Agile Practices for Quantum Software Development: Practitioners' Perspectives

Arif Ali Khan*, Muhammad Azeem Akbar†, Aakash Ahmad‡, Mahdi Fahmideh §, Mohammad Shameem ¶ , Valtteri Lahtinen ‖, Muhammad Waseem**, Tommi Mikkonen**
*M3S Empirical Software Engineering Research Unit, University of Oulu, Oulu, Finland
†Department of Software Engineering, LUT University, Lappeenranta, Finland
‡School of Computing and Communications, Lancaster University Leipzig, Germany
§School of Business at University of Southern Queensland, Queensland, Australia
¶Department of CSE, Koneru Lakshmaiah Education Foundation, Guntur, Andhra Pradesh, India
‖Quanscient Oy, Tampere, Finland
** Faculty of Information Technology, University of Jyvaskyla, Finland
*arif.khan@oulu.fi, †azeem.akbar@ymail.com, ‡a.ahmad13@lancaster.ac.uk, §mahdi.fahmideh@usq.edu.au
¶shameem.ism@gmail.com, ‖valtteri.lahtinen@quanscient.com, **muhammad.m.waseem@jyu.fi, **tommi.j.mikkonen@jyu.fi

***Abstract*—Quantum software engineering is an emerging genre of software engineering that exploit principles of quantum bits (Qubit) and quantum gates (Qgates) to solve complex computing problems effeciently than their classical counterparts. According to its proponents, agile software development practices have the potential to address many of the problems endemic to the development of quantum software. However, there is a dearth of evidence investigating whether agile practices are suitable for, and can be adopted by, software teams in the context of quantum software development. To address this lack, we conducted an empirical study to investigate the needs and challenges of using agile practices to develop quantum software. While our semi-structured interviews with 26 practitioners across 10 countries highlighted the applicability of agile practices in this domain, the interview findings also revealed new challenges impeding the effective incorporation of these practices. Our research findings provide a springboard for further contextualization and seamless integration of agile practices in quantum software engineering (QSE) to develop emerging and next generation of quantum software systems and application.**

***Index Terms*—Agile Practices, Quantum Software Engineering, Empirical Software Engineering**

I. Introduction

The field of Quantum Computing (QC) has received rapidly growing industrial and policy interest at global forums. It is expected to bring revolution in various industrial areas [1]. It is evident by the fact that technology giants such as IBM [2], Google [3], and Microsoft [4] have heavily invested in implementing QC as a service offering to tackle a new class of complex computational problems. Developing QC applications, on the other hand, is a challenging endeavour due to radically distinct challenges posed by QC machines that operate on fundamentals of quantum mechanics. The literature suggests that QC development is, after all, essentially a type of system development endeavor [1]. Given this analogy, adopting a systematic engineering lifecycle perspective for managing the complexity of QC development is acclaimed [1]. This takes precedence over an ad-hoc use of implementation techniques and technologies or relying on the development skills of the individuals that may likely to deliver QC, which may be erroneous and costly to maintain [5] [1]. Hence, to achieve an optimum effect, there must be advanced SE methods to enable QC to live up to its promising potential.

In addition to a plethora of hardware and networking components, the key enabler to develop programmable QCs is quantum software [6]. Quantum software - amongst other requirements such as full stack support ranging from novel techniques and tools - needs processes and methods that explicitly focus on developing software system based on quantum mechanics [6]. However, the present-day quantum software engineering (QSE) processes are far from being mature, and are most often based on hybrid concepts (consisting of quantum-classical tools and practices). The quantum-classical development must be orchestrated; therefore, we previously presented the vision of embracing iterative and agile practices for developing a quantum software [7]. The QSE processes can benefit from well-developed iterative agile practices for team collaboration, short development iterations, and continuous delivery [8] [9]. Presently, adopting a more "agile" approach to develop a quantum software system is reliable option rather than waiting for specific QSE processes and methods [6]. However, no empirical evidence yet exists that ties agile practices and quantum software development activities. This study aims at exploring the significance of agile practices in the QSE domain.

To this end, we conducted semi-structured interviews with a total of 26 software practitioners (having mutually inclusive knowledge of agile software engineering and quantum software development) across 10 countries with diverse roles such as quantum algorithm developer, software developer, agile coach working in varying domains ranging from cyber security to automotive and healthcare, allowing us to gain different perspectives on our research aim. To analyse interview results, we adopted the Grounded Theory (GT) method [10] in a bottom-up approach to derive a mapping of code, concepts, and categories from raw (interview) data.

The results indicate that practitioners view agile practices as the best fit to support quantum software development activities. However, small portion of the interview participants find themselves underprepared due to lack of a basic knowledge of quantum mechanics, insufficient expertise and non-availability of tools, that hinders the progress of agile-driven quantum software development.

We outline the primary contributions of this research as (i) empirically-derived understanding on the adoption of agile practices in quantum software development, (ii) discussing four major categories of challenges *(knowledge and awareness, sustainable scaling, quantum-aware tools and technologies* and *standards and specifications)* that must be addressed for useful adoption of agile practices in quantum software development domain. The study results aim to enlighten software researchers and practitioners by pinpointing the understanding and challenges of using agile practices for developing a quantum software.

II. Background

Firstly, we contextualise the development of software-intensive systems that can be executed or simulated on QC platforms in Section II-A and further introduce agile practices for software development in Section II-B. The concepts and terminologies introduced in this section are used throughout the paper to elaborate technical aspects of SE for QC.

A. Quantum Software Engineering

Quantum computers harness the phenomena of quantum mechanics, e.g., quantum superposition and entanglement to process, store, and transmit quantum information set represented with Qubits that manipulate Qgates. In contrast to classic binary digits represented as (0, 1), Qubit as the most fundamental unit of quantum information set attains a state that is a superposition of 0 and 1, and represented as $|0\rangle$ and $|1\rangle$ [1] [11], where:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{1}$$

From computation (i.e., QC) and software system development (i.e., QSE) perspective, a quantum computer disrupts the idea of traditional digital systems and represent a paradigm shift (transitioning from bits to Qubits) with a vision to support next generation computing [1]. In a QC context, operationalising Qubits that manipulate quantum circuits (Qgates as a primitive unit of quantum hardware), software programmers require quantum age algorithms and programming languages such as Microsoft Q# [4], IBM Qiskit [2] and Cirq by Google [3]. Quantum programming language and their underlying algorithms are well suited to focus on implementation details that produce executable specifications on quantum computer; however, they lack an engineering view for developing quantum software including the activities such as quantum domain engineering, quantum system co-design, quantum algorithm design and source coding and quantum information simulation [7] [12]. QSE as the most recent genre of SE aims to exploit the status-quo (ISO/IEC/IEEE 12207:2017 standard) [13] by leveraging the engineering processes, reference architectures, patterns, tools and framework to develop quantum-intensive software. There is no well established process or even a notion of how QSE process might look, however; some recent research efforts are seen as attempts to find a common denominator of engineering activities to establish foundations for process-centric QSE, e.g., [14] [7] [12]. Building on the notion of a QSE intensive process, new software development processes and methods or customizing the existing conventional methods (hybrid quantum-classical) are required [14] [15] [5].

B. Agile for Quantum Software Development

In line with the legacy of experiences in conventional programming, which also started from hardware-focused, hard-wired techniques in 1950s and then evolved into today's agile system development practices, the QSE should eventually follow the same agile development tradition [16] [17]. However, tools and methods that are used to achieve agility in conventional software engineering need to be examined in line with the characteristics of quantum software development [7]. When coding quantum programs, software developers face new challenges due to switching to an entirely different programming mindset with counterintuitive quantum principles [7]. For example, executing the quantum instructions in the state of qubits and measuring the qubit values [18].

Testing quantum programs requires at least the development of solutions to tackle the following challenges: (1) defining test oracles as the state of a quantum program can be in superposition and thus difficult to get precise state; (2) running efficient quantum test data generation since quantum variables might be exponentially higher than classical variables; and (3) dealing with false positives/negatives due to vulnerabilities to hardware glitch [19] [20]. Finally, faults found in quantum programs with testing must be located, isolated, and patched; thus, debugging is needed. Furthermore, developing effective debugging solution is impeded by the following challenges of: (1) examining values of quantum variables in superposition; (2) interpreting multi-dimensional quantum states, despite the availability of simulation techniques, and (3) lacking guidance on where and what to check when debugging quantum programs [21].

In conventional software engineering, the above mentioned challenges are alleviated via adopting the agile and iterative principles. Proponents of the agile practices claim, in contrast to the traditional methodologies (e.g. Waterfall model) [22], agile solves many of the problems endemic to the field by proposing principles and practices such as active user involvement, short iterations, small and frequent release, and refactoring [8]. This, in turn, supports their state of flow, which is an important ingredient in modern-day software development [23]. In present scenario, QC is still in its infancy, and the situation of developing a quantum software could considerably be eased by adopting agile practices as suggested by Piattini et al. [17]: *"we should adopt a more*

"agile" approach when proposing and developing software engineering quantum techniques, that is, do not wait until quantum programming languages are "stable" or "refined" in order to adapt existing techniques or create new ones, but develop them in parallel with the evolution of the quantum languages, starting from now". Today's quantum software development activities are mostly hybrid (classical and quantum) [14], therefore conventional agile practices provide a set of best practices for developing a quantum software through collaborative efforts of self-organising and cross-functional teams. Gonzalez and Paradela [15] complemented it by explicitly mentioning that: *"Quantum software development projects currently have numerous features that fit neatly into the agile paradigm, such as adding features evolutionarily or using trialand-error algorithms"*.

However, to the best of our knowledge, no empirical study has yet been conducted to explore the perception of practitioners for using agile practices to develop a quantum software system. This study aims to fill the given research gap by conducting interviews with practitioners to address the following core research questions (RQs):

RQ1: Are agile practices best suited for developing quantum software?

RQ2: What are the challenges of adopting agile practices for developing a quantum software system?

III. Methodology

We now detail the research methodology, as illustrated in Figure 1, which comprises the following steps.

A. Context (Process-centred quantum software)

To understand a structured approach for developing quantum software, we collaborated with industrial partner (Quanscient Oy)[1] and studied the incremental quantum software development process [7] - including four iterative activities (i) quantum domain engineering (ii) quantum system co-design (iii) Quantum algorithm design and implementation, and (iv) quantum code simulation and validation (see Figure 1). For illustrative purpose, Figure 1 demonstrates what iterative activities needs to be done and each activity details how it is to be done.

However, this study aims to extend our previous work [7] by using agile practices to support the quantum software development process activities. The findings of this study will provide in-depth understanding of agile-driven quantum software development. Therefore, we structured the interview instrument as follows to achieve the study aims.

B. Interview

Interview instrument: Following the guidelines by Robinson [24], we developed semi-structured questions which covered three categories, including (i) demography and professional details, (ii) suitability of agile practices for developing quantum software (RQ1), recorded via a 5-point Likert scale, detailed later and (iii) challenges of agile-driven quantum software development as open-ended question for spontaneous responses (RQ2) (analysed with GT in Figure 1). The first three authors conducted regular meetings and developed the interview questions; finally, a Zoom meeting was called and invited all other authors to conclude the interview instrument[2].

Recruiting participants: We strictly sought practitioners with a mutually inclusive knowledge of 'agile SE' and 'QC' ensured via contacting the potential participants and informal discussion on their willingness and required basic knowledge of the subject. We used relevant platforms such as GitHub, professional social media networks (i.e., LinkedIn, ResearchGate, WeChat, Facebook, mailing groups), industrial network and personalised emails to contact the targeted population. A total of 73 practitioners were contacted and eventually 26 participated in the interview as shown in Figure 2(a).

We shared the final interview questions with 26 agreed participants a week before the session and provided the author's contact if they had any questions regarding the interview material. The second author conducted the interview sessions online using platforms such as Zoom, VooVmeeting, and Microsoft Teams.

IV. Analysing Interview Data

Based on the semi-structured interview questions (Section III-B), we analysed the data in three steps. First step analysed the demographic details of the practitioners such as their country, years of experience, professional domain etc., as presented in Figure 2. Demography details complement the analysis of interview data corresponding to RQ1 and RQ2. For example to analyse if factors like years of experience, domain of experience (cyber security or automotive engineering etc.) and/or professional roles (agile coach or quantum algorithm developer etc.) impacts practitioners perspective or knowledge on the application of agile practices to quantum software development.

Second step involved recording the interview data for RQ1, using a 5-point Likert scale *(Strongly Agree, Agree, Neutral, Disagree, Strongly Disagree)* followed by an open-ended question to rationalise or describe their choice, as shown in Figure 3.

The last step focused on RQ2 and we adopted the Grounded Theory (GT) method [10] to derive a mapping of code, concepts, and categories as in Figure 1. GT is a systematic method to generate a theory or conceptual view out of raw data and to enable researchers to have amenable interpretation and free comprehension of data in different ways [10]. We adopted the GT approach for various reasons, e.g., Hoda et al. [25] mentioned that GT is the best fit for novel research fields which have not been previously well explored, and the research on agile in quantum software development is scarce. GT has received growing attention in software engineering research and more narrow in the agile domain e.g. [26] [27] [28].

[1] https://quanscient.com/

[2] https://tinyurl.com/5y2fd56m

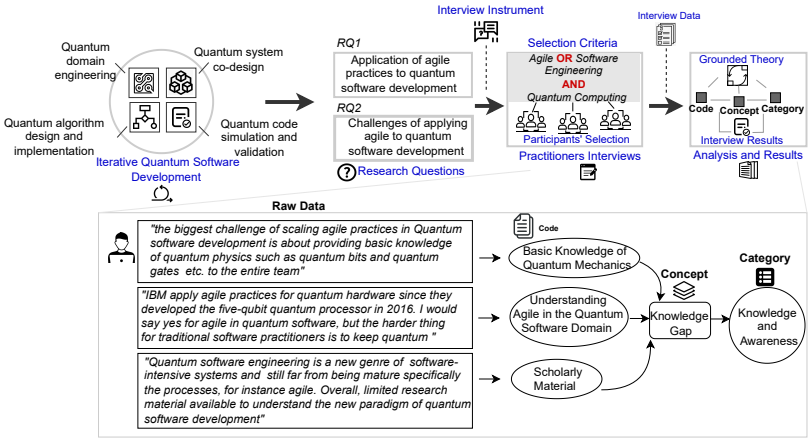

Fig. 1: Overview of the research methodology

The GT open coding [29] and constant comparison approaches [10] were followed to analyze the raw data and identify the core categories of agile-driven quantum software development challenges. The data coding and mapping are preliminary conducted by the first three authors following the GT guidelines [10]. However, the rest of the authors were invited to participate and provide feedback in the final data coding and concept development process. The following raw data example is presented to explain the analysis process stage to coding, concept development, and categorization (Figure 1).

Raw data: *"the biggest challenge of scaling agile practices in Quantum software development is about providing basic knowledge of quantum physics such as quantum bits and quantum gates etc. to the entire team"* [P17], Quantum sofwtare engineer.

Code: *Basic knowledge of quantum mechanics*

The codes evolved from each interview response were constantly compared with the codes of the same interview and the entire data set, including the interviews of other participants. The mentioned code *"basic knowledge of quantum mechanics"* (P17, Quantum software engineer) was found similar to two other codes, namely *"understanding agile in the quantum software domain"* (P10, Scrum master) and *"scholarly material"* (P14, Agile trainer). The similar codes were grouped and defined the common concept, which is the higher level of abstraction.

Concept: *Knowledge gap*

The other relevant concepts that emerged during the constant comparison of codes are the *"team dynamics"* and *"quantum software development education"*. The final constant comparison was conducted for a similar group of concepts to create the final, and high level of abstraction called categories.

Category: *Knowledge and awareness*

Knowledge and awareness is a high-level concept that encapsulates the insights and perceptions of participants, who were more concerned about the guidance, support and awareness of quantum computing concepts and their integration with agile principles.

In this study, the level of abstraction (coding, concept creating, category development) was based on real-world data; therefore, the subsequent discussion is grounded in the context of the collected data. Note that the other categories of challenges were derived following the same open coding and constant comparison methods. However, we provided a sample analysis process only for the knowledge and awareness category but omitted it for all the other categories because of space restrictions. The emerged categories represent the challenges of agile-driven quantum software development. We are not claiming the universal generalizability of the study results grounded on the collected data; however, it perfectly described the investigated context [30].

The categories constituting *knowledge and awareness, sustainable scaling, quantum-aware tools and technologies* and *standards and specifications* along with the important quotes from practitioners are delineated in the following sections. We further considered the developed concepts to support the findings; however, the space restrictions limited us from describing them in detail.

V. Results

This section presents the study results, addressing the two RQs outlined in Section II. Moreover, the demographic analysis was conducted to examine the dimensions and dynamics of the targeted population and contextualize the responses that complement agile-driven quantum software development for a specific group of practitioners (see Figure 2). We noticed that 26 respondents from 10 countries across 4 continents with 16 roles and 15 different work domains participated in this study (see Figure 2(a,d,e)). Note that each interview participant is tagged with a unique id [P#], as shown in Figure 2(d). The experience of interview participants in the quantum software development domain mostly ranges from 0 to 3, which is (n=19) of the total responses (see Figure 2(c)). Of all the responses, the majority (n=16) have 6-10 years of professional working experience as practitioners (see Figure

2(b)). The results illustrate that most (n=13) of the interview participants work as quantum system engineers, which is further classified across 6 different sub-roles (see Figure 2(d)). Therefore, the given demographic findings reveal that the interview participants are diverse with respect to geographical locations, experiences, roles, and industrial domains. It gives us the confidence to generalize the study findings to some extent. The detailed results to address both RQ1 and RQ2 are presented in the following sections.

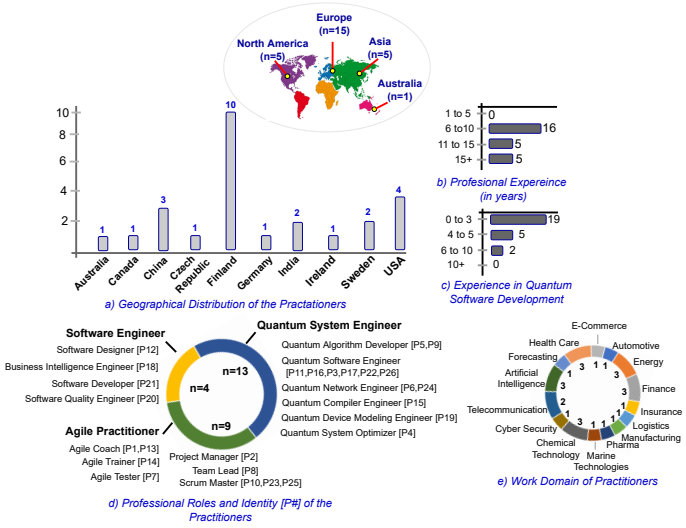

Fig. 2: Overview of interview participants demographics

A. Agile for Quantum Software Development (RQ1)

To understand practitioners' perspectives on the applicability of agile practices to develop quantum software, we sought their feedback using a five-point Likert scale as shown in Figure 3. The selection of a five-point pre-defined response was complemented by an open-ended optional question that allowed practitioners to express the description of their choice spontaneously. Figure 3 shows that a total of 12 practitioners agreed strongly, and 5 agreed (i.e., n=17) that agile practices can be used to develop quantum software. The open-ended descriptions suggest that practitioners perceive QSE as another genre of software engineering, domain modeling, incremental mapping of Qgates and Qubits, and quantum system co-design that can fit well in agile software development. For example, a participant suggested that s/he strongly agrees on the application of agile practices to quantum software development as quantum system co-design, i.e., translating the quantum software requirements into quantum design and source code can benefit from the incremental approach of agile.

On the other hand, three participants remained neutral, rationalizing their view on knowledge of quantum mechanics (operationalizing Qubits) can be challenging for traditional software engineers (see Figure 3). Furthermore, 2 participants strongly disagreed, while 4 disagreed (i.e., n=6) on application of agile for quantum software development. They justified the counterproductivity of applying agile practices in quantum software development by reasons such as immature tools and technologies, lack of professional expertise, and a well-curated architecture. We conclude that while answering RQ1, approximately 2/3 majority recommended the application of agile practices to develop quantum software, an incremental mapping of quantum domain concepts can help to operationalize quantum software. Approximately 1/4 of practitioners cited as non-availability of tools, and professional expertise can be a reason for their disagreement.

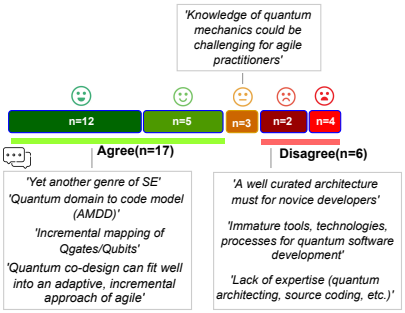

Fig. 3: Agile practices for quantum software development

B. Challenges of Agile-driven Quantum Software Development (RQ2)

The identified five key categories (*knowledge and awareness, sustainable scaling, quantum-aware tools and technologies* and *standards and specifications*) present the challenging aspects that need to consider for scaling agile practices in the quantum software development domain (see section IV). The following sections describe each category in detail with the quote support selected from the interview data.

1) Knowledge and awareness: The awareness and knowledge of agile practices in the quantum software development domain have emerged as an important category, consisting of three concepts: knowledge gap, team dynamics, and quantum software development education.

- **Knowledge gap**

Agile practitioners most likely experience some particular areas of quantum software development more challenging because of the knowledge gap (classical->quantum), such as quantum domain engineering, architecting quantum software, quantum source code interpretation, and quantum algorithm complexity [12] [31]. For instance, the conventional agile practitioner might not know how to design a quantum algorithm attuned to operationalize Qubits rather than classical bits [12].

The feedback of interview participants is aligned with it and perceived that knowledge of quantum computing helps practitioners in the better adoption of agile practices as endorsed by one of the interviewee:

👉 *"But as this wave of innovation continues, a necessary element lags behind: a quantum-skilled workforce. To develop quantum software workforce, the agile iterations give quick feedback and chance of speed learning and experiences"* [P10], Scrum master.

Generally, agility refers to delivering customer value faster, incremental, and with less headaches [8] [22]. Despite this, managing agile activities in the quantum software domain considered with worries [7]. Participants recommended the need for the alignment among skills, quantum awareness, resources, and technologies for developing agile-based quantum software, as mentioned by an interview respondent.

👉 *"Agile teams can get bogged down by failure in the novel quantum paradigm. However, starting with the end goal is imperative when building an agile team to work on a quantum software engineering project. How will you reach your goal, and what skills, technology, and hours will you need? Only then you can build the team"* [P13], Agile coach.

- **Team dynamics**

Agile practices have inherent focus on team structure (e.g., common goals, roles, responsibilities, collaboration), and organized team dynamics is a key to high functional agile productivity [32] [8] [22]. A happy agile team will essentially be more productive; conversely, the team can be extremely ineffective [32]. Agile team dynamics is challenging to manage in the quantum software domain because of the different quantum rules set. The agile team's mindset will likely change due to quantum physics characteristics such as superposition, entanglement, and quantum interference, which significantly influence the way software development team works [7]. This rationale was captured quite well by a respondent:

👉*"Because of flaky characteristics of QC, agile teams face significant challenges due to switching to an entirely different programming paradigm. The bits and qubits interpretations and manipulation bring high-level confusion for agile team forming, storming, norming, and performing"* [P16], Quantum software engineer.

- **Quantum software development education**

Quantum software development as a computer science subject is still at an early stage [33]- in contrast to QC education, which is already taught in various universities [34]. The increasing boom of QC technologies calls for educational approaches that structure core ideas and terms for quantum software development as a subject [33]. The respective applications of core ideas constitute the foundation for defining quantum software development curricula. The above literature findings were supported by the respondents more narrow in an agile-quantum context, for example:

👉 *"Academic treatment will give learning opportunities to realize the capabilities of agile practices for developing quantum software and allow students to exercise their cognitive muscles and joined-up thinking. Incorporating agile-based quantum software engineering as a subject in degree programs will help in preparing a skilled workforce to fulfill the future needs of the quantum software industry"* [P20], Software quality engineer.

2) Sustainable scaling: Emerging quantum software engineering is a paradigm shift that demands enhancing the conventional processes and practices to support requirements of quantum reality [1] [5]. Piattini et al. [17] recommended scaling the conventional agile practices to develop quantum software; however, it is essential to set sustainable strategies to avoid potential risks and scaling pitfalls [35]. For instance, one of the interview participants mentioned that:

👉 *"Organizational and team structure is a huge part of the hybrid agile-quantum process, and top management should be prepared for it. Remember, be fast, and sustainable in integrating agile with quantum software lifecycle by identifying the process aspects that need improvement else immediately find alternatives"* [P2], Project manager.

The sustainable scaling category is based on the two concepts: ethically aligned quantum software and agile-quantum ecosystem.

- **Ethically aligned quantum software**

QSE is a new genre of software development, and the boom of QC from research to the business raises various ethical concerns [36]. The QC transition threatens the existing ethical protections, e.g., security, and transparency [35]. For instance, one of the interview participants justify it more in agile-driven quantum software scenario:

👉 *"I have never come across a framework used to determine the ethical concerns raised in agile-quantum software development. I argue that regulatory bodies shall step forward to consider the ethical risks of quantum theory applications and invest in a coordinated approach to govern the potential risks effectively"* [P18], Business intelligence engineer.

The interviewee's statement is supported by Buchholz and Ammanath [37] broadly in the quantum computing domain by mentioning that the ethical concerns raised by QC are still on the horizon, and not possible to tackle them immediately. However, it's time that industrial and government stakeholders realize the trigger events and start thinking to develop the strategies [37]. The existing classical frameworks could be the best fit to start building ethical guidelines and models in the QC domain.

- **Agile-quantum ecosystem**

QC ecosystem is growing beyond advancements in hardware, and people have started focusing on producing quantum software and business partnerships [38]. Reaping the business interests of quantum software enables teams to connect and collaborate from anywhere. However, developing sustainable collaboration between teams and businesses is challenging as collaboration sits at the intersection of computer science, quantum physics and applied mathematics [39]. The collaboration could be complemented by presenting an ecosystem as highlighted by an interview participant more specific in agile based quantum software development:

👉 *"Agile activities more focus on collaboration and communication across teams, customers and other stakeholders. On the other hand, quantum software development is a paradigm shift and the agile teams must need an ecosystem that help them to seamlessly collaborator with customers and stakeholders from other domains (e.g. quantum physics)"* [P11], Agile coach.

3) Quantum-aware tools and technologies: A wide range of tools are available to support quantum software development activities. For example, Qiskit [2] is a known open-source tool available as a service on IBM cloud for developing quantum software at module, circuit and algorithm levels. Similarly, Microsoft Azure Q# and the Quantum Development Kit offers tools as a service for quantum software development i.e., designing the quantum algorithm, optimizing the solutions, and applying the optimized solutions across the Azure platform to see the real-world impacts [4]. Moreover, Khan et al. [5] thoroughly discussed the existing toolchain support for architecting a quantum software system. However, the interview participants highlighted its lack for customizing agile practices to develop quantum software.

👉*"Presently, it's exceedingly difficult to develop quantum software that offers commercial-level benefits which lift economies. We need to present novel tools and techniques that support hybrid quantum-classical development, for instance, using the traditional (classical) agile practices to support quantum software development activities"* [P6], Quantum network engineer.

The quantum-aware tools and technologies category emerged from two underlying concepts: classic-quantum tailoring and continuous SE infrastructure.

- **Classic-quantum tailoring**

It is commonly acknowledged that one-size-fits-all application of methods and processes is fallacious [9]. In line with it, classical techniques must be tailored to achieve the optimum effect in the quantum software domain [14] [1]. The tailoring could be actualized using domain-specific tools and technologies [9]. The need for such tools and technologies continue as a persistent theme in quantum software development [5] [14].

In this study, classical-quantum tailoring is particularly related to customizing the traditional agile practices for quantum software. However, the interview participants highlighted the lack of tools and techniques to support the tailoring process, for example:

👉*"The conventional agile practices must be used to develop the quantum software system. However, designing a compatible, customizable, and tailorable agile approach that defines the best practices for quantum software required a set of tools and technologies, and presently, the lack of such tools seems a major challenge"* [P8], Team lead.

- **Continuous SE infrastructure**

Agile-driven quantum software development infrastructure is required for implementing a deployment pipeline and monitoring dashboard to support the development experience [7], similar to modern-day continuous software engineering [9]. This pipeline will collect data from the development activities and visualize the practitioner's experiences and customer feedback [7]. However, such infrastructure is not yet implemented on a large scale, as mentioned by the interview participant:

👉*"Agile-based quantum software development includes various phases which consist of multiple components and are often difficult to implement manually. It require infrastructure support complemented by open source tools to automate and enables continuous software engineering practices, e.g., continuous integration"* [P12], Software designer.

4) Standards and Specifications: Standards and specifications establish a common agreement for engineering criteria, terms, principles, items, practices, and processes [40]. In this study, standardization and specifications emerged as the core category of challenging factors based on two elementary concepts: process standardization and optimum documentation.

- **Process standardization**

Standardization plays an important role in portraying and strengthening the QC technologies [41]. In line with it, the interview respondents perceived standardization as a roadmap for considering agile practices to develop a quantum software system. For instance, one of the interview participants described it as:

👉*"Using agile practices to develop a quantum software could be crucial to encapsulate the agile manifesto and develop a quality product. It is important to provide a roadmap (standards) that define rules, guidelines or characteristics for activities used to bring agile practices in quantum software development domain"* [P8], Team lead.

- **Optimum documentation**

In agile software development, optimum documentation provides the best possible process and the adaptability of changes across the development life cycle [42] [43]. Simple and lightweight documentation in agile is "living," and it needs to be collaboratively maintained by the whole team [44]. Similar understanding (optimum documentation) is supported by the interview participants for agile-driven quantum software development. For instance, one of the interviewees mentioned:

👉*"Agile focus on minimizing waste; logically considering it, the project documentation is unnecessary and exhaustive activity. However, it doesn't mean documentation should be neglected and thrown away (particularly for hybrid agile-quantum activities). The new agile quantum software development genre must require documentation and guidelines at some extent to ensure the success of the process, product and project"* [P10], Scrum master.

VI. Discussion

We now summarise the study's core findings (RQ1, RQ2), implications for research and practice and threats to the validity.

A. Agile for Quantum Software Development (RQ1)

Applying agile practices enables QSE teams to tackle quantum software development endeavours by relying on iterative and incremental development to deliver quantum software. For instance, an increment of transforming the quantum domain knowledge (e.g., secure transmission of key in a quantum network) to a quantum algorithm (implementing Shor's algorithm for decryption) emphasizes a piecemeal development and quantum-specific knowledge being translated into implementation details [1] [5].

The overwhelming majority of the interviewed practitioners (i.e., 17/26) endorsed that the application of agile practices enables development teams to undertake quantum software development tasks more effectively and efficiently. Specifically, the practitioners rationalized their endorsement based on the view that QSE is yet another genre of SE based on quantum mechanics, therefore; the principle and practices of conventional agile practices can be applied in a quantum software development lifecycle [15] [17] [7]. Moreover, the advocacy of agile for quantum software highlighted a multitude of reasons by practitioners such as model-driven QSE, increments to map various quantum-centric models (e.g., domain, architecture, code, simulation model), and quantum system co-design as shown in Figure 4. On the contrary, approximately 1/4 (6/21) of the surveyed practitioners disagreed with agile-driven quantum software development sharing their skepticism as immature tools and technologies [1], lack of professional expertise such as quantum software modeling [45], and the needs of a well-curated architecture [12], among the reasons for the rejection (see Figure 4). Three participants remained neutral, rationalising the reasons that the knowledge of quantum mechanics or lack of it may be determinantal factors for developing a team to opt for agile practices or not while undertaking quantum software projects (see Figure 4).

We noticed that the disagreement or neutral factors raised by the interview participants were later identified as the core categories of challenges. For example, the participants rejected using agile practices for developing quantum software because of immature tools and technologies, which is directly related to the *Quantum-aware tools and technologies* category of challenges as shown in Figure 4. It reveal that most of the disagreement and natural factors are emerged as the core challenging categories (see Figure 4).

B. Agile Practices Challenges in Quantum Software Development (RQ2)

Agile SE to develop quantum software involves a multitude of challenges that vary from lack of knowledge and skills (people-centric), quantum-specific tools and development platforms (technology-centric), sustainable scaling (societal-centric) along with models and documentation (process-centric) (see Figure 4).

Quantum-centric awareness and knowledge: The practitioners' responses reveal that lack of quantum-awareness in agile teams, i.e., limited or non-existent knowledge, skills, teamwork and education, are among the prime challenges for seamless and at a scale adoption of agile practices in the quantum software development context [15] [33]. The study analysis suggested that practitioners explicitly referred to knowledge as quantum mechanics know-how, such as quantum entanglement and quantum superposition etc. In quantum software development, agile teams are perceived to work better when built-around practitioners who could be quantum-aware (team dynamics); this means that instead of a traditional project manager or scrum master a quantum domain engineer is perceived as the right professional to lead the software development tasks in a bottom-up approach from domain knowledge to quantum modelling, implementation, and execution/simulation [12]. Lack of education and knowledge are obvious challenges, however, it may be seen as an opportunity to design a course curriculum (degree level) that can help preparing the skilled workforce with extensive knowledge and understanding of agile-driven quantum software development. Such courses may help software engineers and developers to leverage their existing skill set to work on quantum software development [33] [31].

Ethically aligned agile-quantum ecosystem: In addition to the technical, socio-technical aspects faced by the quantum software team is perceived as a challenge with far-reaching consequences of what is referred to as the ignorance of ethics and the lack of agile-quantum ecosystem (section V-B2). With the increased adoption of QC technologies, ongoing debates both in the academic and industry advocate for framework and guidelines for ethically-aligned quantum computing technologies [36]. From an agile perspective, practitioners did emphasize that quantum-age software needs an ecosystem that provides services and processes for seamlessly bridging the gap between agile teams and stakeholders from other domains e.g., quantum physics and applied mathematics. Similarly, the computational supremacy of QC technologies in areas like security (e.g., data en-/decryption) and bio-inspired computing (e.g., gene editing) should adhere to development practices, audits, and metrics that ensure quantum software potential does not undermine social norms and values.

Lack of maturity and openness in tools: Software tools such as model-to-code translators and test case generators are used to automate and customize developmental lifecycle tasks that may be time-consuming and error-prone [1] [14]. One of such tasks is the conventional agile and Quantum tailoring [15] that lacks the tool support, for example source code execution via classical (binary gates) and integration through quantum (Qgates). Another hindrance relates to the lack of infrastructure to visualize and automate the agile-driven continuous QSE activities. The availability of open-source tools can ensure community-wide initiatives to develop such infrastructure that can be customized, readily evolved, and widely adopted rather than individual solutions that may apply to a limited set of problems [5].

Standards and optimal documentation: Agile based SE is seen as a light and adaptive mechanism to develop and deliver software-intensive systems and products [8]. In line with the agile manifestation for minimal documentation [8], quantum software standards can be treated as necessary instruments to define agile-aligned quantum software roles and essential documentation to guide the team. For example, a readiness model which provide a set of best practices and guidelines for classic-quantum transformation or hybridization (classic-quantum co-design) [46]. In line with [46], standards can act as sufficient documentation and specifications that assist in moving from ad hoc agile-driven quantum software development practices to more mature processes.

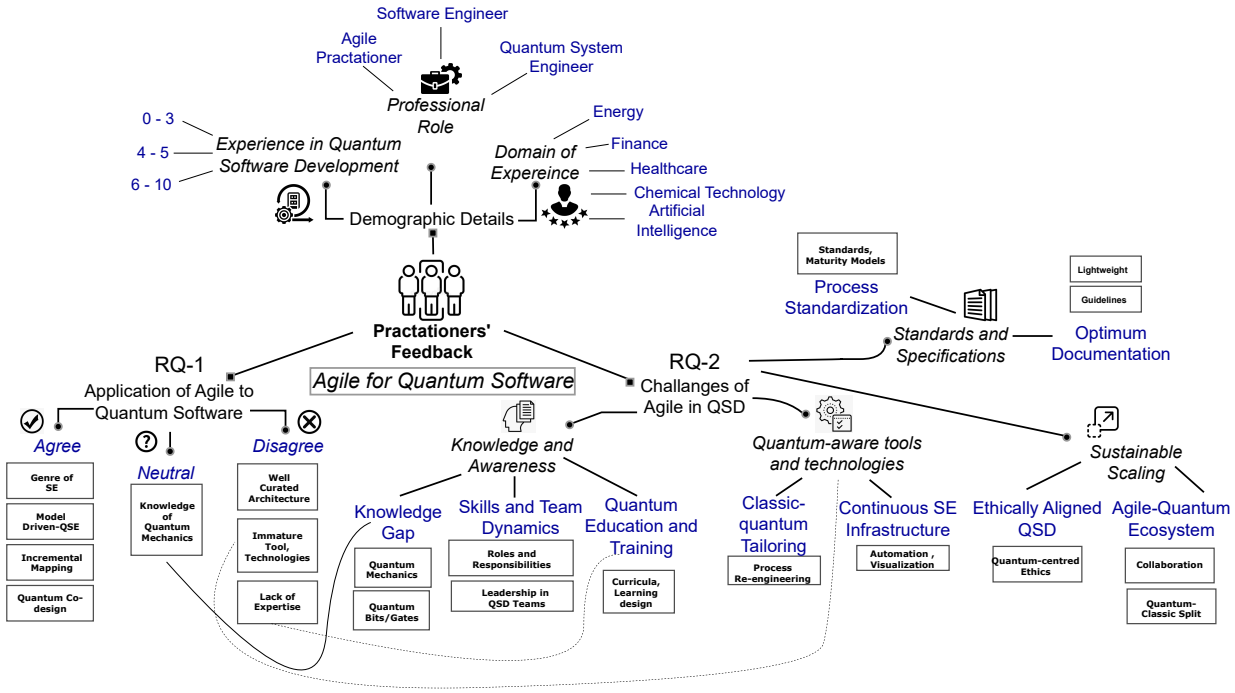

Fig. 4: Discussion

C. Implications for Research and Practice

Some implications arise from this research as listed in the following.

Implications for research: The results of the study complement the recently emerging streams of research on quantum software engineering, particularly add practitioner-centric view via empirically-based findings to agile-driven quantum software development. Researchers can utilize the results to quickly lookup the streamlined challenges (i.e., new hypotheses to be tested) regarding the knowledge, skills, tools, technologies, ethics, sustainability, and process aspects of quantum software engineering. The study's findings, on the other hand, give quick access to the body of knowledge based on the practitioner's understanding of agile-driven quantum software development.

Implications for practice: The study delivers inferential empirical findings to industrial practitioners and/or stakeholders interested in the fine-grained analysis of agile practices for quantum software development. The overview and interpretations of adaptability and potential challenges of agile-driven quantum software development provided a roadmap enlightening software practitioners interested in using agile practices for developing quantum software. In particular, the practitioners perspectives could help developers to engineer the next generation of quantum software.

D. Threats to validity

Various potential threats could affect the validity of the study findings. The relevant threats are broadly categorized across internal, construct, and external validity [47].

Internal validity: Internal validity is the extent to which particular factors affect the methodological rigor. In this study, the first threat to internal validity is the interview participant's understanding of the interview questions. This threat has been mitigated by conducting pilot interviews to ensure the understandability and readability of the interview questions (see Section III-B). The interpersonal bias in the data collection process may threaten the internal validity of study findings. We mitigated this threat by organizing regular consent meetings between all the authors for interview instrument development, feedback and GT data analysis (i.e. coding, concept development, categorization).

Construct validity: Construct validity is the extent to which the study constructs are well-defined and interpreted. In this study, the interview participants' perceptions of agile adaptability and relevant challenges are the core constructs. The verifiability of constructs is a known limitation of GT studies. It could be inferred from the research method efficacy and from the evidence that the study findings are presented based on the data collected using the selected research method [48]. Therefore, we explicitly discussed the step-by-step process of the research method (see Section IV), the defined categories supported with quotes from the interview participants, and our observations in section V-B. It exhibits how the systematically defined research protocol and reported findings (RQ1, RQ2) support the verifiability of the study constructs.

External validity: External validity refers to broadly generalizing the study findings in other contexts. In this study, the sample size and sampling approach may not provide a strong foundation to generalize the findings. However, it is known

that QSE, particularly using agile practices for developing quantum software, are new research areas and not in practice at a high level. We tried to mitigate this threat by using all possible sources (see Section III-B) to approach the potential population. Moreover, we plan to extend this study in the future with a large data sample from multiple data sources (i.e. mining the Q&A platforms, conducting industrial surveys and interviews) (see Section VIII).

VII. Related Work

We discuss the most relevant existing work, reviewing state-of-the-art on processes and development lifecycle(s), for quantum software systems [1]. A conclusive summary at the end reiterates the scope and contributions of the proposed research.

A. Process-centred Engineering of Quantum Software

Quantum software engineering – most recent genre of software engineering – aims to apply existing processes, practices, methods, tools, and techniques to design and develop quantum age software systems and applications effectively and efficiently [6] [17]. The existing body of SE knowledge provides foundations for quantum software development [1], however; unique characteristics of quantum software (e.g., quantum system co-design, quantum source coding) and quantum specific professional expertise (e.g., quantum domain engineer, quantum source coder) requires traditional SE methods to be tailored to address QSE challenges [12] [5]. To investigate quantum specific software architecting and development, recently conducted systematic reviews aim to establish empirical foundations to design [5] and develop [49] quantum software by streamlining the required process, existing patterns, and ideal tool chain(s) that can empower the roles of traditional software developers as quantum software architect and developers. QSE as a discipline is still in its infancy, however; community wide initiatives are gaining momentum, for example with dedicated conference[3], workshop (Q-SE)[4], literature reviews [49] [1] [50] [5], and repositories mining [18] efforts with a common denominator of using best practices and reusable knowledge (processes, patterns, reference architectures, etc.) to effectively and efficiently develop quantum software [6]

B. Iterative Development of Quantum Software

To synergise an iterative development with quantum software engineering process, Khan et al. [7] present the vision for continuous iterative development and delivery of quantum software. However, iterative enabled quantum software development is still in its infancy and demands novel domain specific techniques and processes as suggested in [7] [15] [14]. Considering the existing QSE methods and techniques [14] [45] [19] [51] there is still no or even consensus on agility in quantum software. The adoption of agile for quantum software development can help practitioners to iteratively experiment new ideas, and improve performance- all in parallel [7] [15]. In line with this, Gonzalez and Paradela [15] conceptualised a hybrid project management framework (unifying classical and quantum SE) for agile and incremental management of software that can be adapted to the needs of quantum-age computing. However, the study do not presented any empirical evidence to actualise the proposed framework. Similarly, using the contemporary SE practices, Weder et al. [14] introduced a quantum-based software development process that comprises of eight classical phases with induction of a new phase called quantum-classical splitting. The quantum-classical splitting phase is tailored for quantum software systems and it distinguishes between parts of the quantum software that need to be executed on a quantum computer and that ones that can be executed on classical computers.

C. Comparative Analysis

In this research, we argue that prior to adopting any specific processes and/or reference models [14] [15] for agile-driven QSE, there is a need to analyse practitioners perceptions, professional practices, and empirically grounded findings on the potential and challenges of adopting agile SE in the context of quantum software. Our research aims to complement the existing efforts, such as establishing models [45], architectures [5], and processes [16] with an overall aim to systemize the development of quantum software. This research follows-up on our previous works (i.e., the cases of architecting [12] [5] and iterating [7] quantum software development) with a practitioner's interview on the potential of adopting agile practices for quantum software development.

VIII. Conclusions and Future Work

Quantum software engineering leverages the principle and practices of classical SE - empowering the role of architects and developers to deliver quantum-age software applications providing impetus to QC. We conducted this study, rooted in empirical analysis and grounded theory, to understand practitioners' perspectives on the potentials and pitfalls of agile practices in quantum software development endeavours. We engaged a total of 26 practitioners from 10 countries generally classified as software engineers, agile experts, and quantum software developers (see Figure 2) with an overwhelming majority agreeing (n=17), approximately one fourth disagreeing (n=6) and relative minority (n=3) not sure on applying agile practices to quantum software development, answering RQ1.

To gain better understanding and fine-granular analysis of the disagreements, we presented open-ended questions on challenges, i.e., hindrances perceived by practitioners on agile-oriented quantum software development, answering RQ2. After collecting and synthesizing the responses, we identified 4 categories organized into 9 sub-categories (concepts) perceived as challenges by practitioners (see Figure 4). These four categories include: *1) Knowledge and awareness*, which provide practitioners insights and understanding of challenges that could hinder the adaptability of agile practices for developing quantum software. For instance, the existing knowledge gap

[3] https://conferences.computer.org/qsw/2022/
[4] https://conf.researchr.org/home/q-se-2022

between classical and quantum software development, different mindsets (classical->quantum), and lack of agile-quantum specific education and guidelines. *2) Sustainable scaling*, encapsulates the challenges related to harming software sustainability practices. QSE is a paradigm shift, and using agile to develop quantum software is more likely to threaten the existing ethical protections (e.g., transparency, accountability) and the agile-quantum ecosystem. *3) Quantum-aware tools and technologies*, category consist of the challenges related to the immature tool support and lack of infrastructure to tailor, customize, automate, and configure the agile practices for developing quantum software. *4) Standards and specifications*, category highlights the lack of common rules and principles for agile-quantum software scenario. From the perspective of participants, to adopt agile for quantum software the necessary standards, models, and documentation should be provided.

Our future work is to extend this study by identifying the challenges from social coding platforms in open-source quantum projects. To achieve this, we plan to *1)* mine publicly available developers' discussions and code repositories (i.e., Stack Overflow, GitHub) to explore solutions that the QC community employs for the effective adoption of agile methods, *2)* map these solutions to the identified challenges with respect to causes, and 3) validate this mapping via a large sample quantitative survey and interviews of QSE practitioners.

References

[1] J. Zhao, "Quantum software engineering: Landscapes and horizons," *arXiv preprint arXiv:2007.07047*, 2020.

[2] IBM, "Qiskit." https://qiskit.org/, 2021.

[3] Google, "Cirq." https://quantumai.google/cirq/google/concepts, 2022.

[4] Microsoft, "Q# and the quantum development kit." https://azure.microsoft.com/en-us/resources/development-kit/quantum-computing/#overview, 2021.

[5] A. A. Khan, A. Ahmad, M. Waseem, P. Liang, M. Fahmideh, T. Mikkonen, and P. Abrahamsson, "Software architecture for quantum computing systems-a systematic review," *arXiv preprint arXiv:2202.05505*, 2022.

[6] S. Ali, T. Yue, and R. Abreu, "When software engineering meets quantum computing," *Communications of the ACM*, vol. 65, no. 4, pp. 84–88, 2022.

[7] A. A. Khan, M. Fahmideh, A. Ahmad, M. Waseem, M. Niazi, V. Lahtinen, and T. Mikkonen, "Embracing iterations in quantum software: A vision," 2022.

[8] K. Beck, M. Beedle, A. Van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, *et al.*, "Manifesto for agile software development," 2001.

[9] B. Fitzgerald, G. Hartnett, and K. Conboy, "Customising agile methods to software practices at intel shannon," *European Journal of Information Systems*, vol. 15, no. 2, pp. 200–213, 2006.

[10] B. G. Glaser and A. L. Strauss, *The discovery of grounded theory: Strategies for qualitative research*. Routledge, 2017.

[11] E. G. Rieffel and W. H. Polak, *Quantum computing: A gentle introduction*. MIT Press, 2011.

[12] A. Ahmad, A. A. Khan, M. Waseem, M. Fahmideh, and T. Mikkonen, "Towards process centered architecting for quantum software systems," in *2022 IEEE International Conference on Quantum Software (QSW)*, pp. 26–31, IEEE, 2022.

[13] ISO/IEC/IEEE, "Iso/iec/ieee 12207:2017 - systems and software engineering - software life cycle processes." https://www.iso.org/standard/63712.html, 2017.

[14] B. Weder, J. Barzen, F. Leymann, and D. Vietz, "Quantum software development lifecycle," *arXiv preprint arXiv:2106.09323*, 2021.

[15] G. J. Hernández González and C. A. Paradela, "Quantum agile development framework," in *International Conference on the Quality of Information and Communications Technology*, pp. 284–291, Springer, 2020.

[16] E. Moguel, J. Berrocal, J. García-Alonso, and J. M. Murillo, "A roadmap for quantum software engineering: Applying the lessons learned from the classics.," in *Q-SET@ QCE*, pp. 5–13, 2020.

[17] M. Piattini, M. Serrano, R. Perez-Castillo, G. Petersen, and J. L. Hevia, "Toward a quantum software engineering," *IT Professional*, vol. 23, no. 1, pp. 62–66, 2021.

[18] M. De Stefano, F. Pecorelli, D. Di Nucci, F. Palomba, and A. De Lucia, "Software engineering for quantum programming: How far are we?," *Journal of Systems and Software*, vol. 190, p. 111326, 2022.

[19] X. Wang, P. Arcaini, T. Yue, and S. Ali, "Generating failing test suites for quantum programs with search (hot off the press track at gecco 2022)," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 47–48, 2022.

[20] X. Wang, T. Yu, P. Arcaini, T. Yue, and S. Ali, "Mutation-based test generation for quantum programs with multi-objective search," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1345–1353, 2022.

[21] Y. Huang and M. Martonosi, "Statistical assertions for validating patterns and finding bugs in quantum programs," in *Proceedings of the 46th International Symposium on Computer Architecture*, pp. 541–553, 2019.

[22] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Information and software technology*, vol. 50, no. 9-10, pp. 833–859, 2008.

[23] T. Mikkonen, "Flow, intrinsic motivation, and developer experience in software engineering," *Agile Processes in Software Engineering and Extreme Programming*, p. 104, 2016.

[24] O. C. Robinson, "Sampling in interview-based qualitative research: A theoretical and practical guide," *Qualitative research in psychology*, vol. 11, no. 1, pp. 25–41, 2014.

[25] R. Hoda, J. Noble, and S. Marshall, "Organizing self-organizing teams," in *2010 ACM/IEEE 32nd international conference on software engineering*, vol. 1, pp. 285–294, IEEE, 2010.

[26] K. Madampe, R. Hoda, and J. Grundy, "A faceted taxonomy of requirements changes in agile contexts," *IEEE Transactions on Software Engineering*, 2021.

[27] Z. Masood, R. Hoda, and K. Blincoe, "Real world scrum a grounded theory of variations in practice," *IEEE Transactions on Software Engineering*, 2020.

[28] R. Hoda and J. Noble, "Becoming agile: a grounded theory of agile transitions in practice," in *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*, pp. 141–151, IEEE, 2017.

[29] G. Allan, "A critique of using grounded theory as a research method," *Electronic journal of business research methods*, vol. 2, no. 1, pp. pp1–10, 2003.

[30] S. Adolph, W. Hall, and P. Kruchten, "A methodological leg to stand on: lessons learned using grounded theory to study software development," in *Proceedings of the 2008 conference of the center for advanced studies on collaborative research: meeting of minds*, pp. 166–178, 2008.

[31] H. Li, F. Khomh, M. Openja, *et al.*, "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)*, pp. 343–354, IEEE, 2021.

[32] I. D. Inc, "Agile team dynamics." https://www.intelliware.com/agile-team-dynamics/, 2014.

[33] G. Peterssen, "Quantum technology impact: The necessary workforce for developing quantum software.," in *QANSWER*, pp. 6–22, 2020.

[34] D. Carberry, A. Nourbakhsh, J. Karon, M. N. Jones, M. Jadidi, K. Shahriari, C. Beenfeldt, M. P. Andersson, and S. S. Mansouri, "Building knowledge capacity for quantum computing in engineering education," in *Computer Aided Chemical Engineering*, vol. 50, pp. 2065–2070, Elsevier, 2021.

[35] S. Hunter, M. Pijselman, P. Clinton-Tarestad, H. Lewis, and M. Bell, "Before a quantum transition, what sustainability risks and opportunities should organisations recognise, and prepare for?." https://www.ey.com/en_uk/emerging-technologies/how-can-your-quantum-vision-be-transformed-into-a-sustainable-reality, 2022.

[36] K. Saurabh and V. Rustagi, "Ethical and sustainable quantum computing: Conceptual model and implications," *Journal of Contemporary Issues in Business and Government Vol*, vol. 28, no. 01, 2022.

[37] S. Buchholz and B. Ammanath, "Quantum computing may create ethical risks for businesses. it's time to prepare." https://www2.deloitte.com/uk/en/insights/topics/cyber-risk/quantum-computing-ethics-risks.html, 2022.

[38] V. Combs, "Quantum computing ecosystem expands in all directions." https://www.techrepublic.com/article/quantum-computing-ecosystem-expands-in-all-directions/, 2022.
[39] A. Branczyk, "What skills do i need to work at a quantum tech start-up?." https://www.linkedin.com/pulse/what-skills-do-i-need-work-quantum-tech-start-up-aggie-branczyk-/, 2021.
[40] AcqNotes, "Standardization." https://acqnotes.com/acqnote/tasks/standardization-requirements, 2021.
[41] A. Nikitin, "Standards for quantum technology." https://delft-circuits.com/standards-for-quantum-technology/, 2021.
[42] B. Linders, "Documentation in agile: How much and when to write it?." https://www.infoq.com/news/2014/01/documentation-agile-how-much/, 2014.
[43] R. Anwar, "Agile documentation: Best practices." https://delft-circuits.com/standards-for-quantum-technology/, 2016.
[44] Nuclino, "Agile development methodology and documentation to document or not to document?." https://www.nuclino.com/articles/agile-documentation, 2016.
[45] F. Gemeinhardt, A. Garmendia, and M. Wimmer, "Towards model-driven quantum software engineering," in *2021 IEEE/ACM 2nd International Workshop on Quantum Software Engineering (Q-SE)*, pp. 13–15, IEEE, 2021.
[46] M. A. Akbar, S. Rafi, and A. A. Khan, "Classical to quantum software migration journey begins: A conceptual readiness model," *arXiv preprint arXiv:2209.05105*, 2022.
[47] X. Zhou, Y. Jin, H. Zhang, S. Li, and X. Huang, "A map of threats to validity of systematic literature reviews in software engineering," in *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*, pp. 153–160, IEEE, 2016.
[48] R. Hoda, J. Noble, and S. Marshall, "Developing a grounded theory to explain the practices of self-organizing agile teams," *Empirical Software Engineering*, vol. 17, no. 6, pp. 609–639, 2012.
[49] B. Heim, M. Soeken, S. Marshall, C. Granade, M. Roetteler, A. Geller, M. Troyer, and K. Svore, "Quantum programming languages," *Nature Reviews Physics*, vol. 2, no. 12, pp. 709–722, 2020.
[50] S. S. Gill, A. Kumar, H. Singh, M. Singh, K. Kaur, M. Usman, and R. Buyya, "Quantum computing: A taxonomy, systematic review and future directions," *Software: Practice and Experience*, vol. 52, no. 1, pp. 66–114, 2022.
[51] S. Ali and T. Yue, "Modeling quantum programs: Challenges, initial results, and research directions," in *Proceedings of the 1st ACM SIGSOFT International Workshop on Architectures and Paradigms for Engineering Quantum Software*, pp. 14–21, 2020.