Quantum Software Engineering: A New Genre of Computing

Muhammad Azeem Akbar
azeem.akbar@lut.fi
Department of Software Engineering, LUT University
Lappeenranta, Finland

Arif Ali Khan
arif.khan@oulu.fi
M3S Empirical Software Engineering Research Unit,
University of Oulu
Oulu, Finland

Sajjad Mahmood
smahmood@kfupm.edu.sa
Information and Computer Science Department, King Fahd University of Petroleum and Minerals
Dhahran, Saudi Arabia

Saima Rafi
s.rafi@napier.ac.uk
School of Computing, Engineering the Built Environment,
Edinburgh Napier University
EH10 5DT, Scotland

Abstract

The quantum computing (QC) field is rapidly moving beyond the realm of pure science to become a commercially viable technology that may be able to overcome the drawbacks of traditional computing. Major technology tycoons have spent in building coding frameworks and hardware to create applications specifically designed for quantum computing over the last few years. The development of QC hardware is accelerating, however, the requirement for software-intensive methodology, approaches, procedures, instruments, roles and responsibilities for creating industrial-focused quantum software applications arises from operationalizing the QC. This paper outlines the concept of quantum software engineering (QSE) life cycle, which entails the engineering of quantum requirements, design, implementation, testing and maintenance of quantum software. This paper notably advocates for collaborative efforts between the industrial community and software engineering research to propose practical solutions to support the complete activities for the development of quantum software. The proposed vision makes it easier for researchers and practitioners to suggest new procedures, reference designs, cutting-edge equipment, and methods for utilizing quantum computers and creating the newest and most advanced quantum software.

CCS Concepts

• **Software and its engineering → Software development process management**.

Keywords

Quantum computing (QC), Quantum software engineering (QSE), Quantum software development life cycle

ACM Reference Format:
Muhammad Azeem Akbar, Arif Ali Khan, Sajjad Mahmood, and Saima Rafi. 2024. Quantum Software Engineering: A New Genre of Computing. In *Proceedings of the 1st ACM International Workshop on Quantum Software Engineering: The Next Evolution (QSE-NE '24), July 16, 2024, Porto de Galinhas, Brazil*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3663531.3664750

QSE-NE '24, July 16, 2024, Porto de Galinhas, Brazil

ACM ISBN 979-8-4007-0673-8/24/07
https://doi.org/10.1145/3663531.3664750

1 Introduction

Quantum computing (QC) uses quantum bits(qubits) instead of the binary digits (bits) used in conventional electric computing, this allows the processing of information exponentially rapidly than it can by classical computers thanks to quantum physics properties like quantum entanglement and quantum states [16, 33]. Researchers and businesses are starting to recognize some possible uses for tenfold powerful processors than those already in use as quantum supremacy, where a quantum computer can indeed be demonstrated to process any work faster than a conventional computer, approaches reality. These applications will soon be able to raise our standard of living. It is inconceivable how machine learning enabled by quantum computers can enhance our quality of life [1].

By simulating complicated physical systems or using machine learning techniques, QC promises to handle many issues more effectively or accurately with conventional computers [2, 23]. The importance of creating suitable quantum applications and software and incorporating them into current software architectures is rising as a result of recent developments in creating more strong quantum computers [13, 31]. However, the creation of such quantum applications is challenging and necessitates the expertise of specialists in a number of disciplines, including mathematics, computer science, and physics[24].

Quantum software Engineering (QSE) is a developing field of study that looks at ideas, rules, and best practices for creating, maintaining, and evolving quantum applications [6, 24]. It seeks to enhance the reusability and quality of the resultant quantum applications by methodically employing software engineering concepts throughout every development stage, from the preliminary requirement analysis till the completion of the program [25]. Software development lifecycles (SDLC) are frequently used in traditional software engineering to record the various stages of development that a software or application artifact goes through [8, 28]. Moreover, such SDLC lists the best practices, techniques, and technologies that can be used in the different phases[10, 39, 44]. As a result, they can instruct new developers about develop process by giving a general understanding or by acting as a starting point for collaboration

with experts in other domain [3, 7].
Quantum algorithms must be implemented as software in order to create useful and real-world QC applications. Taking a cue from the world of traditional computing, creating reliable software necessitates adhering to an SDLC, which typically consists of phases for requirements elicitation, design and architecture, building, testing, deployment, and maintenance. Since the creation of quantum software is still in its infancy, there is no specific SDLC for it. The goal of this study is to define a reliable quantum SDLC and to raise awareness of the necessity of quantum software engineering in the modern day.

2 BACKGROUND

Innovative methods, procedures, tools, and strategies that specifically target creating software applications consist of quantum physics are required for quantum software developers. Fundamental quantum physics properties like entanglement and superposition make it difficult to design a quantum software algorithm. Since designing quantum software is the most crucial stage in creating QS systems, there is a tremendous need for new principles and approaches [1].

2.1 Importance of Quantum in Emerging Technologies

QC is a new field with enormous potential, particularly in optimization issues. The software technique for QC differs from conventional computing, QC also uses a distinct mechanism. A wide range of social and industrial problems that require expensive computations can be solved via QC. A few notable applications are the development of vaccines and drugs more quickly, portfolio management, financial optimization, and complicated physics simulations to better understand the world. QC success will therefore unavoidably have a substantial impact on our daily lives and transform the majority of industries across numerous disciplines. As examples, the implications of QC in emerging technologies are discussed below [11, 41, 45].
(1) QC in Smart Cities
Future fully connected and automated smart cities would be one use of QC-based Iot application. This would regulate the creation and distribution of energy, the handling of trash, the movement of people and things, lighting, and even the regulation of the atmosphere [32]. Cities are expanding along with the human species, and in order to prevent the consequences of climate change on ecological systems, the future cities will be required to manage an increasing number of people. QC might make it possible for city dwellers to retain a high standard of living despite the expectations of large local populations[29].
(2) QC in Smart Road Networks
Road accidents might be liminated, fuel consumption could be managed for efficiency, and traffic congestion could be drastically reduced with the help of smart cars and, in the coming days, self-driving automobiles interconnected to an IoT using QC principles [15, 19]. Thus, QC gives a potential solution for managing automatic driving in a vehicle in everything context.
(3) QC in Smart Air Traffic Control
Similarly, QC could be used to process air traffic control. This would result in significant increases to accuracy (and, thus safety), as well as a more manageable ATC network size [42]. This is required in future in order for crewless aircraft (drones) to replace several physical delivery jobs that are currently being ineffectively carried out by people travelling on roadways. It will make it possible to improve quality of life by providing fast, individualized delivery of new goods, medications, and even travelers to locations outside of major cities [30].
(4) QC in Smart Factories
Suppose that the automated manufacturing sector can benefit from the speed of QC. If so, factories in the future would become incredibly efficient and capable of taking over an increasing number of the menial activities currently carried out or managed by humans [21]. Economic benefits would result from this, and human labour would be free to investigate other fulfilling ways to pass the time [38].
(5) QC in Smart Power Supply and Distribution
The distribution system and energy supply will be made more efficinet with a QC-enabled power grid connected to the IoT, which should reduce human energy comsumption while preserving modern standards of living[27]. Forecasting and modeling with QC support. Quality of life advancements have already been made in the 20th century as a result of traditional computers modelling intricate human and environmental systems (such as the planet's atmosphere and stock markets) and utilizing there models and simulators to accurately anticipate the future. This is referred to as 'forecasting. These can be significantly improved by introducing QC, gathering and taking inputs across entire complex systems, processing them, and predicting how they will interact[43].
(6) QC-enabled Machine Learning
Unimaginable latest applications for systems could result from machine learning, which allows computers to learn and reconfigure themselves for higher accuracy or efficiency in accomplishing task[14]. Future quantum computers will be far more powerful, which will allow machine learning to improve quickly. This, along with modelling and forecasting that are supported by QC, has the ability to enhance medicine and eradicate numeroud diseases, deliver effective treatment, and promptly detect and treat illness (supported by the IoT)[14]. Further, as computers begin to adapt and learn themselves to ensure human aims of exixtence, compatibility with the earth's ecosystems, and opulent and effortless lives, QC-enabled machine learning will also improve our life quality in ways we cannot now envision[18].
(7) QC in 6G Technology
Quantum computing (QC) is poised to revolutionize 6G technology. Its unparalleled computational power will optimize network infrastructure, enhance data transmission efficiency, and bolster cybersecurity. QC-driven machine learning algorithms promise to revolutionize network management and predictive analytics. As QC evolves, it will unlock new frontiers in telecommunications, shaping the future of 6G and beyond[4].

2.2 Why quantum software Engineering

QC has captured the attention of the general public, scientists, and engineers over the last few decades. Quantum computers have

enormous processing capacity because they can conduct numerous parallel computations using quantum superposition that are not achievable with conventional computers[22]. By utilizing this capacity, quantum software and QC enable applications that are normally not possible with conventional computing, like speedier artificial intelligence (AI) methods and drug discovery[17]. There are currently several technologies being used to create quantum computers, including ion trapping and superconducting[17]. While government organizations invest in quantum technology, private corporations like IBM and Google are creating their own quantum computers[37]. For instance, according to the websites for European Union's (EU) Quantum Flagship Project, the EU commission is investing €100 million on quantum technologies. Their primary objective is to minimize hardware flaws that prevent the usage of quantum computers in real-world applications. Whether or if technology ultimately wins the war for quantum hardware, quantum program is the primary tool for developing QC applications[24].
A stack of quantum software that includes compilers, programming languages, and operating systems is required to support quantum software [37]. Conventional software engineering techniques cannot be used to create realistic applications based on quantum software due to the intrinsic properties of quantum computing, such as entanglement and superposition. Moreover, the transition to a completely different programming paradigm with counterintuitive quantum concepts presents substantial difficulties for software engineers when creating quantum programmers[44].
QSE needs to provide methods for developing quantum software. QSE requires tasks such as the design of quantum programs, implementation techniques for quantum algorithms, and testing and maintenance of quantum software [44]. QSE should eventually transition to an incremental, iterative, and agile development process, following the conventional thinking in software programming, that nearly began with hard-wired, hardware focused methodologies in the 1950s and later developed into current agile, iterative development [40]. Therefore, we must develop new QSE approaches (with tool supportive) that encompass various QSE stages. Building on the principles of traditional software engineering, creating reliable software necessitates adhering to an SDLC, which typically consists of phases for requirement gathering, design and architecture, coding, testing and maintenance.

3 QUANTUM SOFTWARE DEVELOPMENT PHASES

To create and imagine quantum software systems, QC is a revolutionary technology that necessitates a new paradigm for software engineering [44]. In QSE , developing a quantum software code is not the top priority problem, however, the problems of design and requirements are far more prevalent and require correction [5]. Therefore, additional aspects of QSE should receive more attention that just quantum coding concerns when discussing quantum software development methodologies[34]. The promise that quantum software development approaches have for achieving design and analysis reuse and reducing difficulty during analysis and design in considerable[40]. A entirely new approach, including a life cycle, must be employed to address these other issues if it is understood that the development of quantum software entails more than just quantum programming.
QSE is the application of solid engineering principles to the creation, deployment, and regular maintenance of quantum programs and any related documentation in order to obtain reasonably priced quantum software that operates effectively on quantum computers[**?**]. Therefore, QSE asks for unique approaches, instruments, procedures, and methods that specifically target the creation of software applications based on quantum physics. In the following sections, we briefly discuss the quantum software development life-cycle phases (i.e., quantum software requirements engineering, quantum software design, quantum software implementation, quantum software testing, and quantum software maintenance) [20, 24, 40, 44]. Table 1 presents the important artifacts of QSE life cycle, and a series of activities for quantum software development derived from[24] and presented in Figure 1.

3.1 Quantum Software Requirements Engineering

We believe that quantum requirements engineering will be like requirements engineering process done for classical computing due to focus on requirement elicitation and management aspect of the phase. However, quantum requirements engineering will need new modeling and specification techniques to model QC aspects such as login functions and state. We believe that requirements engineering research community need to extend classical use cases, user stories and goal modeling techniques to support quantum requirements engineering process[40, 44].

3.2 Quantum Software Design

Like classical software development, quantum software design involves two major phases, i.e., architectural and detail design. Architectural design is quantum software's abstract-level design, where the main components' interactions are described [20]. The detailed design explicitly describes the data structure, algorithms, and interfaces for module interactions. Developing algorithms for quantum software design is challenging compared to classical software systems because of some fundamental quantum computing features, including superposition and entanglement[24].

3.3 Quantum Software Implementation

Some work has been done to develop quantum programming languages [14]. There are several quantum programming languages, i.e., C (QCL), C++ (Scafflod), C# (Q#), Python (ProjectQ, Qiskit, Forest), F# (LIQUi|⟩), Scala (Chisel-Q), and Haskell (Quiper) [14, 41]. However, research community need to focus on developing commercial integrated development environments to support the implementation of quantum software.

3.4 Quantum Software Testing

The testing phase begins to find defects and verify the system's behavior. It might be possible that the programmers make various mistakes when performing the quantum software testing because different characteristics of quantum computers include entanglement, superposition, and the absence of cloning, which make it challenging to predict the quantum software's behavior [40, 44]. There is a strong need for different testing tools and techniques

that consider the quantum computing characteristics such as reading intermediate states[12], handling probabilistic test oracles and facing the decoherence problems.

3.5 Quantum Software Maintenance

The maintenance phase includes updating, changing, and modifying the quantum software to meet customer needs. This is an emerging research area in quantum software engineering, specifically focusing on re-engineering the available classical information systems and their integration with quantum algorithms[12]. In the short term, quantum computers might not be able to give full-fledged features because of the high cost and lack of processes, tools, and techniques [40, 44]. However, migrating the quantum algorithms to the existing classical information systems is the best option[36]. Therefore, it is important to revisit the reengineering (maintenance) concepts to deal with the migration process.

4 CALL FOR ACTION

A wide range of complex computational industrial and societal problems are subject to solution via quantum computing. A few notable examples are the development of vaccines and quick drugs discovery, portfolio management, financial optimization, and intricate physics simulators to better comprehend the world[**?**]. Therefore, the success of QC will unavoidably and significantly affect our daily lives and transform the majority of industries across many disciplines. Such influence must be achieved through quantum software, whose development must be systematically supported through quantum software engineering[44]. QSE offers new research topics to develop practical applications by establishing research societies across domains (such as physics, mathematics, software engineering, and computer science) and collaborations with some other disciplines, such as business, medicine, and chemistry. The rise of QC is revolutionizing many facets of society. It will alter how we perceive and address complex issues and challenges. The methodical and economic development of tomorrow's robust, trustworthy, and useful QC applications depends on QSE[36, 44].

Despite recent improvements in tools for quantum programming, quantum software developers still face two major obstacles. First, it seems sense to be afraid of betting on platforms or languages that might be terminated. Second, the classical software engineering methods, principles and knowledge of quantum programming must be understood by quantum computer scientists[26]. Similarly, the complexity of writing quantum software is another challenge. A working knowledge of quantum physics, a command of mathematics, and expertise od quantum information theories are requirements for quantum programmers[26, 35]. As a result, organizations face problems developing QSE teams[26, 35, 40, 44].

In the software development process, requirements engineering works as a foundation and steers quality software development as per expectations[5]. Quantum software requirements engineering requires a team aware of quantum mechanisms' computation and working style. Similarly, the requirements analysis team needs to develop quantum software engineering methods that specify the stakeholders' needs, quantum characteristics, and quantum attributes[44]. Features and properties of quantum, which are currently unclear, must be defined to perform functional and non-functional activities of a system. No guidelines on quantum mechanics are available to structure complex models. There is a need to develop new and effective hybrid (quantum-classic) guidelines to assist the requirement engineering to consider quantum mechanics while analyzing the requirement and developing the use case and associated UML models.

In QSE, the design phase gives a foundation to construct a hybrid structure of a software system. Hence, it is an essential step to model the quantum-classic characteristics of a software application. In hybrid software design, It is crucial to think of the system as a collection of parts or modules with distinct behaviors and boundaries for each module[24]. A hybrid design should be correct, complete, efficient, flexible, consistent, and maintainable[24]. However, in this new genre of QSE, there is a need to build an understanding of architecting hybrid design software at both high- and low-level abstraction. To make the hybrid design efficient and complete, there is a need for new methods and tools to define abstract specifications, develop quantum modules, describe the functionality of each module, and understand the constraints and elements of quantum gates.

Similarly, low-level abstraction, including data structure design, quantum circuit, algorithm design, quantum interface design, and module communications, are the areas that need the attention of practitioners and the scientific community to develop common representations for elements of quantum software.

The implementation phase concerns software translating hybrid design specifications into the quantum source code. For quantum programming, the programmer requires a basic understanding of the quantum mechanism and software engineering [24, 44]. Using traditional software implementation methods is rare as we deal with circuits and quantum programming languages. We are in the early stage of quantum software implementation. We need more actions in terms of developing tools and technologies, quantum simulators, executing programs on servers, designing circuit diagrams, and developing of quantum programming roadmap. By putting the coding quality standards into practice, many issues and the possibility of project failure reduces. Thus, there is a call for action for developing quantum programming standards and programming guidelines.

Software testing is very important in ensuring system quality control. It must ensure that the requested expectations of quantum software stakeholders have been satisfied[8]. The testing phase is also important to detect the errors of early phases that are related to project success as it directly reflects the project budget, time, and resources. As QSE is a new branch of software engineering, where quantum software has different characteristics such as no cloning, entanglement, and superposition, which makes it challenging to predict the behavior of quantum software [24, 44]. Dealing with quantum software testing using classical testing techniques and tools is not enough to make the quantum software error-free and up to the mark. This paper call for action to the development of testing tools, techniques, processes, and standards guidelines considering the quantum characteristics of the software system.

Table 1: Quantum Software Development Process

QS Requirements Engineering	**Functional**	Define quantum features	Focus on user requirements with quantum	Capture quantum mechanism in the use case	Authentication and authorization levels
	Non-Functional	Define quantum properties	Focus on user expectations with quantum	Capture quantum as a quality attribute	Usability, reliability, scalability, performance
Quantum Software Design	**High-level design**	Quantum abstract specification	Develop Quantum Modules	Describe the functionality of each module	Understanding of quantum gates
	Low-level design	Quantum data structure design	Quantum Circuit and Algorithm design	Quantum interface design	Module communication
Quantum Software Implementation	Translating quantum circuit diagrams	Specify Quantum programming language	Program; specifies the quantum circuit	Test using Quantum simulator	Execute program on server
Quantum Software Testing	Develop test plans	Test design description	Test case description	Test reports and logs	Measuring the Qubits
Quantum Software Maintenance	Determination of maintenance need	Determination of maintenance scope	Quantum Architectural Design Maintenance	Quantum Modules Maintenance	Unit and acceptance testing

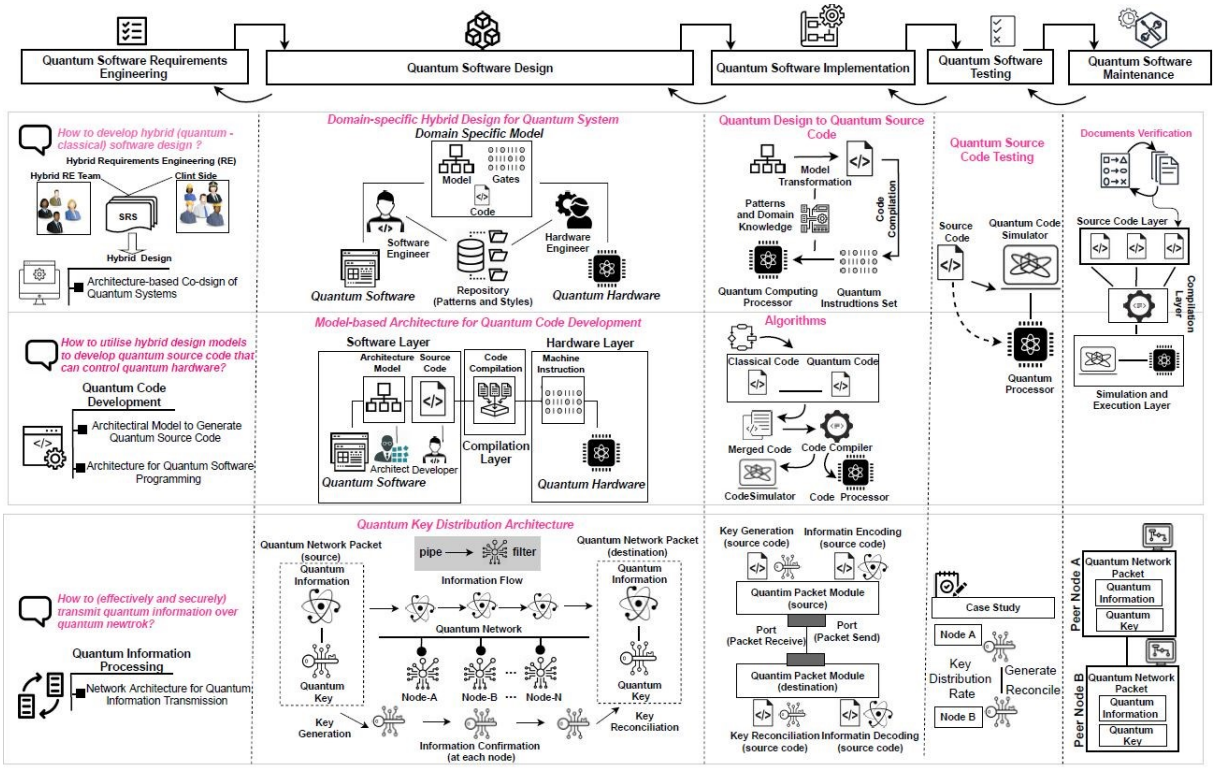

Figure 1: Quantum software development activities

Quantum software maintenance refers to the modifications created by correcting, inserting, deleting, extending, and enhancing the baseline of quantum software for successful maintenance, it is obligatory to estimate the demanded maintenance and the impact of change [9, 40]. There is a need for a quantum software experts-based change control board to identify, and verify the root cause, and effects of requested maintenance. As quantum software engineering is a new paradigm, there is a lack of quantum-oriented tools, techniques, and matrices to measure the scope of request maintenance. Hence, there is a demanding need for the develop of a set of tools, techniques, matrixes, and standards guidelines for the quantum software maintenance process.

Acknowledgments

We are thankful to the research teams of LUT University, Research Unit for Empirical Software Engineering at M3S, King Fahd University of Petroleum and Minerals, and University of Oulu for giving their insights to come up with this vision paper. Moreover, this research is supported by the PHP Foundation with the grant 20220006.

References

[1] Aakash Ahmad, Arif Ali Khan, Muhammad Waseem, Mahdi Fahmideh, and Tommi Mikkonen. 2022. Towards process centered architecting for quantum software systems. In *2022 IEEE international conference on quantum software (QSW)*. IEEE, 26–31.

[2] Aakash Ahmad, Arif Ali Khan, Muhammad Waseem, Mahdi Fahmideh, and Tommi Mikkonen. 2022. Towards process centered architecting for quantum software systems. In *2022 IEEE international conference on quantum software*

(QSW). IEEE, 26–31.
[3] Muhammad Azeem Akbar, Arif Ali Khan, and Zhiqiu Huang. 2023. Multicriteria decision making taxonomy of code recommendation system challenges: A fuzzy-AHP analysis. *Information Technology and Management* 24, 2 (2023), 115–131.
[4] Muhammad Azeem Akbar, Arif Ali Khan, and Sami Hyrynsalmi. 2024. Role of quantum computing in shaping the future of 6 G technology. *Information and Software Technology* 170 (2024), 107454.
[5] Muhammad Azeem Akbar, Arif Ali Khan, Sajjad Mahmood, and Alok Mishra. 2023. SRCMIMM: the software requirements change management and implementation maturity model in the domain of global software development industry. *Information Technology and Management* 24, 3 (2023), 195–219.
[6] Muhammad Azeem Akbar, Saima Rafi, and Arif Ali Khan. 2022. Classical to quantum software migration journey begins: a conceptual readiness model. In *International Conference on Product-Focused Software Process Improvement*. Springer, 563–573.
[7] Muhammad Azeem Akbar, Jun Sang, Arif Ali Khan, Fazal-E Amin, Shahid Hussain, Mohammad Khalid Sohail, Hong Xiang, Bin Cai, et al. 2018. Statistical analysis of the effects of heavyweight and lightweight methodologies on the six-pointed star model. *IEEE Access* 6 (2018), 8066–8079.
[8] Muhammad Azeem Akbar, Jun Sang, Arif Ali Khan, Muhammad Shafiq, Shahid Hussain, Haibo Hu, Manzoor Elahi, Hong Xiang, et al. 2017. Improving the quality of software development process by introducing a new methodology–AZ-model. *IEEE Access* 6 (2017), 4811–4823.
[9] Muhammad Azeem Akbar, Mohammad Shameem, Arif Ali Khan, Mohammad Nadeem, Ahmed Alsanad, and Abdu Gumaei. 2021. A fuzzy analytical hierarchy process to prioritize the success factors of requirement change management in global software development. *Journal of Software: Evolution and Process* 33, 2 (2021), e2292.
[10] Muhammad Azeem Akbar, Kari Smolander, Sajjad Mahmood, and Ahmed Alsanad. 2022. Toward successful DevSecOps in software development organizations: A decision-making framework. *Information and Software Technology* 147 (2022), 106894.
[11] Melis Ozge Alas, Fehmi Burak Alkas, Ayca Aktas Sukuroglu, Rukan Genc Alturk, and Dilek Battal. 2020. Fluorescent carbon dots are the new quantum dots: an overview of their potential in emerging technologies and nanosafety. *Journal of Materials Science* 55 (2020), 15074–15105.
[12] Shaukat Ali, Tao Yue, and Rui Abreu. 2022. When software engineering meets quantum computing. *Commun. ACM* 65, 4 (2022), 84–88.
[13] Ehud Altman, Kenneth R Brown, Giuseppe Carleo, Lincoln D Carr, Eugene Demler, Cheng Chin, Brian DeMarco, Sophia E Economou, Mark A Eriksson, Kai-Mei C Fu, et al. 2021. Quantum simulators: Architectures and opportunities. *PRX quantum* 2, 1 (2021), 017003.
[14] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. 2017. Quantum machine learning. *Nature* 549, 7671 (2017), 195–202.
[15] Yuan Cao, Yongli Zhao, Qin Wang, Jie Zhang, Soon Xin Ng, and Lajos Hanzo. 2022. The evolution of quantum key distribution networks: On the road to the qinternet. *IEEE Communications Surveys & Tutorials* 24, 2 (2022), 839–894.
[16] Manuel De Stefano, Fabiano Pecorelli, Dario Di Nucci, Fabio Palomba, and Andrea De Lucia. 2022. Software engineering for quantum programming: How far are we? *Journal of Systems and Software* 190 (2022), 111326.
[17] Thomas G Draper. 2000. Addition on a quantum computer, 2000. *Los Alamos Physics Preprint Archive* (2000).
[18] Vedran Dunjko, Jacob M Taylor, and Hans J Briegel. 2016. Quantum-enhanced machine learning. *Physical review letters* 117, 13 (2016), 130501.
[19] Trung Q Duong, Long D Nguyen, Bhaskara Narottama, James Adu Ansere, Dang Van Huynh, and Hyundong Shin. 2022. Quantum-inspired real-time optimization for 6G networks: opportunities, challenges, and the road ahead. *IEEE Open Journal of the Communications Society* 3 (2022), 1347–1359.
[20] Iaakov Exman and Alon Tsalik Shmilovich. 2021. Quantum software models: the density matrix for classical and quantum software systems design. In *2021 IEEE/ACM 2nd International Workshop on Quantum Software Engineering (Q-SE)*. IEEE, 1–6.
[21] Tiago M Fernandez-Carames and Paula Fraga-Lamas. 2019. A review on the application of blockchain to the next generation of cybersecure industry 4.0 smart factories. *Ieee Access* 7 (2019), 45201–45218.
[22] Laszlo Gyongyosi and Sandor Imre. 2019. A survey on quantum computing technology. *Computer Science Review* 31 (2019), 51–71.
[23] Thomas Häner, Damian S Steiger, Krysta Svore, and Matthias Troyer. 2018. A software methodology for compiling quantum programs. *Quantum Science and Technology* 3, 2 (2018), 020501.
[24] Arif Ali Khan, Aakash Ahmad, Muhammad Waseem, Peng Liang, Mahdi Fahmideh, Tommi Mikkonen, and Pekka Abrahamsson. 2023. Software architecture for quantum computing systems—A systematic review. *Journal of Systems and Software* 201 (2023), 111682.
[25] Ryan LaRose. 2019. Overview and comparison of gate level quantum software platforms. *Quantum* 3 (2019), 130.
[26] Heng Li, Foutse Khomh, Moses Openja, et al. 2021. Understanding quantum software engineering challenges an empirical study on stack exchange forums and github issues. In *2021 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 343–354.
[27] Gwo-Ching Liao. 2012. Solve environmental economic dispatch of Smart MicroGrid containing distributed generation system–Using chaotic quantum genetic algorithm. *International Journal of Electrical Power & Energy Systems* 43, 1 (2012), 779–787.
[28] Nabil Mohammed Ali Munassar and A Govardhan. 2010. A comparison between five models of software engineering. *International Journal of Computer Science Issues (IJCSI)* 7, 5 (2010), 94.
[29] Vankamamidi S Naresh, Moustafa M Nasralla, Sivaranjani Reddi, and Iván García-Magariño. 2020. Quantum diffie–hellman extended to dynamic quantum group key agreement for e-healthcare multi-agent systems in smart cities. *Sensors* 20, 14 (2020), 3940.
[30] Miroslaw Nawrocki, Krzysztof Kurowski, and Radoslaw Gorzenski. 2021. Cyber Space and Aviation 4.0–New Testing Facilities for Next Generation of Cyber-Physical, Autonomous and Air Traffic Control Systems. In *Modern Technologies Enabling Safe and Secure UAV Operation in Urban Airspace*. IOS Press, 70–84.
[31] Laurentiu Nita, Laura Mazzoli Smith, Nicholas Chancellor, and Helen Cramman. 2023. The challenge and opportunities of quantum literacy for future education and transdisciplinary problem-solving. *Research in Science & Technological Education* 41, 2 (2023), 564–580.
[32] Ramakrishna SS Nuvvula, Elangovan Devaraj, Rajvikram Madurai Elavarasan, Seyed Iman Taheri, Muhammad Irfan, and Kishore Srinivasa Teegala. 2022. Multi-objective mutation-enabled adaptive local attractor quantum behaved particle swarm optimisation based optimal sizing of hybrid renewable energy system for smart cities in India. *Sustainable Energy Technologies and Assessments* 49 (2022), 101689.
[33] Carlos Outeiral, Martin Strahm, Jiye Shi, Garrett M Morris, Simon C Benjamin, and Charlotte M Deane. 2021. The prospects of quantum computing in computational molecular biology. *Wiley Interdisciplinary Reviews: Computational Molecular Science* 11, 1 (2021), e1481.
[34] Guido Peterssen, Jose Luis Hevia, and Mario Piattini. 2022. Quantum software development with QuantumPath®. In *Quantum Software Engineering*. Springer, 251–268.
[35] Mario Piattini, Guido Peterssen, and Ricardo Pérez-Castillo. 2021. Quantum computing: A new software engineering golden age. *ACM SIGSOFT Software Engineering Notes* 45, 3 (2021), 12–14.
[36] Mario Piattini, Manuel Serrano, Ricardo Perez-Castillo, Guido Petersen, and Jose Luis Hevia. 2021. Toward a quantum software engineering. *IT Professional* 23, 1 (2021), 62–66.
[37] Somayeh Bakhtiari Ramezani, Alexander Sommers, Harish Kumar Manchukonda, Shahram Rahimi, and Amin Amirlatifi. 2020. Machine learning algorithms in quantum computing: A survey. In *2020 International joint conference on neural networks (IJCNN)*. IEEE, 1–8.
[38] Chi-Sheng Shih and Kai-Wei Yang. 2019. Design and implementation of distributed traceability system for smart factories based on blockchain technology. In *Proceedings of the Conference on Research in Adaptive and Convergent Systems*. 181–188.
[39] Benjamin Weder, Johanna Barzen, Frank Leymann, Marie Salm, and Daniel Vietz. 2020. The quantum software lifecycle. In *Proceedings of the 1st ACM SIGSOFT International Workshop on Architectures and Paradigms for Engineering Quantum Software*. 2–9.
[40] Benjamin Weder, Johanna Barzen, Frank Leymann, and Daniel Vietz. 2022. Quantum software development lifecycle. In *Quantum Software Engineering*. Springer, 61–83.
[41] Robert Wille, Bing Li, Ulf Schlichtmann, and Rolf Drechsler. 2016. From biochips to quantum circuits: Computer-aided design for emerging technologies. In *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 1–6.
[42] Nan Wu and Jingjuan Sun. 2022. Fatigue detection of air traffic controllers based on radiotelephony communications and self-adaption quantum genetic algorithm optimization ensemble learning. *Applied Sciences* 12, 20 (2022), 10252.
[43] Jia Yang, Yingxu Wu, Qiang Xu, Xiaoli Liu, and Huan Sheng. 2019. Online Monitoring and Routing Algorithm for WSN in Smart Distribution Network Based on Quantum Ant Colony. In *2019 Chinese Automation Congress (CAC)*. IEEE, 2537–2541.
[44] Jianjun Zhao. 2020. Quantum software engineering: Landscapes and horizons. *arXiv preprint arXiv:2007.07047* (2020).
[45] Torben Zülsdorf, Christopher Coenen, Arianna Ferrari, U Fiedler, Colin Milburn, and Matthias Wienroth. 2011. Quantum engagements: social reflections of nanoscience and emerging technologies. *Studies of New and Emerging Technologies* (2011).

Received 2024-04-25; accepted 2024-05-06