of the Q-SUM that manages the consistency of different views of quantum software based on existing view- and model-based technologies. To design such a model, it is necessary to identify and analyze the various possible views, and their respective benefits and drawbacks. The design has to be extensible to integrate existing and future QPLs and support design patterns, best practices, modularity, and compositionality. Implementation challenges are the design and implementation of the IDE using a selection of state-of-the-art tooling, and the integration of external quantum simulators and interfaces to quantum hardware. For the evaluation of the Q-SUM, use case scenarios have to be defined and the capabilities of the approach have to be evaluated against them. Finally, a user study with domain experts should be conducted to rate the usability of our approach and its supporting IDE.

3.15 On HPCQC Software Stacks Needs

Laura Schulz (LRZ – München, DE)

License © Creative Commons BY 4.0 International license
 © Laura Schulz

From the last few years running a QC department in an HPC center, we’ve developed thoughts on what roles an HPCQC software stack needs to have for users, developers, administrators, facility team and the systems themselves. This would be an overview talk based on experience that is driving some of the requirements we’re baking into our Munich Quantum Software Stack. I’d also posed some open questions based on data we’re getting from the systems that may influence design decisions for the software.

3.16 Scalable approaches to constrained combinatorial optimization

Ulrike Stege (University of Victoria, CA)

License  Creative Commons BY 4.0 International license
© Ulrike Stege

Our group is developing hybrid quantum-classical approaches for solving constrained combinatorial optimization problems for gate-based quantum computing hardware – using the quantum approximate approximation algorithm (QAOA), or quantum annealing. Both approaches take as input format quadratic unconstrained binary optimization (QUBO) problems. To describe a constrained combinatorial optimization problems as QUBO, one typically introduces penalty terms to the objective function. While this introduces infeasible solution to the solution space, it maintains optimality. However, when interested in not only optimal solutions but high quality solutions, setting the penalties for the terms is a hard problem. Tackle this issues using two approaches: (1) by transforming the constrained problem to an unconstrained one while maintaining the solution quality and (2) by a systematic investigation of the solution landscape wrt different encodings of the QUBO (ie, one-hot and domain-wall encodings).

We also survey our tools in optimization (QPLEX), quantum machine learning (piQture) and in the quantum computing education space, including QWalkViz, QGrover, QNotation and QuantumCrypto.

3.17 Design Automation for Quantum Computing

Robert Wille (TU München, DE)

License  Creative Commons BY 4.0 International license
© Robert Wille

With physical realizations of quantum computing becoming accessible to a broader audience and several potential applications on the horizon, the efficient design of quantum computing solutions is now a key focus. As with classical systems, software plays a crucial role. But can we simply repurpose established software from the classical realm, or must we start from scratch for quantum computing? Additionally, how do we build bridges between computer scientists, who can develop efficient tools, and platform providers, who possess deep technological knowledge? This talk aims to provide answers to these questions, illustrated by current developments from the Munich Quantum Toolkit (<https://www.cda.cit.tum.de/research/quantum/>).

3.18 Multi-Objective Optimization for Quantum Programs

Manuel Wimmer (Johannes Kepler Universität Linz, AT)

License  Creative Commons BY 4.0 International license
© Manuel Wimmer

Joint work of Felix Gemeinhardt, Stefan Klikovits, Manuel Wimmer

Today, the processing of quantum information is mostly defined by quantum circuits. However, the design of such quantum circuits is a challenging task which requires significant knowledge in quantum information theory. On top of that, finding an accurate solution (correct

functionality) with low computational cost requires to reason about trade-offs between conflicting goals. This talk summarizes an ongoing series of works [1, 2] we perform to provide dedicated automation support for the synthesis, optimization, and debugging of quantum operators and circuits by using multi-objective optimization. More specifically, we use a meta-heuristic search approach by utilizing genetic programming in combination with numerical parameter optimizers as well as techniques from quantum information theory. The evaluation of the current framework on a set of quantum programs collected from literature and open-source projects shows that the approach is able to synthesize quantum operators as well as to correct and optimize existing quantum circuits.

References

- 1 Felix Gemeinhardt, Stefan Klikovits, Manuel Wimmer: GeQuPI: Quantum Program Improvement with Multi-Objective Genetic Programming. *J. Syst. Softw.* 219: 112223 (2025)
- 2 Felix Günther Gemeinhardt, Stefan Klikovits, Manuel Wimmer: Hybrid Multi-Objective Genetic Programming for Parameterized Quantum Operator Discovery. *GECCO Companion* 2023: 795-798

4 Working groups

4.1 Report of the working group on “Quality assurance for quantum programs”

Paolo Arcaini (National Institute of Informatics – Tokyo, JP), Shaukat Ali (Simula Research Laboratory – Oslo, NO), José Campos (University of Porto, PT), Ignacio García Rodríguez de Guzmán (University of Castilla-La Mancha, ES), Andriy Miransky (Toronto Metropolitan University, CA), Anila Mjeda (Munster Technological University – Cork, IE), and Lei Zhang (University of Maryland – Baltimore County, US)

License © Creative Commons BY 4.0 International license
 © Paolo Arcaini, Shaukat Ali, José Campos, Ignacio García Rodríguez de Guzmán, Andriy Miransky, Anila Mjeda, and Lei Zhang

The discussion of the group was related to quality assurance of quantum programs [1]. In order to understand how to perform quality assurance for quantum programs, we discussed how the different phases of the software life-cycle are affected:

■ Requirements

- In terms of functional requirements, requirements engineering is not too different from that for classical programs;
- Instead, for non-functional requirements, new approaches are needed. For example, the deterioration of the operation of a machine is a problem specific to quantum computing; this requires a specific non-functional requirement that describes how and when recalibration should be performed. Similarly, the noise is a characteristic specific to each quantum computer that must be considered when specifying requirements.
- Security and privacy concerns may be specific to quantum computing. For example, specific malicious quantum circuits may be developed, and techniques must be developed to detect these security threats.

■ Design

- Designing quantum programs requires new types of abstractions. In this way, more stakeholders can reason on more abstract versions of the program (e.g., model-based testing). Moreover, we need proper links between different levels of abstraction.

- We need new techniques to design *hybrid systems* that combine classical and quantum programs. Such techniques should be able to properly define the boundaries between the classical and the quantum parts.
- Identify and create new design and architectural patterns. It will be necessary to address different development strategies for hybrid systems according to the functional and non-functional requirements and characteristics of quantum infrastructure.
- Also, the definition of assertions and pre-/post-conditions may require new formalisms.
- **Implementation**
 - As for the design languages, we also need new programming paradigms that would allow us to specify the program at higher levels of abstraction than what is currently supported by existing languages. Such abstractions should allow one to write a quantum program without having a deep knowledge of quantum computing.
 - The programming languages should be able to support the integration between classical and quantum code.
- **Validation and Verification**
 - As for classical problems, testing will be one of the main assessment approaches. However, due to the nature of quantum computing, completely new testing approaches need to be defined:
 - * New coverage criteria must be defined for quantum programs. While black-box coverage criteria seem easier to define, the definition of white-box coverage criteria seems more difficult.
 - * We need new ways to specify test inputs. For example, we need ways to specify inputs in superposition.
 - * The oracle problem is very challenging. First, complete oracles may not be available. Moreover, the presence of noise makes it difficult to assess whether the deviation from the expected output is due to a fault or to noise.
- **Operation**
 - Execution of a quantum program consists of running the program multiple times. So, it will be particularly important to decide how to map the workload across machines (for cost, noise, etc. concerns) and maximize the utilisation of QPUs (e.g., running multiple circuits in parallel).
- **Maintenance/Evolution**
 - Programming environments like Qiskit change frequently. It will be necessary to be able to adapt to the new APIs effectively.

References

- 1 Murillo, J.M., Garcia-Alonso, J., Moguel, E., Barzen, J., Leymann, F., Ali, S., Yue, T., Arcaini, P., Pérez-Castillo, R., García Rodríguez de Guzmán, I., Piattini, M., Ruiz-Cortés, A., Brogi, A., Zhao, J., Miranskyy, A., Wimmer, M.: Quantum software engineering: Roadmap and challenges ahead. *ACM Trans. Softw. Eng. Methodol.* (Jan 2025). <https://doi.org/10.1145/3712002>

4.2 Adaptive Hybrid, Distributed Quantum Systems

Antonio Brogi (University of Pisa, IT), Schahram Dustdar (TU Wien, AT), Michael Felderer (DLR – Köln, DE), Hausi A. Müller (University of Victoria, CA), Juan Manuel Murillo (University of Extremadura – Cáceres, ES), Ricardo Pérez-Castillo (University of Castilla-La Mancha, ES), Antonio Ruiz Cortés (University of Sevilla, ES), Shinobu Saito (NTT – Tokyo, JP), and Ulrike Stege (University of Victoria, CA)

License © Creative Commons BY 4.0 International license

© Antonio Brogi, Schahram Dustdar, Michael Felderer, Hausi A. Müller, Juan Manuel Murillo, Ricardo Pérez-Castillo, Antonio Ruiz Cortés, Shinobu Saito, and Ulrike Stege

The working group discussed the different types of adaptation [1] that hybrid [2], distributed quantum systems [3] should feature and classified them into three main dimensions: cost-driven, resource-driven, and quality-driven.

The working group highlighted the important need to trade off the aforementioned three dimensions (cost, resources, and quality) in order to realize valuable adaptations of quantum systems.

The working group then focused on identifying the new specific parameters to be taken into account for adapting quantum systems, such as circuit depth and height, coherence, entanglement, measurement, and noise.

The working group discussed two possible use cases for adaptive quantum systems: the travelling salesperson problem and the shot-wise distribution of quantum computations.

Hybrid quantum-classical and distributed quantum computing require dynamic adaptation due to evolving hardware, quantum networks, runtime system software, and problem constraints. Continuous monitoring and automated adaptation are essential for the long-term evolution of hybrid quantum systems. One promising avenue is to develop model-based frameworks [4] to explore, evaluate, and track different design choices systematically using autonomic computing techniques.[5]

References

- 1 de Lemos, R., Giese, H., Müller, H.A., Shaw, M.: Software Engineering for Self-Adaptive Systems II, Springer Lecture Notes in Computer Science LNCS 7475 (2013) <https://doi.org/10.1007/978-3-642-35813-5.pdf> <https://link.springer.com/content/pdf/10.1007/978-3-642-35813-5.pdf>
- 2 Endo, S., Cai, Z., Benjamin, S.C., Yuan, X.: Hybrid quantum-classical algorithms and quantum error mitigation. J. Phys. Soc. Jpn. 90: 032001 (Mar 2021). <https://doi.org/10.7566/JPSJ.90.032001>
- 3 Caleffi, M., Amoretti, M., Ferrari, D., Illiano, J., Manzalini, A., Cacciapuoti, A.S.: Distributed quantum computing: A survey. Computer Networks 254: 110672 (Dec 2024). <https://doi.org/10.1016/j.comnet.2024.110672>
- 4 Ishikawa, F., Saito, S.: Model-based framework for continuous adaptation and evolution of quantum-classical hybrid systems. Proceedings 8th International Conference on the Art, Science, and Engineering of Programming, pp. 118–125 (Jul 2024). <https://doi.org/10.1145/3660829.36608>
- 5 IBM: An architectural blueprint for autonomic computing. Autonomic Computing. White Paper. Fourth Edition. (Jun 2006). <https://www.scribd.com/document/507031803/10-1-1-150-1011>

4.3 Software Abstraction in Quantum Computing

Sebastian Feld (Delft University of Technology, NL), Michael Falkental (Kipu Quantum – Karlsruhe, DE), Michael Felderer (DLR – Köln, DE), José Manuel García Alonso (University of Extremadura – Cáceres, ES), Wolfgang Mauerer (Ostbayerische Technische Hochschule – Regensburg, DE), Yehuda Naveh (Classiq Technologies – Tel Aviv, IL), Ina Schaefer (KIT – Karlsruher Institut für Technologie, DE), Laura Schulz (LRZ – München, DE), Robert Wille (TU München, DE), and Manuel Wimmer (Johannes Kepler Universität Linz, AT)

License  Creative Commons BY 4.0 International license

© Sebastian Feld, Michael Falkental, Michael Felderer, José Manuel García Alonso, Wolfgang Mauerer, Yehuda Naveh, Ina Schaefer, Laura Schulz, Robert Wille, and Manuel Wimmer

Quantum computing is promised to redefine a variety of application domains. However, its practical realization remains hindered by the complex interplay between hardware limitations and software intricacies. At the Dagstuhl Seminar, our working group concentrated on the question how software abstraction can effectively bridge this gap and make quantum computing both accessible and scalable. Building upon the principles of Model-Driven Architecture (MDA) [1], we discussed methods to structure quantum software development into distinct layers that align high-level requirements with hardware-specific constraints.

In our discussions, we focused on a three-tiered abstraction approach. At the highest level, the computation-independent model (CIM) is used to capture essential requirements and business objectives without engaging with technical details. This model is then refined into a platform-independent model (PIM) that defines core functionalities, which can be applied uniformly across different quantum architectures. Finally, the platform-specific model (PSM) adapts these functionalities to the particular characteristics and limitations of individual quantum hardware systems. We see this layered approach as a practical method to manage complexity and to support a gradual evolution of quantum systems in line with hardware advances.

Another central topic was the potential use of domain-specific languages (DSLs) [2] for quantum computing. We discussed how DSLs could provide an intuitive syntax and semantic framework tailored to quantum operations, offering more expressive tools than conventional programming languages. However, we also recognized key challenges, such as ensuring long-term maintainability, achieving compatibility with existing systems, and adapting to rapid technological progress.

Furthermore, our stakeholder analysis revealed that different user groups – ranging from quantum researchers and algorithm developers to business application engineers – have distinct requirements. Researchers demand high-fidelity abstractions to model quantum phenomena accurately, while industry practitioners need interfaces that integrate seamlessly with traditional computing environments. We therefore recommend a stakeholder-centric approach, supported by iterative feedback through user studies and workshops.

References

- 1 Kleppe, A., Warmer, J. & Bast, W. MDA explained: the model driven architecture: practice and promise. (Addison-Wesley Professional, 2003)
- 2 Mernik, M., Heering, J. & Sloane, A. When and how to develop domain-specific languages. *ACM Computing Surveys (CSUR)*. **37**, 316-344 (2005)

5 Panel discussions

5.1 A Quantum Computing Song: Qubit Entanglement

Hausi A. Müller (University of Victoria, CA)

License © Creative Commons BY 4.0 International license
© Hausi A. Müller

Joint work of The Dagstuhl Quantum Software Engineering Singers

Nightly sing-alongs inspired the arrangement of the “Qubit Entanglement” song in the renowned Dagstuhl music room. In this Quantum Software Engineering seminar, the attendees’ daytime momentum, commitment, and enthusiasm during the technical and scientific discussion sessions continued well into the evening discussions and singing sessions. In this stimulating atmosphere, the “Qubit Entanglement” song emerged naturally; thus, its lyrics are included in this report. The lyrics are arranged for the melody of the iconic Beatles song “Eight Days a Week.”

Qubit Entanglement

To the Melody of “Eight Days A Week” by The Beatles (Lennon & McCartney)

Ooh, I need en-tangle-ment,
Guess you know it’s true.
Hope you are my qubit,
Just like I am yours.

Super me, position me,
Super me, position me.
Love you in ev’ry eigen-state,
Eight days a week.

Quantum love is hard to find,
Always on my mind.
Keeps collapsing all the time,
Love you all the time.

Super me, position me,
Super me, position me.
Love you in ev’ry eigen-state,
Eight days a week.

Eight days a week,
I lo-oo-ve you.
Eight days a week
Is not enough to CNot gate.

Ooh, I need coherence,
Guess you know it’s true.
Hope you need my quantum love,
Just like I need you. Ooh,

Super me, position me,
Super me, position me.
Love you in ev’ry eigen-state,
Eight days a week.

Eight days a week,
I lo-oo-ve you.
Eight days a week
Is not enough to CNot gate.

Quantum love is hard to find,
Always on my mind.
Keeps collapsing all the time,
Love you all the time.

Super me, position me,
Super me, position me.
Love you in ev’ry eigen-state,
Eight days a week.
Eight days a week.
Eight days a week.

Participants

- Shaukat Ali
Simula Research Laboratory – Oslo, NO
- Paolo Arcaini
National Institute of Informatics – Tokyo, JP
- Antonio Brogi
University of Pisa, IT
- José Campos
University of Porto, PT
- Schahram Dustdar
TU Wien, AT
- Michael Falkental
Kipu Quantum – Karlsruhe, DE
- Sebastian Feld
Delft University of Technology, NL
- Michael Felderer
DLR – Köln, DE
- José Manuel García Alonso
University of Extremadura – Cáceres, ES
- Ignacio García Rodríguez de Guzmán
University of Castilla-La Mancha, ES
- Kostas Magoutis
University of Crete, Heraklion, GR & FORTH-ICS, Heraklion, GR
- Wolfgang Mauerer
Ostbayerische Technische Hochschule – Regensburg, DE
- Andriy Miranskyy
Toronto Metropolitan University, CA
- Anila Mjeda
Munster Technological University – Cork, IE
- Hausi A. Müller
University of Victoria, CA
- Juan Manuel Murillo
University of Extremadura – Cáceres, ES
- Yehuda Naveh
Classiq Technologies – Tel Aviv, IL
- Ricardo Pérez-Castillo
University of Castilla-La Mancha, ES
- Antonio Ruiz Cortés
University of Sevilla, ES
- Shinobu Saito
NTT – Tokyo, JP
- Ina Schaefer
KIT – Karlsruher Institut für Technologie, DE
- Laura Schulz
LRZ – München, DE
- Ulrike Stege
University of Victoria, CA
- Robert Wille
TU München, DE
- Manuel Wimmer
Johannes Kepler Universität Linz, AT
- Lei Zhang
University of Maryland – Baltimore County, US

