Software Engineering Challenges for Quantum Computing: Report from the First Working Seminar on Quantum Software Engineering (WSQSE 22)

Michael Felderer
German Aerospace Center
(DLR), Germany
University of Cologne, Germany
University of Innsbruck, Austria
michael.felderer@dlr.de

Davide Taibi
University of Oulu, Finland
Tampere University, Finland
davide.taibi@oulu.fi

Fabio Palomba
Software Engineering (SeSa) Lab
University of Salerno, Italy
fpalomba@unisa.it

Michael Epping
German Aerospace Center
(DLR), Germany
michael.epping@dlr.de

Malte Lochau
University of Siegen, Germany
malte.lochau@uni-siegen.de

Benjamin Weder
University of Stuttgart, Germany
benjamin.weder@iaas.uni-stuttgart.de

ABSTRACT

The First Working Seminar on Quantum Software Engineering (WSQSE 2022) took place on December 15 and 16, 2022 in Innsbruck, Austria. An audience of 33 quantum computing and software engineering researchers and practitioners joined the two-day event. The workshop enabled us to foresee short-term and long-terms perspectives of Quantum Software Engineering, as well as a set of requirements, issues, and challenges for architecting, programming, and testing Quantum Software Engineering applications. In this report, we provide the summary of the workshop, by reporting on the structure of the event and the main results coming from the sessions and working groups.

1. INTRODUCTION

Quantum software enables the development of the promised revolutionary applications in many domains. To build reliable and correct quantum software, Quantum Software Engineering (QSE) methods are needed [2]. In recent years, QSE's importance in building reliable software is being realized by researchers [9].

QSE is a new emerging research area. Therefore, we believe it is crucial to identify high-level research goals and devise potential research directions driven by the QSE community.

For that purpose, we organized the First Working Seminar on Quantum Software Engineering (WSQSE 2022) on December 15 and 16, 2022 in Innsbruck, Austria.

During the workshop, 33 researchers and practitioners collaboratively worked toward the definition of QSE issues, research ideas, and a common roadmap. In particular, we investigated short-term vs. long-term perspectives of QSE, architectures for quantum applications, testing, and programming QSE systems.

In this paper, we present the result of our workshop, describing the outcomes of the working sessions and providing future directions for QSE research.

The rest of the paper is organized as follows: Section 2 briefly describes the structure of the working seminar, including a short summary of the presented lightning talks. The next four sections summarize the performed working sessions. Section 3 summarizes the outcome of the short-term/long-term perspectives of QSE. Section 4 presents the results of the QSE application architecture session. Section 5 presents the quantum programming session, and Section 6 describes the issues and requirements identified in the quantum software testing session. Finally, Section 7 draws conclusions and provides an outlook.

2. STRUCTURE OF THE WORKING SEMINAR

The workshop gathered participants from different backgrounds, including physics, computer science, and chemistry. In total, 33 quantum computing and software engineering researchers and practitioners participated in the workshop. The participants mainly came from European countries such as Austria (13 participants), Germany (7), Finland (4), Italy (2), Spain (2), Netherlands (2), Norway (1), and Portugal (1). We also had a participant from Israel. WSQSE was designed to have a strong interconnection between academia and industry, as proved by the participation of six companies involved in the development of quantum technologies.

The seminar was composed of a set of lightning talks, and four working sessions.

The overall program is available at `https://sites.google.com/view/wsqse-2022/schedule?authuser=0`

2.1 Lightning Talks

After a short welcome from the workshop co-chairs, the first session included a total amount of 15 ten-minute lightning talks from the workshop participants. The talks focused on multiple themes connected to quantum software engineering. The session had the primary goal to let participants know each other and be exposed them to the various areas of expertise covered within the workshop. More specifically, the session kicked off with a talk related to quantum theories. Afterward, the subsequent lightning talks focused on four main themes. At first, we included talks connected to *quantum programming*, which overviewed the lessons learned over the first decades of quantum programming, other than addressing specific challenges such as hybrid software modernization, the resolution of the Min-k-Union problem with quantum computing, how to program quantum annealers, and parity quantum computing [8]. In the second place, we moved toward *modeling and architecting quantum programs*. In this respect, we included talks focusing on the concept of quantum pictorialism [3] and quantum service-oriented computing. Then, we focused on the *processes* required to effectively build quantum programs, by treating aspects connected to software engineering lifecycle, processes, and conformance and mutation testing for quantum programs. Last but not least, we included three lightning talks on *transfer and application* with the aim of touching base with the advantages of quantum computing for early adopters, the open challenges faced by developers when employing quantum technologies, and emerging frameworks such as QuBRA[1] and ProvideQ.[2]

2.2 Working Sessions

Based on the preferences specified in a questionnaire we distributed before the workshop, we proposed 5 main session topics. Participants then voted on their preferred topics, identifying three main topics for the following working sessions.

The four selected sessions focused on short-term vs. long-term perspectives of quantum software engineering, architectures for quantum applications, programming, and testing.

3. SHORT-TERM VS. LONG-TERM PERSPECTIVES OF QSE

In this session, we discussed how to prioritize the various research challenges for software engineering in the upcoming era of quantum computation.

One key insight throughout the session is that quantum computation does not constitute a new programming paradigm in the classical sense, but, first of all, proclaims a new computational architecture together with novel computational thinking. In this regard, although quantum computing will obviously influence software engineering as a whole in the near future, there are many successful and established concepts, tools, and techniques in today's software engineering which will not all suddenly become inappropriate and obsolete in the presence of quantum computing.

Instead, many software engineering principles and practices like separation of concerns and encapsulation as we know them today are particularly intended to be agnostic to (existing and not-yet-existing) computational platforms thus mostly requiring careful adjustments to cope with major characteristics of quantum software.

In addition, quantum software development as we experience it today mostly happens at the source code level and reaches downwards to the assembly level. Hence, short-term engineering support is desperately needed at those lower levels, whereas higher levels of abstraction are not yet within reach.

To gain a better understanding of the short-term challenges of quantum software engineering, we characterized three paradigmatic use cases exemplifying different aspects of quantum computation applications today.

As a result, we identified three short-term research challenges for quantum software engineering:

- Making quantum computing accessible to users and developers
- Providing benchmarks for demonstrating the quantum advantage
- Facilitating general-purpose computing for hybrid quantum systems based on generic specifications of problem descriptions and platform constraints.

4. ARCHITECTURES FOR QUANTUM APPLICATIONS

In this session, we discussed multiple issues and approaches when developing and operating complex quantum applications.

First, quantum applications that go beyond small prototypes are typically hybrid, comprising multiple quantum and classical programs. Following the concept of service-oriented computing, these quantum and classical programs should be provided as services, as exemplarily shown by the first approaches motivating the need for quantum service-oriented computing [6, 7]. Thus, different benefits can be achieved when using these services to develop quantum applications, such as modularity, reuse of well-tested components, or reduced development time.

However, one identified challenge is the granularity of these services, e.g., if they should comprise whole quantum algorithms with all pre- and post-processing steps or if single functionalities, such as mitigating errors, should be provided by small microservices. Furthermore, the various services must be orchestrated, i.e., the control and data flow between them must be ensured.

For this, different approaches were discussed, reaching from local files to sophisticated solutions, such as using workflows to benefit from their robustness and scalability [11].

Thereby, we emphasized the need to enable an easy exchange of single execution steps, e.g., if an improved quantum computer is available. The second part of the session focused on increasing the reproducibility, understandability, and quality of quantum applications. One key insight was that this requires two major approaches:

- Sophisticated versioning for all components involved in the development and operation process of the quantum application. This includes, e.g., the used quantum computer with its characteristics or the utilized compiler.
- Monitoring and data collection throughout the whole application lifecycle. Thus, occurring errors or possible improvements of the quantum application can be identified using this data.

Finally, in the last part of the session, we discussed the need for a playground or sandbox to enable the reproducibility of results by simulating the quantum computer behavior. Thus, a simulator with a very sophisticated error model is required to provide a controllable testing and execution environment. To push this issue forward, it is planned to identify the metrics and characteristics that should be collected from quantum computers and to get in touch with hardware providers, such as AQT and IBM, to evaluate the possibility of extracting this data.

[1] The QuBRA project: `https://rb.gy/tyqpyf`.

[2] The ProvideQ project: `https://www.provideq.org`.

5. QUANTUM PROGRAMMING

The session on Quantum Programming had only very few participants and thus allowed us to get to know each other more. Consequently, the discussions were determined by the participants' specific backgrounds and interests.

We started by collecting our experience with frameworks and what we currently use.

As for Quantum Programming frameworks, not surprisingly, we all are familiar with qiskit [1]. Another python-based tool is QuTiP [5].

We quickly moved on to discuss the peculiarities of qudits (i.e. using higher-dimensional systems instead of qubits). Our discussions were inspired by experiments carried out in Innsbruck [10]. While we did not see a fundamental theoretical advantage of the system, we imagine practical advantages e.g. w.r.t. noise and possible simplifications when the quantum algorithm is well-suited for qudits, because it is formulated in a way that includes many small integer variables. The amount of available tools that specifically target qudits is currently very limited to the best of our knowledge. For example, a compiler that automatically decides when to use qudits would be beneficial. We continued to discuss other compiler needs, e.g. hybrid compilation which includes the classical and the quantum part and allows for optimizations that are not restricted to only one of the realms [4].

6. QUANTUM SOFTWARE TESTING

The session on testing showed that there are plenty of open questions and interesting research directions to explore in this area.

We started out by discussing performance testing. To support the development process, a tool for automatic test generation would be very helpful. However, to the best of our knowledge, it is not available, yet. We discussed the added difficulty of performance testing due to the inherent randomness of quantum algorithms. A problem that is also present in classical non-deterministic algorithms, of course. However, due to the low technological readiness level (TLR) of quantum computers, we had vivid discussions on how to appropriately measure the performance of algorithms and devices. For some of the participants, time to solution is not yet the best figure of merit. But others stated that this is the quantity that customers are interested in at the end of the day and they will frequently request comparisons in this regard. We continued to discuss the necessity of standards in this context, e.g. on how to measure the performance of quantum algorithms. Research on these topics is still in an early stage, but a common language and well-defined terms would be useful. We did not conclude here.

We moved on to the topic of automated testing to cover more of the broad field of "testing" and also let other participants contribute their current interests. One possible direction of research is to define appropriate fitness functions in the context of quantum algorithms. The goal is to identify good test cases from the large space of possible tests. These tests can be generated via combinatorial testing, where a number of possible inputs to the system under test are combined in a meaningful way. In our lively discussion, we also discussed on test coverage, which requires a white box approach, while in many cases we are interested in a black box approach to the quantum algorithms.

Similarly to the discussion we had about performance testing, we talked about the implications of the stochastic nature of quantum algorithms. There are many interesting questions involved here, e.g. when to accept a result in the presence of noise? How many shots are required? For the latter question, we concluded that it can only be answered on a use-case basis. We noted that it is possible to simulate noise on common platforms. However, the quality of the noise model, i.e. the validity of using it for a specific purpose, is not obvious, as well as choosing meaningful noise parameters is non-trivial in practice. This led us to discuss features for quantum computers, which would be nice to have. A discussion that was also stirred by our visit to Alpine Quantum Technologies (AQT) and which is not only related to testing, but it is worth mentioning here and we are planning to follow up on it.

Two requirements we identified in our discussion are:

- To understand how long after the calibration a computer is still “good to use”
- To get the environment parameters of the quantum computer in case of a failed test. These parameters could include temperature, power consumption, and many other quantities.

Hardware manufacturers can also benefit from this information when we will be able to identify correlations between test failure and parameters whose influences are currently insufficiently understood. This again highlights how expertise in many different disciplines is required to make quantum computing possible, a great challenge that often became obvious during our interdisciplinary workshop.

7. CONCLUSION

In this work, we report on the outcomes of the First Working Seminar on Quantum Software Engineering (WSQSE 22).

During the workshop, 33 researchers and practitioners met in Innsbruck (Austria) to discuss challenges, and shared research directions in the field of Quantum Software Engineering (QSE).

The workshop enabled us to distill QSE issues, research ideas, and a common research roadmap.

One of the major challenges that emerged during the workshop was the difference in background and in the terminology adopted by different persons working in different backgrounds. This issue was then also reflected in open issues in the development of QSE application. As an example, QSE developers with a background in physics usually lack software engineering, and in particular software process and quality practices, that might bring integration issues of QSE in software systems. Another issue is due to the complexity of the quantum algorithms and programming that is usually not accessible to all developers, because of the complexity itself.

We are planning to again organize a working seminar on quantum software engineering in the future to get an update on the status of QSE issues and update our roadmap, and last but not least, to establish a QSE community.

Acknowledgements

We thank all participants of the working seminar for their active contribution.

8. REFERENCES

[1] Anis, M. S., Abraham, H., AduOffei, R. A., Agliardi, G., Aharoni, M., Akhalwaya, I. Y., Aleksandrowicz, G., Alexander, T., Amy, M., Anagolum, S., et al. Qiskit: An open-source framework for quantum computing. *Qiskit/qiskit* (2021).

[2] Bertels, K., Sarkar, A., Hubregtsen, T., Serrao, M., Mouedenne, A., Yadav, A., Krol, A., and Ashraf, I. Quantum computer architecture: Towards full-stack quantum accelerators. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)* (2020), IEEE.

[3] Coecke, B. Quantum picturalism. *Contemporary physics 51*, 1 (2010), 59–83.

[4] Epping, M. Hybrid simplification rules for boundaries of quantum circuits, 2022.

[5] Johansson, J., Nation, P., and Nori, F. Qutip 2: A python framework for the dynamics of open quantum systems. *Computer Physics Communications 184*, 4 (2013), 1234–1240.

[6] Kumara, I., Heuvel, W.-J. V. D., and Tamburri, D. A. Qsoc: Quantum service-oriented computing. In *Symposium and Summer School on Service-Oriented Computing* (2021), Springer, pp. 52–63.

[7] Moguel, E., Rojo, J., Valencia, D., Berrocal, J., Garcia-Alonso, J., and Murillo, J. M. QSOC: Quantum Service-Oriented Computing. *Software Quality Journal* (2022), 1–20.

[8] Paul, G., Chattopadhyay, A., and Chandak, C. Designing parity preserving reversible circuits. In *International Conference on Reversible Computation* (2017), Springer, pp. 77–89.

[9] Piattini, M., Peterssen, G., and Pérez-Castillo, R. Quantum computing: A new software engineering golden age. *SIGSOFT Softw. Eng. Notes 45*, 3 (July 2020), 12–14.

[10] Ringbauer, M., Meth, M., Postler, L., Stricker, R., Blatt, R., Schindler, P., and Monz, T. A universal qudit quantum processor with trapped ions. *Nature Physics 18*, 9 (jul 2022), 1053–1057.

[11] Weder, B., Breitenbücher, U., Leymann, F., and Wild, K. Integrating Quantum Computing into Workflow Modeling and Execution. In *Proceedings of the 13th IEEE/ACM International Conference on Utility and Cloud Computing (UCC)* (2020), IEEE, pp. 279–291.