Agile meets quantum: a novel genetic algorithm model for predicting the success of quantum software development project

Arif Ali Khan[1] · Muhammad Azeem Akbar[2] · Valtteri Lahtinen[3] ·
Marko Paavola[4] · Mahmood Niazi[5,6] · Mohammed Naif Alatawi[7] ·
Shoayee Dlaim Alotaibi[8]

Abstract

Quantum software systems represent a new realm in software engineering, utilizing quantum bits (Qubits) and quantum gates (Qgates) to solve the complex problems more efficiently than classical counterparts. Agile software development approaches are considered to address many inherent challenges in quantum software development, but their effective integration remains unexplored. This study investigates key causes of challenges that could hinders the adoption of traditional agile approaches in quantum software projects and develop an Agile-Quantum Software Project Success Prediction Model (AQSSPM). Firstly, we identified 19 causes of challenging factors discussed in our previous study, which are potentially impacting agile-quantum project success. Secondly, a survey was conducted to collect expert opinions on these causes and applied Genetic Algorithm (GA) with Naive Bayes Classifier (NBC) and Logistic Regression (LR) to develop the AQSSPM. Utilizing GA with NBC, project success probability improved from 53.17 to 99.68%, with cost reductions from 0.463 to 0.403%. Similarly, GA with LR increased success rates from 55.52 to 98.99%, and costs decreased from 0.496 to 0.409% after 100 iterations. Both methods result showed a strong positive correlation (rs = 0.955) in causes ranking, with no significant difference between them ($t = 1.195$, $p = 0.240 > 0.05$). The AQSSPM highlights critical focus areas for efficiently and successfully implementing agile-quantum projects considering the cost factor of a particular project.

Keywords Agile approaches · Quantum software development · Prediction model

1 Introduction

Quantum Computing (QC) has emerged as a significant area of interest for industrial practitioners and academic researchers worldwide. Its potential to bring transformative changes across various industries is undeniable (Zhao 2007). This potential is evident in the investments and efforts of technology leaders such as IBM (2021), Google (2022), and Microsoft (2021), who are actively working to harness QC for solving complex computational challenges. However, as QC machines operate on the principles of quantum mechanics, the development of software applications for them presents unique challenges (Akbar et al. 2211). Literature suggests that the development process for QC applications closely mirrors traditional software development (Zhao 2007), underscoring the importance of a well-structured engineering process to address the complexities of QC (Zhao 2007). Therefore, refined Software Engineering (SE) methodologies are necessary to fully realize the benefits of QC (Zhao 2007; Akbar et al. 2211; Khan et al. 2023).

A key component in operationalizing QC systems is quantum software (Ali et al. 2022). This software, requiring comprehensive stack support and innovative tools and techniques, demands specific processes and methods tailored to create software systems based on quantum mechanics principles (Ali et al. 2022). However, the field of Quantum Software Engineering (QSE) is still evolving. Currently, it primarily relies on hybrid (quantum and classical) processes, tools, and methods, necessitating careful coordination between classical and quantum systems. In this regard, we previously conducted an empirical study introducing the idea of utilizing traditional (classical) iterative and agile methodologies for developing quantum software (Khan, et al. 2023, 2022). These agile methods, which promote team collaboration, quick iterations, and continuous delivery, could significantly benefit QSE activities (Khan, et al. 2023; Beck, et al. 2001; Fitzgerald et al. 2006). Nevertheless, more evidence is needed to directly connect agile methods to quantum software development.

Drawing from our previous research, where we interviewed 26 software professionals from 10 different countries (Khan, et al. 2023), we found that many view existing traditional agile methods as well-suited for quantum software development. They also identified potential challenges perceived to hinder scaling traditional agile methods in the quantum domain. These challenges fall into four main categories: (1) *Knowledge and awareness*, offering practitioners a deeper understanding of potential changes that might impede the successful adoption of agile methodologies in quantum software development; (2) *Sustainable scaling*, involving challenges related to scaling software sustainably and ethically; (3) *Quantum-aware tools and technologies*, encompassing challenges related to technological infrastructure that could support customizing, automating, and tailoring agile practices for developing quantum software; and (4) *Standards and specifications*, providing insights into challenges related to the lack of standards, frameworks, and common rules for adopting existing agile approaches in the quantum software domain.

With these findings, this paper aims to extend the previous study (Khan, et al. 2023) by exploring more in-depth the causes behind these challenges and

providing effective best practices to mitigate them. Based on the study results, we will introduce a model to predict the costs and efforts required for scaling agile methods in the quantum software development domain. This model utilizes a nature-inspired optimization algorithm, the Genetic Algorithm (GA) (Mirjalili and Mirjalili 2019), to evaluate the likelihood of success in terms of cost. Within the GA, we designed a measure called the fitness function, aiming for the highest success rate of agile quantum software projects while considering costs. We tested this optimization model using two prediction models: the Naive Bayes Classifier (NBC) and Logistic Regression (LR). Through this comprehensive approach, we aim to provide a roadmap for both researchers and practitioners in the quantum software development field, emphasizing the integration of agile practices.

2 Background

In the rapidly evolving world of technological advancements, QC has emerged as a pivotal innovation (Zhao 2007). Based on the principles of quantum mechanics, such as quantum superposition and entanglement, QC platforms offer a new computational paradigm. Unlike traditional binary digits, which strictly adhere to a 0 or 1 state, QC operates based on Qubits that can simultaneously exist in a superposition of both states, represented as I0 and I1 (Zhao 2007; Rieffel and Polak 2011). This shift from classical bits to Qubits signifies a profound transformation in the landscape of computing, pushing the boundaries of what is achievable with computational processes (Zhao 2007).

To navigate this quantum realm effectively, software developers are required to be familiar with quantum-age algorithms and programming languages like Microsoft Q# (2021), IBM Qiskit (2021), and Google Cirq (2022). However, there is a lack of a holistic engineering approach integral for quantum software development, e.g., quantum software requirements engineering, quantum software design, maintenance, evolution, and information simulation (Khan, et al. 2022; Ahmad et al. 2022). This gap underscores the importance of Quantum Software Engineering (QSE), as highlighted in a national agenda for software engineering research and development presented by SEI-Carnegie Mellon University (Carleton et al. 2022): "*the development of Quantum Computing Software Systems is considered a pivotal future focus area. The objectives of this research area are initially focused on facilitating the development of quantum software for existing quantum computers and then enable increasing abstraction as larger, fully fault-tolerant quantum computing systems become available. A key challenge is to eventually, fully integrate these types of systems into a unified classical and quantum software development life cycle.*" As the most novel field, QSE seeks to leverage the best practices from traditional software engineering, such as robust engineering processes, architectures, and patterns, facilitating the development of quantum-intensive software solutions. There is no existing quantum-specific process model for QSE available, except for a few research studies which focused on defining the fundamental engineering activities for a structured QSE approach (Khan, et al. 2022; Ahmad et al. 2022; Weder et al. 2022).

In the seek for structure and a robust QSE process, the agile approach from traditional software engineering presents itself as a promising contender. Tracking the evolution of traditional software programming, which began with a strong focus on hardware in the 1950s and later shifted to today's agile practices (Moguel et al. 2020; Piattini et al. 2021), opens the door for the novel QSE field to adopt traditional agile methods over time. However, given the unique challenges of quantum software development, such as qubit entanglement and superposition, it is increasingly compelling to examine the suitability of traditional agile approaches in line with the characteristics of quantum software development (Stefano et al. 2022; Wang et al. 2021, 2022). For example, composing, designing, testing, and maintaining quantum programs would be challenging due to switching to an entirely different mindset, examining the superposition states of quantum variables, and interpreting multi-dimensional quantum states.

In traditional software development, the above-mentioned activities are addressed by using agile and iterative approaches. Agile methods, in contrast to traditional models such as Waterfall, are considered best suited to propose practices and principles such as active user involvement, iterative development, frequent releases, continuous integration, and deployment (Beck, et al. 2001). However, the QSE field is still in its infancy, and the process of creating quantum software can be made simpler by embracing agile methods, as recommended by Piattini et al. (Piattini et al. 2021) and Khan et al. (Khan, et al. 2023).

Following this, we previously conducted an empirical study to evaluate the suitability of the traditional agile approaches and identify the challenges that could be potential barriers for agile-based quantum software development (Khan, et al. 2023). In our prior study, we reached the conclusion that agile approaches are the preferred choice for developing quantum software. The study also brought to light a range of challenges that could be potential barriers to adopting agile practices. Based on our earlier work, this research aims to investigate deeper into the causes of these challenges and best practices in-depth to formulate a predictive model that will estimate the chances of success when traditional agile techniques are used in quantum software projects.

No previous research has been conducted on developing a prediction model to aid decision-making about the use of agile practices in the quantum software domain. We have set out to address this gap by proposing a model that we name AQSSPM. This proposed model will serve as a tool for organizations that are considering the adoption of agile methods for quantum software development, highlighting the most significant key areas that can predict the success of their endeavors in this new and complex field.

For the creation of this model, we have incorporated not just the findings from our previous studies (Khan, et al. 2023), but also an expanded investigation into the causes behind these challenges and the most effective practices to address them. An industrial survey was conducted to gather practitioners' views on the challenges, causes, and best practices in quantum software development as identified in our previous research. The feedback obtained from this survey has been crucial in applying genetic algorithms to develop the proposed model—AQSSPM.

3 Research methodology

In this study we used the survey questionnaire approach to collect the data from industrial practitioners to evaluate the causes of the identified challenges. Based on the collected data, we trained the GA with NBC and LR to estimate the success probability of agile-quantum software projects. The details are provided in the following sections.

3.1 Designing the questionnaire

To collect the training data, we initiated data collection process through unstructured interviews with 13 experts in the domains of quantum computing and agile software development. These conversations, facilitated via digital meeting platforms like Google Meet and Zoom, spanned an average of 25 min each. Building upon the insights derived from these interactions, we constructed a detailed, closed-ended survey. The questionnaire survey allowed us to pinpoint 19 causes contributing to the challenges faced in agile-quantum software development. We used the questionnaire survey methodology to collect the data from a broad and varied group, working in agile-quantum paradigm.

The survey was divided into two sections. The first section describes the demographic details of participants, while the subsequent part contains closed-ended questions centered around the core variables (causes). To assess the feedback, we used a 9-point likert scale, range from extremely low (EL) to extremely high (EH). Using surveys for data collection, especially when observational techniques fall short, is a best suited approach, as evidenced in different other studies (What is sustainable software development 2023; Khan et al. 2017).

3.2 Pilot assessment of the questionnaire

To enhance the accuracy and consistency of our survey insturment, we conducted a pilot assessment of the questionnaire. As highlighted in existing studies, such pre-tests are important to refine the survey structure and ensure relevant feedback (Khan et al. 2021; Failed 2023). For this early assessment, we involved ten specialists, of which six participated in our prior informal discussions and four were new contributors. These specialists were associated with research institutions, including the University of Oulu in Finland, Norwegian University of Science and Technology in Norway, and Aligarh Muslim University in India.

Based on the feedback received, we updated the questionnaire structure by classifying it into three sections: demographic details, specific questions about identified causes of agile-quantum project, and cost-related data for tackling the identified causes. We also adjusted the wording of the survey items for clarity. One expert suggested presenting the questions in a tabular format, and other

asked to improve the understandability of each question. The refined question-naire can be accessed at: https://tinyurl.com/43r7uxx6.

3.3 Process of predictive model development

The presented study seeks to understand the impact of pinpointed causes by identifying the scale at which the project has the highest likelihood of success. This insight will guide software professionals to prioritize critical elements in the agile-quantum workflow, ensuring its optimal operation. To achieve this, we employed Genetic Algorithms (GA) (Mirjalili and Mirjalili 2019) aiming to optimize the chances of project success while restricting the related costs. GA works by generating factor states, determining success probabilities with predictive models, and simultaneously evaluating the balance between success likelihood and cost, thereby pinpointing feasible factor scales. The efficiency of GA hinges on three main components: (1) Predictive models based on Naive-Bayes and logistic regression, which inform GA of the success probability aligned with specific risk factor values; (2) The expense linked to each risk factor's scale value; (3) A function grounded in success probability and cost, guiding GA about the merit of a certain set of feature values. In this context, once the predictive models are adequately trained, they provide GA with the necessary probability, with our primary focus on the Naive Bayes algorithms. The cost metrics for different process areas, derived from expert opinions, are detailed in Table 1. Lastly, an efficiency function is crafted to meet the third component's needs, taking into account both success probability and cost.

3.4 Predictive models

In our research, we employed predictive models grounded in Naive Bayes and Logistic Regression methodologies. These models produce the likelihood of a class variable reaching a particular outcome, representing success or failure in our context.

3.5 Naive bayes classifier (NBC)

The NBC model estimates the probability of a specific result for a class variable, like success or failure. Bayesian Networks offer an array of classifiers rooted in probability (Kotsiantis et al. 2006). Notably, the Naive Bayes Classifier (NBC) is distinguished by its straightforwardness and efficacy (Berrar 2018).

The NBC presumes that, given the class, the predictors or attributes function independently. In this framework, every independent variable possesses a sole parent: the class or outcome variable. Because of its robust mathematical underpinning, the NBC algorithm is renowned for its rapidity, simplicity in execution, and adaptability to manage datasets with vast dimensions. Such effectiveness stems mainly from NBC approach of determining the probability for each attribute autonomously (Berrar 2018; Cerpa et al. 2016).

Table 1 Causes at different scales, along with associated costs

Scale	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19
EL	2	2	2	2	1	1	1	1	3	1	1	2	2	4	3	1	2	1	1
VL	3	2	2	2	2	2	1	1	3	2	2	3	3	3	3	1	2	4	2
L	3	2	3	3	2	2	2	2	4	2	3	3	3	2	4	2	4	3	2
SL	4	4	4	4	2	3	3	3	4	2	4	4	4	4	4	5	4	5	2
Neutral	5	5	4	4	4	4	5	4	5	5	5	5	5	4	5	4	5	5	4
SH	6	6	5	6	6	6	6	4	5	6	6	6	6	6	5	3	6	6	6
MH	6	6	6	6	6	6	6	5	6	6	7	7	7	7	6	5	7	7	8
VH	7	7	7	7	7	7	6	6	7	6	7	8	8	7	7	6	7	8	7
EH	8	8	8	8	7	7	7	8	8	7	8	9	9	8	8	8	8	9	7

Equation 1 depicts the computation of the maximum posterior probability for the target variable T in relation to the attributes F of the observation, grounded in Bayes' theorem:

$$Prob(T|F) = \frac{Prob(T) * Prob(F|T)}{Prob(F)} \tag{1}$$

NBC considers that all the elements of $F = \{f_1, f_2, \ldots, f_n\}$, given T, are conditionally independent, and therefore, the probability described in Eq. 1 can be computed according to Eq. 2.

$$Prob(T|F) = \frac{Prob(T) \prod_{i=1}^{n} Prob(f_i|T)}{Prob(F)} \tag{2}$$

Equation 2 can be rewritten as Eq. 3 in its extended form:

$$Prob(T|f_1, f_2, \ldots f_n) = \frac{Prob(T) * Prob(f_1|T) * Prob(f_2|T) \cdots * Prob(f_n|T)}{Prob(F)} \tag{3}$$

For categorization endeavors, Eq. 3 often suffices to pinpoint the likeliest state of the target variable based on a defined set of factors. Yet, in our study, we harness the Naive Bayes methodology, utilizing input values from diverse factors to forecast the project's success probability. After the model's training phase, it's equipped to gauge the potential for a positive result.

3.6 Logistic regression (LR)

In our study, we additionally utilize logistic regression (LR) as a forecasting tool, mainly to determine the likelihood of a binary outcome (Cerpa et al. 2016). This is perceived as an evolution of regression methods tailored for predicting continuous outcome variables. A drawback with conventional regression techniques is that they can produce predicted values for the outcome variable that fall outside the (0,1) range, with 0 representing a negative outcome (Failure, False, or No) and 1 a positive outcome (Success, True, or Yes) (Wolpert and Macready 1997). To address this challenge, logistic regression employs a logistic function, as presented in Eq. (4):

$$S(x) = \frac{1}{1 + e^{-x}} \tag{4}$$

Initially, a function is defined based on independent variables as follows:

$$func(F) = \beta_0 + \beta_1 f_1 + \ldots + \beta_n f_n \tag{5}$$

In Eq. (5), β_i denotes the significance of attribute fi. The core goal of the Logistic Regression (LR) algorithm is to determine the best-fit values for each β_i. Using Eq. (6), LR produces probabilistic estimates, categorizing the binary target variable T into either 1 or 0.

$$Prob(T = 1) = \frac{1}{1 + e^{-func(F)}} \tag{6}$$

Equation (7) can be used to calculate the probability ($T = = 0$):

$$\text{Prob}(T = 0) = 1 - \text{Prob}(T = 1) \tag{7}$$

The LR algorithm executes a series of procedures to refine the β_i values according to Eqs. (5) and (6) until a state of relative balance is achieved, where the β_i values remain largely consistent. At this stage, the LR algorithm integrates the attributes to optimize the likelihood of discerning the state of T in terms of probability. In our research context, we employed both NBC and LR models. These models intake the input values from diverse factors to forecast the likelihood of project success. Once honed, these models are adept at gauging the potential for a favorable result.

3.7 Optimization problem

This segment exmined the mathematical framing of the optimization challenge, paving the way for the implementation of the Genetic Algorithm (GA). The predictive models are primed to compute probability figures using the supplied data. Definitions for both probability and cost will be established, and from these metrics, an effectiveness function will be subsequently formulated.

3.8 Probability of success

Based on a given set of attributes, a project's probability of success can be expressed as follows:

$$\text{Prob(S)} = \text{p} \tag{8}$$

Prob (S) takes in a prospective solution S and evaluates its success probability p, ranging between 0 and 1. The set S can be articulated as follows:

$$S = \{s_1, s_2, \ldots, s_i, \ldots, s_n\} \tag{9}$$

In Eq. 9, the variable s_1 represents the scale of the ith factor, with n indicating the aggregate count of factors being examined. Here, s_i can take on values from 1 through 9, and n matches the count of these factors. Equation 10 showcases a particular example of the solution set S, denoted as S (n $=$ 19).

$$S' = \{6, 5, 4, 2, 3, 7, 1, 3, 8, 2, 6, 1, 9, 6, 8\}1 \tag{10}$$

In this case, S' represents a solution set in which the first factor has a scale value of 6, the second factor holds a value of 5, and so on. Introducing S' as an input into the model produces the associated probability, given that the predictive models have been previously trained.

3.9 Cost calculation

An important section of defining the problem is identifying the costs tied to each factor's scale. These costs have been determined manually by domain experts. The objective is to amplify the likelihood of project success while constraining the overall expenses. The variable c_{ij} represents the cost associated with the jth scale value of variable i. The cumulative cost for the solution S can be determined through Eq. 11.

$$C(S) = \sum_{i=1}^{n} c_{ij} \tag{11}$$

Table 1 outlines the expenses linked to different scales of factors. Whenever a fresh instance of S is produced, the overall project cost is ascertained based on Table 1 in conjunction with Eq. (11).

3.10 Efficacy

A project's effectiveness is determined by evaluating its likelihood of success alongside its associated cost, see Eq. 12. This challenge can be perceived as a dual-objective optimization task, striving to enhance the probability of project success while simultaneously curbing associated expenditures. To merge these two objectives into one optimization dilemma, an effectiveness function was crafted, as detailed below:

$$E = Prob - C \tag{12}$$

One practical approach to gauge the effectiveness of a particular instance S is by determining "the disparity between the success probability and its cost." This straightforward method integrates both parameters into a singular function. Yet, as illustrated in Eq. 8, the cost C often dominates, given that the probability Prob consistently lies between [0,1], whereas C might attain a peak value of max(C) when all factors are scaled at 9. To address this discrepancy, the normalized cost, as presented in Eq. 13, can be utilized.

$$norm(C) = \frac{C - \min(C)}{\max(C) - \min(C)} \tag{13}$$

In Eq. 13, C denotes the cost awaiting normalization, while min(C) and max(C) represent the minimum and maximum costs for a project, respectively. In the context of our problem, there are 19 causes (meaning, n = 19). As such, max (C) is 126, and min (C) stands at 19 (assuming all factors are scaled at 1). Utilizing Eq. 13, we confine the cost within the range (0,1), guaranteeing it doesn't solely influence Eq. 12. The consequent effectiveness of S is then determined using Eq. 14.

$$E(S) = Prob(S) - norm(C(S)) \tag{14}$$

It's pertinent to mention that various approaches exist to integrate the cost into the problem formulation. One approach is to treat both cost and success probability

as separate objectives, which would morph the present issue into a multi-objective optimization challenge. Another perspective would be to perceive the cost as a constraint rather than a component of the fitness function. In this scenario, the aim would be to identify a solution with the utmost success probability, given that the cost remains below a predetermined maximum allowable expense, represented as C_{max}. However, for its straightforwardness and lucidity, this research opted to use Eq. 14 as the guiding objective function.

3.11 Mathematical modeling of the optimization problem

After detailing the essential components of the problem in the previous sections, we now present the optimization challenge and its corresponding mathematical representation, which is to be maximized.

$$\text{Maximize } E(S) = \text{Prob}(S) - \text{norm}(C(S)) \tag{15}$$

where $S = \{s_1, s_2, \ldots, s_i, \ldots, s_n\}$. Given: $s_{min} <= s_i <= s_{max}$.

In the given equation, S_{max} and S_{min} denote the maximum and minimum scale values, respectively. From Eq. 15, it is evident that our goal is to identify an instance S that yields the optimal effectiveness value, which is influenced by a high probability of success paired with a minimized normalized cost.

3.12 Optimization problem, genetic algorithm, and its significance

While traditional solutions like exhaustive search can address optimization problems, they become impractical when dealing with extensive search spaces (Kumar et al. 2023). In our current context, with fourteen attributes, each potentially ranging from 1 to 9, we encounter over 22.8 trillion potential solutions ($9^{19} > 22.8$ trillion). Hence, meta-heuristic methods, like GA, are favored as they can deliver near-optimal results within acceptable timeframes. GA stands out as a prevalent meta-heuristic strategy for tackling optimization challenges, with numerous scholars employing it across various domains, especially in combinatorial optimization tasks. While other meta-heuristics exist, the No-Free-Lunch Theorem (Wolpert and Macready 1997) suggests that no singular metaheuristic outshines the rest, and overall, they yield comparable results. Numerous studies have showcased GA's application across diverse sectors for combinatorial optimization issues, consistently yielding encouraging outcomes (Komaki and Kayvanfar 2015; Hu et al. 2020; Mirjalili et al. 2014; Mahmoodabadi et al. 2013). Therefore, this research employs GA.

3.13 Genetic algorithm

The Genetic Algorithm (GA) is an evolutionary technique rooted in Charles Darwin's theory of natural selection and was formulated by John Holland in the 1970s (Holland 1992). It leverages the principle of natural selection to choose potential parent solutions from a population, with the aim of generating superior offspring in subsequent generations (Holland 1992). GA refines the set of solutions with each

iteration, progressively nearing the optimal solution. It proves especially potent when dealing with objective functions that are stochastic, non-linear, non-differentiable, or discontinuous. GA is anchored on three core operations: selection, crossover, and mutation. These operations guide the generation of the optimal solution with each successive iteration (Mirjalili and Mirjalili 2019). The foundational steps of the standard GA technique are showcased in Algorithm 1.

Algorithm 1 Standard Genetic Algorithm

Step 1: Randomly generate the initial population of chromosomes, say P

Step 2: Calculate fitness function f (x) for each chromosome x in P

Step 3: Generate child population C by applying selection, crossover, and mutation operators on P

Step 4: P←C

Step 5: Repeat Steps (2–4) until convergence

Fine-tuning the elements of GA to the specific optimization problem at hand is pivotal for attaining optimal performance. Given GA's widespread recognition, we won't delve deeply into its components within this article. For a comprehensive understanding, readers are directed to established research (Mirjalili and Mirjalili 2019; Mitchell 1998). However, it's essential to tailor the algorithm to the unique contours of the problem in focus. Subsequent sections will elucidate how we've molded the conventional GA to align with our distinct context.

4 Results and analysis

This section presents the results and analysis of this study. A through summary of the previous findings of this research project (Khan, et al. 2023) and identified causes of agile-quantum projects are discussed in Sect. 4.1. The results and analysis of developed success probability prediction model is detailed in Sect. 4.2.

4.1 Previous study and identified causes-practices

We extended the previous empirical study (Khan, et al. 2023) findings by summarizing the core categorizing of identified challenges with respect to their relevant causes and practices. Thematically, we categorized the identified challenges across core four categories- knowledge and awareness, quantum-aware tools and technologies, sustainable scaling, and standards and specifications.

We now summarize these categories with respect to the causes of the challenges and their respective best practices.

4.2 Knowledge and awareness

This category encapsulates the substantial challenges related to the knowledge and understanding of using traditional agile practices within the quantum software development realm. The most prominent challenges in this category include *knowledge*

gap, *team dynamics*, and *quantum software development education*. These challenges stem from a lack of *domain specific knowledge*, *lack of market interest*, *limited academic research*, *funding sources*, and *gap between research and practice*. To tackle these challenges, interviewees suggested a list of best practices that recommend fostering team unity through team-building and cohesiveness-enhancing activities, and improving quantum literacy via targeted workshops and training. Strengthening the infrastructure and promoting industry awareness, alongside academic research support, can advance the field. Similarly, understanding the commercial and operational complexities of quantum systems and applying domain best practices are also crucial for effective agile integration. This analysis underscores the criticality of addressing the "Knowledge and Awareness" barriers to successfully scale agile practices in this cutting-edge domain. The list of identified challenges, their causes and best practices is provided in Appendix A.

4.3 Sustainable scaling

This category explores the challenges of bringing agile practices sustainably within the quantum software development sphere, highlighting two principal challenges: *ethically aligned quantum software development* and *agile-quantum ecosystem*. Their underlying causes—*complex technical requirements*, *cross-disciplinary integration difficulties, emerging working culture, quantum-agile interface stability, security, reliability,* and *rapid pace of innovation*—demand targeted practices for resolution. In response, interviewees suggested strengthening quantum domain-specific expertise and establishing defined knowledge management criteria as fundamental steps. Building a technical infrastructure dedicated to quantum software development activities is also important. The development of universally applicable quantum-enabled simulators could democratize access to quantum technologies across various platforms. Furthermore, leveraging software to reduce energy consumption through 'green agile' techniques could contribute to more sustainable quantum computing practices. Tools for continuous quantum–classical hybridization are essential to bridge the gap between traditional/classical computing infrastructure and quantum systems. Finally, positioning quantum ethics at the nexus of quantum information science, technology ethics, and moral philosophy is necessary for a holistic assessment of the impacts posed by emergent quantum technologies. These practices aim to create a scalable, ethical, and agile framework that is responsive to the unique demands of quantum software development.

4.4 Quantum-aware tools and technologies

This category of challenges emerged from two major challenging factors- *classic-quantum tailoring* and *continuous SE infrastructure*. The core causes of these challenges include *right tool for the right job*, *lack of industrial interest*, *limited resources*, *transforming process from legacy to quantum software*, and *technological paradigm shift*. To navigate these challenges, a suite of best practices is essential. Developing agile toolkits specifically for quantum software will facilitate the precise

tailoring needed for quantum tasks. Regular evaluation and selection of development tools for agile based quantum software development will ensure a consistent alignment with project requirements. Allocating dedicated resources within agile frameworks to quantum exploration can address the limitations of funding and focus. Forming quantum interest groups within agile teams can spark industrial interest and foster collaborations. Designating sprints for the transformation from legacy to quantum software allows for a structured approach to this complex transition. Workshops aimed at enhancing agility in quantum computing can prepare teams for the technological paradigm shift. Finally, advancing CI/CD pipelines to accommodate quantum development ensures that the agile practices can be sustained in a continuous SE infrastructure. Integrating these practices can provide a robust framework for overcoming the challenges of agile-quantum software development realm.

4.4.1 Standards and specifications

In the domain of standards and specifications for agile-based quantum software development, there are two pivotal challenges: *process standardization* and *optimum documentation*. These challenges stem from *varying interpretations of agile methodologies*, *rapid evolution of quantum technologies*, *complexity of quantum computing concepts*, the *necessity to integrate with existing standards*, and *interdisciplinary collaboration barriers*. Addressing these challenges necessitates a coherent set of practices:

For process standardization, maturity models tailored to agile-quantum development are essential. They assess and guide teams toward refined agile practices. Complementing these models, standard assessment protocols are necessary to gauge the effectiveness of agile practices and documentation in the quantum software development environment. A comprehensive integration framework is also necessary, providing best practices for blending agile methodologies with the quantum software development process, inclusive of standardized process checklists and templates. Simultaneously, establishing quality standards for documentation specific to the quantum software domain is essential. These standards should guide teams in creating documentation that is clear, concise, and essential for the project at hand. Agile teams should incorporate sprints dedicated to ensuring compliance with established frameworks and standards, allowing for consistent realignment with best practices throughout the development cycle. Furthermore, advocating for the utilization of Software Development Kits (SDKs) with inherent standards and documentation guidelines can streamline adherence to best practices, reducing the overhead for agile teams. Developing agile-quantum playbooks offers a standardized yet adaptable approach to procedures and documentation templates across various quantum development projects. Furthermore, to enhance documentation quality, conducting workshops is key, where agile teams are trained to craft documentation tailored to the unique demands of quantum software development, emphasizing the principles of agility and relevance. Likewise, dedicating specific sprints to the iterative development and refinement of documentation ensures that it evolves iteratively with the rapid development cycles characteristic of quantum software projects. This proactive approach ensures the documentation remains current with the latest project

insights, architectural decisions, and code modifications, achieving an optimal balance throughout the project's lifecycle.

4.5 Agile-quantum software project success prediction model (AQSSPM)

This section provides an overview of the survey participants' demographics (see Sect. 4.2.1), describe the application process of Genetic Algorithms (GA) (see Sect. 4.2.2), and the results of the proposed model are discussed in Sect. 4.2.3.

4.5.1 Experts demographics

We conducted a frequency analysis to examine the collected data, a method suitable for evaluating multiple variables, including numeric and ordinal data. Our study involved 104 participants from 16 countries across four continents, covering 12 professional roles and 12 different project directions, as shown in Fig. 1a–c. A significant number of participants have experience levels between 3 and 5 years in their fields, as depicted in Fig. 1d. We also explored gender representation in the collected responses. Participants were asked to self-identify their gender. The data revealed a majority of male participants at 64%, compared to 19% females. 21% opted not to specify their gender, as shown in Fig. 1e.

To understand our participants' professional roles, we used thematic mapping. This helped categorize their roles into 9 different categories, shown in Fig. 2b. The roles of documentation and training specialist were the most common, with 21 participants. Additionally, we identified 9 main working domains among the
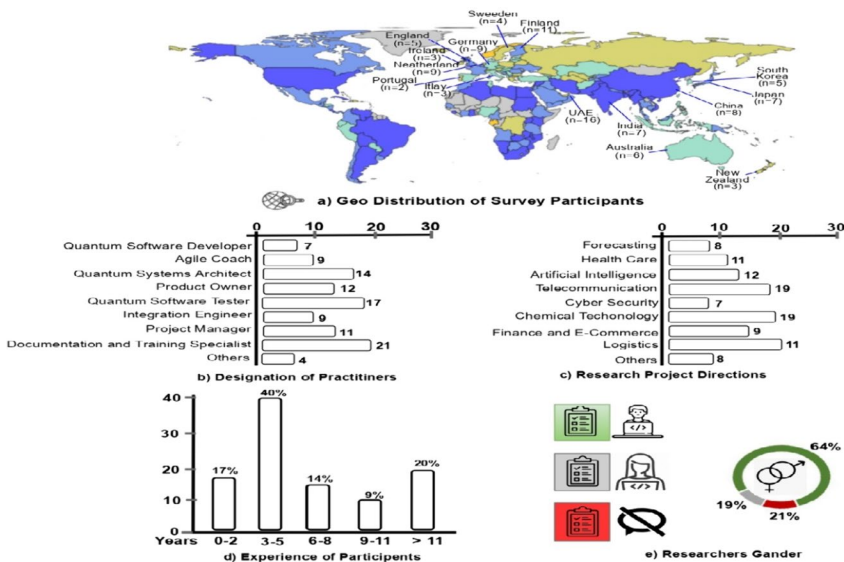

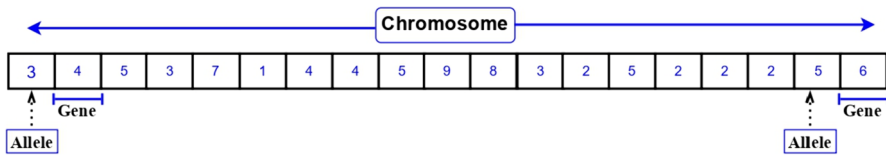**Fig. 1** Demographics data collection process participant

Fig. 2 The structure used to represent a chromosome

participants. Telecommunication and chemical technology emerged as dominant sectors, each with 19 participants, as illustrated in Fig. 1c.

4.5.2 Genetic algorithm application process

This section formally presents the optimization problem of success probability maximization in the presence of cost. It is explained that the GA algorithm needs to be customized for this specific problem. Additionally, the necessary modifications to GA and its components are also discussed. We have employed the GA in the following steps:

Step-1 Representation

In the current study, each potential solution, i.e., chromosome, comprises an ordered set of values. The order of the values in the set corresponds to the order of attributes described in Fig. 2. The first value in the set represents the first attribute scale, the second value represents the second attribute level, and so on.

Step- 2 Initialization

The research utilized a sample size of 50, where the initial set was produced randomly. Precisely, 50 chromosomes were established at the onset of the algorithm, and every chromosome component was allocated a random whole number ranging from 1 to 9, inclusive.

Step-3 Fitness function

The efficiency of a chromosome is determined using Eq. 15, denoting the chromosome's fitness metric. A superior fitness metric signifies a heightened likelihood of success and reduced related expenses.

Step -4 Constraint handling

Numerous practical challenges come with constraints, suggesting that a solution might not always be viable. Chromosomes formed in GA might sometimes reside in an impractical search area, indicating they don't adhere to the parameters outlined in Eq. 15. To ensure that chromosomes conform to the stipulated guidelines, they are designed to meet specific constraints. This ensures they remain within boundaries, even after iterative changes across generations. The value range was confined between S_{max} and S_{min}, and the GA settings were tailored to produce chromosomes within these acceptable zones.

Step-5 Selection & Reproduction

The present research adopted the roulette-wheel selection method from the given options due to its straightforward nature and efficiency.

Step–6 Crossover

A set probability value of 0.8 is employed for chromosome crossover, and the single-point crossover technique was chosen for its straightforward approach.

Step–7 Mutation

We opted for the random mutation approach, which has a probability of 0.1. This means there is a 10% chance for each chromosome to be substituted with a random number ranging between S_{max} and $S_{min.}$

Step-8 Stopping criteria

The study adopted a threshold of 100 generations as the termination condition. All essential elements and their corresponding values pertinent to the optimization issue outlined in Eq. 15 can be found in Table 2.

Execution of GA- In the sections above, we elaborated on all the essential components and their associated values required for the GA execution. Algorithm 2 offers a thorough recapitulation of the algorithm's steps.

Algorithm 2 Steps of Genetic Algorithm employed

Step 1: Train a predictive model to produce success probability

Step 2: Set the values of GA parameters

Step 3: Randomly generate the initial population of chromosomes, says *Pop*

Step 4: Calculate the success probability *Prob*(*S*) of each chromosome *S* in Pop using Eq. 8

with the help of the predictive model mentioned in Step 1

Step 5: Calculate the normalized cost of each chromosome *S* in Pop using Eq. 13

Step 6: Calculate efficacy *E*(*S*) for each chromosome *S* in Pop using Eq. 15, and it will serve

as the fitness function *f*(*S*) of S

Step 7: Generate child population C by applying selection, crossover, and mutation operators on
  Pop

Step 8: Pop ← C

Step 9: Repeat Steps (4–8) until the stopping criteria are met

---

Table 2 The values of the GA parameters

Parameter name	Value
Max. iterations	100
Size of population	50
Total genes	40
Type of gene	Integer
Max. gene value	9
Min. gene value	1
Selection method	Roulette wheel
Crossover method	Single point
Probability of crossover	0.8
Mutation method	Random
Probability of mutation	0.1
Fitness function	Efficacy (Eq. 15)

Upon successful execution of Algorithm 2, the chromosome labeled SB, having the optimal fitness value, is retained. Leveraging the scale values of each variable encompassed in SB, an optimal solution conducive to project success can be derived.

4.5.3 AQSSPM results

This section presents the final results obtained by implementing the GA with Naive Bayes classifier (NBC) and the Logistic regression (LR) model. Furthermore, the results of NBC and LR correlation analysis are also presented in sub-sequent section.

5 Results with GA-NBC

The results of the GA-NBC for agile-quantum project success probability are presented in Table 3 while the best fitness order of the identified 19 causes are presented in Table 4. Initially, the success probability and cost were 53.17% and 0.463, respectively. After 100 generations, the optimal solution yielded a cost of 0.403 and a success probability of 99.68%, signifying a 46.51% improvement in success probability and 6% decrease in cost, as given in Table 3. The results from the NBC and GA analyses indicate the initial cost of developing quantum software using agile was relatively high.

The initial low success rate and high cost in quantum software development using agile methodologies can be attributed to various causes faced by the industry as given in Appendix A. One of the primary obstacles is the C1 (lack of domain-specific knowledge), which hampers the efficient application and understanding of quantum principles. Furthermore, a noticeable C2 (lack of market interest), combined with C3 (limited academic research), means there is a dearth of foundational and advanced knowledge being disseminated and applied. Furthermore, C4 (funding sources) are limited, making it challenging to invest in innovative solutions and tools that could streamline the development process. The frequent iterations underlying to agile methods exacerbate resource consumption, potentially straining already limited resources. C6 (cross-disciplinary integration difficulties) adds another layer of complexity, and developing quantum code at the gate level remains a specialized and intricate task.

The agile approach demands the C8 (right tool for the right job), but this is often lacking in the agile-quantum realm. The stability, security, and reliability of the quantum-agile interface are still under scrutiny, and the dynamics and evolution of business models in this nascent field can be unpredictable. Classical agile practices might not seamlessly integrate with quantum software systems, raising concerns about their compatibility.

Moreover, the quantum software development life cycle is still in its infancy, with many organizations grappling with the transition from classical to quantum software. This transformation is further hampered by a discernible gap between research and practical application. As the industry evolves, C18 (emerging working cultures)

Table 3 The initial and ending fitness achieved by the NBC model for all variables

Stages	Generations	Initial success probability (%)	Ending success probability (%)	Change in probability (%)	Initial cost	Ending cost	Change in cost
Naive–bayes	100	53.17	99.68	+46.51	0.463	0.403	-6.0%

Table 4 The best fitness of causes obtained from all stages of GA-NBC

Model	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19
Naive–bayes	7	7	2	5	4	9	2	6	6	8	8	4	6	5	3	2	3	6	2

and the imperative of adhering to business ethics and corporate governance can influence the trajectory of agile adoption for quantum software development.

However, as the process matures and these causes are addressed, there is potential for a significant increase in success probability, as endorsed by the GA-NBC analysis. The subsequent decrease in cost also suggests that with time and focused efforts, the quantum software industry can optimize its agile methodologies, leading to more efficient and cost-effective outcomes.

In the realm of agile-quantum software development, several causes could influence the success probability and cost. The results given in Table 4 suggest the best fitness of identified causes to increase the project success probability and to decrease the cost. For instance, the cause C6 (Cross-disciplinary integration difficulties) holds the most significant impact. This highlights the need of integrating cross-disciplinary practices within quantum projects to achieve optimal outcomes. Following closely are causes C10 (limited resources) and C11 (varying interpretations of agile methodologies). The scarcity of specialized tools and expertise underscores the importance of resource optimization. Additionally, the interpretations of agile methods and techniques should be rigorous, emphasizing stable protocols and efficient integrations. Together, addressing these primary causes can significantly bolster the success rate of agile-quantum projects.

6 Results with logistic regression model

The success probability of agile-quantum projects was also assessed using a GA-LR model. Table 5 presented that when the LR model is paired with GA, it considerably impacts the project's outcome. Initially, the project had a success rate of 55.52% at a cost of 0.496. But after 100 iterations with the GA-LR, the outcome improved dramatically, showing a success rate of 98.99% at a lowered cost of 0.409. This indicates a notable 43.47% improvement in success probability and an 8.70% cost reduction (Table 5). This indicated that by focusing on the main causes, we can increase the agile-quantum project's success probability by 43.47% while also cutting down costs.

Therefore, we determined the best fitness of identified causes to increase the project success probability and to decrease the cost by applying GA-LR model (Table 6). For instance, C1 (lack of domain specific knowledge) stands out as the most critical cause, as the field of quantum software engineering demands specialized knowledge for effective project execution. Without a deep understanding of the domain, teams can face significant barriers, leading to inefficiencies and increased costs. Moreover, C10 (limited resource), agile methodologies, known for their iterative nature, require consistent and specialized resources. A scarcity can hinder the smooth progression of the development cycle, potentially escalating costs. C11 (varying interpretations of agile methodologies) is equally crucial. As quantum computing introduces new paradigms, ensuring a seamless, secure interface with agile practices becomes paramount. Any vulnerabilities can not only risk the project's success but also inflate costs due to unforeseen issues. Additionally, C2 (lack of market interest) plays a pivotal role. For any project to be successful, it needs to align with

Table 5 The initial and ending fitness achieved by the LR model for all variables

Stages	Generations	Initial success probability	Ending success probability	Change in probability	Initial Cost	Ending Cost	Change in cost
Logistic–regression	100	55.52%	98.99%	+43.47%	0.496	0.409	-8.70%

Table 6 The best variables are obtained for all the stages from the LR classifier

Model	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19
Logistic–regression	8	7	2	5	4	7	2	6	6	8	8	3	5	5	4	1	4	7	2

market demands. If the agile-quantum project doesn't resonate with market interest, its chances of success diminish, and resources invested might not yield the desired return. Lastly, C6 (cross-disciplinary integration difficulties) is essential to the success of agile-quantum projects, as it enables the harmonious combination of diverse expertise and insights. Successfully addressing C6 is key to fostering innovation and ensuring the efficacy of agile-quantum initiatives. In essence, prioritizing and addressing these causes head-on can significantly enhance the success trajectory of agile-quantum projects while ensuring cost-effectiveness.

6.1 Comparison between NBC and LR based best fitness of identified causes

In this section, we compared the results (best fitness of causes) derived from the GA-NBC with those of the GA-LR approach to gain a holistic insight into their respective efficacies.

Variable Best Fitness Order—Our proposed framework for gauging the success probability in agile-quantum projects presents two key benefits. Firstly, it suggests an ideal cost allocation across different causes to amplify the success probability of agile-quantum project. Secondly, it offers insights into the comparative significance of the identified causes.

Table 3 presents the initial and ending cost to fix the identified causes necessary to approach a near-perfect project success rate. The results suggests that some causes might not be pivotal, as elevated success percentages can be maintained even with reduced cost allocations. For instance, cause C7 (rapid pace of innovation) need 2 units to attain a success probability of 98.68% using the GA-NBC. Conversely, C6 (cross-disciplinary integration difficulties) requires an allocation of 9 units, indicating a more substantial financial and effort commitment to optimize the probability of success in the agile-quantum project (see Table 4).

Further assessments employing GA-LR learned the financial allocation per cause critical for enhancing the success probability of agile-quantum project. Table 5 reveals that certain causes employ minimal influence on success, sustaining high success percentages even with minimum allocated funds and effort. For instance, C3 (limited academic research) demands only a 2-unit allocation, while C1 (lack of domain-specific knowledge), C10 (limited resources), and C11 (varying interpretations of agile methodologies) each necessitate 8 units. Based on these findings, emphasizing C1, C10, and C11 while limiting the budget for C3 ensures an elevated success probability i.e., 99.29%.

By analyzing the relationship between each cause financial allocation and the project's success probability, we can order the causes based on their cost-effectiveness and overall relevance. Employing results from the GA-NBC (refer to Table 4) and GA-LR (see Table 6), we allocate rankings to each cause and the determined ranks are given in Table 7.

For example, C6 (cross-disciplinary integration difficulties), with an allocation of 9 units in the NBC analysis and 7 units in the LR analysis, holds the first and second ranks, respectively, highlighting its paramount importance. Using a similar methodology for all causes, we compile a comprehensive ranking (displayed in Table 7),

Table 7 Best fitness ranking of causes for both classifiers

Model	Particulars	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19
GA-NBC	Best Fitness	7	7	2	5	4	9	2	6	6	8	8	4	6	5	3	2	3	6	2
	Ranks	3	3	8	5	6	1	8	4	4	2	2	6	4	5	7	8	7	4	8
GA-LR	Best Fitness	8	7	2	5	4	7	2	6	6	8	8	3	5	5	4	1	4	7	2
	Ranks	1	2	7	4	5	2	7	3	3	1	1	6	4	4	5	9	5	2	7

assisting professionals in emphasizing the pivotal cause for agile-quantum projects, thereby substantially improving the project success probability.

6.2 NBC and LR correlation analysis

We undertook a non-parametric statistical analysis to identify the similarities and differences between the optimal fitness rankings derived from the Naive Bayes Classifier (NBC) and Logistic Regression (LR) (Failed 2004; Winter et al. 2016). Several relative research studies have adopted this non-parametric statistical analysis approach (Khan et al. 2017; Akbar et al. 2022; Khan and Akbar 2020). For a uniform metric, we employed Spearman's Rank-Order Correlation Coefficient, organizing each factor associated with the agile-quantum project in accordance with the supreme fitness values yielded by both NBC and LR (see Table 7). This correlation coefficient gauges the linear association between factors, with potential values spanning from $+1$ to -1. A coefficient of $+1$ signifies an absolute linear correlation.

A Spearman's Rank-order correlation coefficient value was (rs $=0.955$, $p = 0.000$), as presented in Table 8. This high correlation value underscores a strong association between the rankings derived from Naive Bayes Classifier (NBC) and Logistic Regression (LR) methodologies. This remarkable agreement implies that practitioners can utilize either of these techniques to pinpoint the significant factors (causes) affecting agile-quantum projects.

Although there is significant alignment, minor variations in rankings attributed to specific factors by NBC and LR were detected. For example, C1 (lack of domain-specific knowledge) secured the 3rd rank using NBC but ascended to the top rank with LR. Similarly, C2 (lack of market interest) and C4 (funding resources) showed slight shifts in rankings (as seen in Table 8). These nuanced differences hint at a measure of variation in the ranking system between the two techniques, a factor to bear in mind when discerning pivotal aspects of agile-quantum projects.

Beyond the Spearman's Rank-order correlation, an independent t-test was applied to evaluate the average rank differences generated by both methods (see Table 9). The Levene's Test outcome was non-significant ($p = 0.967 > 0.05$), confirming uniform variances, and our analysis proceeded under this assumption. The t-test results ($t = 1.195$, $p = 0.240 > 0.05$) revealed no significant differences between the ranks produced by both methodologies. This uniformity confirms that, for determining

Table 8 Correlation between the ranking of NBC and LR

			LR_Ranking	NBC_Ranking
Spearman's rho	LR_ranking	Correlation Coefficient	1.000	0.955[**]
		Sig. (2-tailed)	–	0.000
		N	19	19
	NBC_ranking	Correlation Coefficient	0.955[**]	1.000
		Sig. (2-tailed)	0.000	
		N	19	19

**. Correlation is significant at the 0.01 level (2-tailed)

Table 9 Independent Samples *t*-test of NBC and LR rankings

	Levene's Test for equality of variances		t-test for Equality of Means						95% confidence interval of the difference	
		F	Sig	t	df	Sig. (2-tailed)	Mean difference	Std. Error difference	Lower	Upper
Rank	Equal variances assumed	0.002	0.967	1.195	36	0.240	0.89474	0.74886	− 0.62402	2.41349
	Equal variances not assumed			1.195	35.941	0.240	0.89474	0.74886	− 0.62410	2.41358

variable importance, practitioners can confidently adopt either NBC or LR, optimizing resource distribution and amplifying the success prospects of agile-quantum projects.

7 Summary of results

Now we present the summary of the developed AQSSPM model results in Sect. 5.1 and the core implications of this research project are reported in Sect. 5.2.

7.1 AQSSPM

The objective of this research project was to explore the challenges, causes behind these challenges and the most effective practices to address them. Finally, based on the identified causes, develop the AQSSPM model which could evaluate the success probability of agile-quantum projects, serving as a roadmap to industry practitioners for informed decision-making. Central to this model's development is a thorough examination of key causes, validated by industry experts through structured questionnaire study. To develop AQSSPM, we used the GA, NBC and LR approaches, which analyze the causes in terms of financial allocation, and cost (team composition, monetary pledges, and other essential assets). Following this, our study gives deep insights of the core causes need to be addressed on priority basis to increase the agile-quantum project success and to minimize the cost.

The developed AQSSPM is presented in Fig. 3, that shows that using GA alongside NBC, we observed the success probability of an agile-quantum project soaring from an initial 53.17% to a remarkable 99.68% over its lifecycle, while concurrently reducing costs from 0.463 to 0.403%. On other hand the GA with LR, the success rate leaped from 55.52 to 98.99%, coupled with a cost saving, as costs declined from 0.496 to 0.409% after 100 iterations.

Enhancing the power of GA in combination with both NBC and LR granted us the ability to perceive the fundamental determinants behind the success of agile-quantum projects. This evaluation gives a body of knowledge to practitioners with a solid roadmap, underscoring the significance of domain specific knowledge, proper cross-disciplinary integration, availability of resources, adaptability to technological paradigm shifting, and increase market interest. Acknowledging and ranking these causes can revolutionize project execution, setting the stage for agile-quantum endeavors that are successful and financially practical. In summation, the result of this study serves the industry practitioners and researchers with the knowledge and tool required to amplify the success probability of their agile-quantum projects.

7.2 Academic implications

The research presented in this paper has substantial academic implications, particularly for the growing field of QSE. By investigating the suitability of traditional agile methodologies within quantum software development, this study
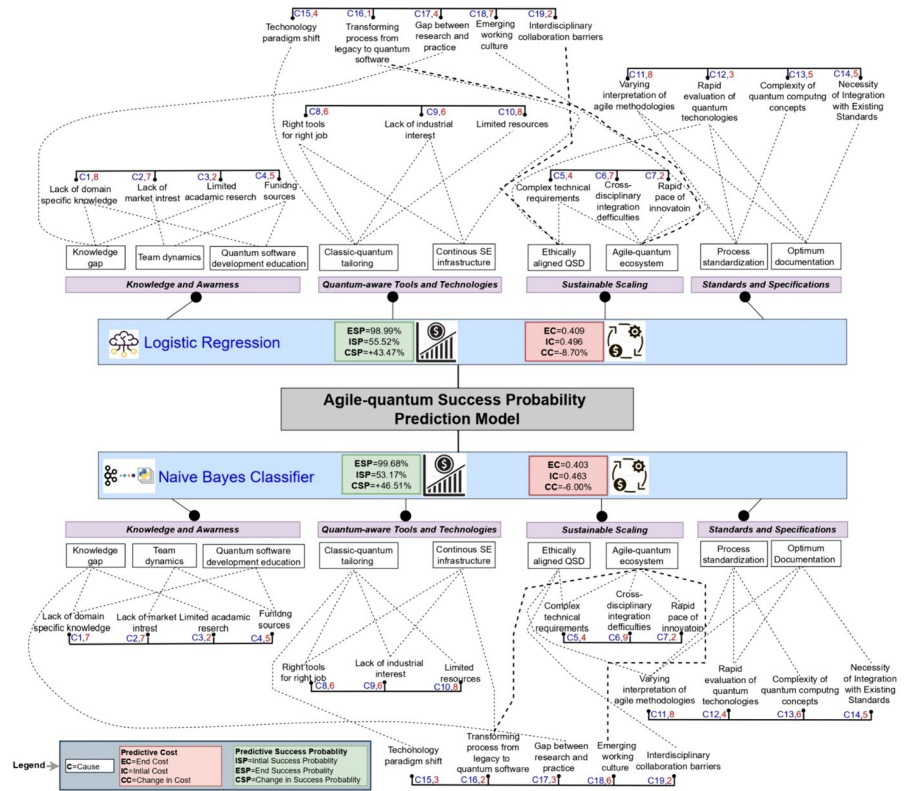
Fig. 3 Agile-quantum project success probability prediction model

contributes to the theoretical foundation of QSE, an area that is still in its infancy. The paper extends the knowledge base by identifying challenges specific to quantum software development, their root causes, and offering best practices to overcome them, thus filling a gap in the existing literature.

The introduction of a predictive model based on genetic algorithms represents an innovative approach to address decision-making in quantum software projects. This model, which employs nature-inspired algorithms to evaluate project success probability, not only adds a new tool for academic research but also sets the stage for future studies to refine and validate predictive models in this domain. It encourages further empirical research to explore the effectiveness of agile practices in quantum software development, potentially leading to the development of new frameworks and methodologies that are specifically designed for the quantum realm.

Moreover, the interdisciplinary nature of this study, bridging software engineering with QC, illustrates the importance of cross-disciplinary research in addressing complex technological challenges. It provides a template for how agile methodologies can be adapted to fit new and emerging fields, offering a blueprint that can be studied and replicated across different domains of technology research.

Based on this study, future research could be conducted to integrate traditional software engineering methodologies, tools, and techniques within the quantum software domain. This direction would be critical, considering that the current infrastructure for QSE is emerging and largely unexplored. Subsequent studies could evaluate how existing software engineering practices could be adapted for quantum computing, thus maximizing the concept of hybridization between conventional and quantum software development. This integration is essential for establishing a robust quantum software engineering ecosystem that can support the rapid evolution and unique demands of quantum computing technologies.

7.3 Industrial implications

For the industry, the implications of this research are equally significant. The transition from traditional computing to quantum computing represents a paradigm shift in how computational tasks are approached and executed. By offering a detailed analysis of the challenges and best practices for adopting agile methods in quantum software development, the paper provides industry professionals with a framework to navigate this transition effectively.

The predictive model introduced in this paper serves as a practical tool for organizations considering the integration of existing agile practices into their quantum computing projects. It enables them to make informed decisions about the potential costs and outcomes of their projects, thereby reducing uncertainty and increasing the likelihood of successful implementation.

Additionally, by highlighting the need for quantum-aware tools and technologies, the research underscores the market demand for specialized solutions that organize to the unique requirements of quantum software development. This could stimulate investment and innovation in the development of such tools, fostering a supportive ecosystem for the growth of the quantum software industry.

Overall, the findings from this research provide valuable insights that can guide the development of quantum software, ensuring that the industry can keep pace with technological advancements while maintaining software quality, sustainability, and ethical standards.

8 Threats to validity

Various potential threats could affect the validity of the study findings. We have examined potential threats across the four essential categories of validity threats: internal validity, external validity, construct validity, and conclusion validity, as defined by Wohlin et al. (2012).

8.1 Internal validity threats

Internal validity refers to the extent to which particular factors affect the results and findings of the collected data. The internal validity of our study may be

compromised by several factors based on the survey design and the subjectivity of responses. The use of a closed-ended questionnaire, while beneficial for standardizing responses, may limit the depth of insight into the complex integration of traditional and quantum software engineering practices. The 9-point rating scale assumes a linear perception of the attributes (variables) being measured, which may not accurately capture the detailed opinions of agile and quantum computing practitioners. Additionally, the study validity can be compromised if participant responses are influenced by their individual levels of familiarity and hands-on experience with quantum software development, potentially introducing biases that could skew the results. Moreover, the misinterpretation of survey questions by the participants could jeopardize the accuracy of the data collected. To minimize these threats, we piloted our survey instrument, refining it with feedback from experts in empirical software engineering to enhance its reliability, understandability, structure, and validity.

Furthermore, the potential biases stemming from the use of genetic algorithm, NBC, and LG—such as premature convergence, feature independence assumption, and assumption of linearity—are primarily threats to the internal validity of the study findings. These biases can influence the causal relationships and the accuracy of the model predictions due to selection bias, and assumptions about data characteristics that do not hold true. However, we collected data from 104 participants across 16 different countries, consisting of 12 professional roles and project directions which significantly enhances the diversity and representativeness of the study sample. This wide range of data helps mitigate biases related to sample selection and algorithm assumptions by ensuring a broad spectrum of insights and experiences. It contributes to the model generalizability and reduces the risk of overfitting.

8.2 External validity threats

External validity concerns the extent to which a study results can be generalized beyond the particular sample, setting, or conditions examined. This involves considering how the study findings are applicable to other groups, environments, or scenarios not directly investigated in the research. The external validity of our study is based on the representativeness of our participant pool. With data collected from 104 agile and quantum computing practitioners, there is a risk that our sample may not be sufficiently diverse or large enough to generalize findings across the entire field of quantum software development. Despite efforts to ensure a diverse population, the possibility that our sample is not fully representative of the global community or all expertise levels in quantum computing and agile methodologies remains a threat to the validity of our study findings. Furthermore, the rapid evolution of quantum computing technology may mean that our findings have a limited shelf-life, potentially reducing their applicability as the field advances. Therefore, to mitigate the stated threats, our survey reached participants from 16 countries across four continents, broadening the cultural and geographical scope of our study. Our sample size is on par with similar studies in the growing field of quantum software development (Khan et al. 2023), supporting the

generalizability of our results. The international reach of our survey supports a global perspective, potentially reflecting wider trends in quantum computing practices.

Additionally, the broader applicability of the proposed model presents a significant threat to the external validity of the study findings. Its practical use in real-world projects could be affected by certain factors which were not directly considered on this study, such as existing (classical) software development practices and the software practitioners willingness to adopt new technologies. To address this issue and improve the study external validity, future research should aim at conducting case studies or pilot implementations of the model in real-world quantum software development projects.

8.3 Construct validity threats

Construct validity addresses the degree to which the survey questions accurately measure the concepts they are intended to assess. In our study, the construct validity is dependent upon how well the survey questionnaire design captures the study aim of integrating traditional agile practices for developing a quantum software system. There is a possibility that the survey items might not fully encapsulate the challenges of bringing agile practices into a quantum software environment, which could result in an incomplete understanding of the targeted problem. To ensure that the constructs are valid and the survey questions are well-calibrated, we incorporated iterative feedback from a diverse range of experts in the field to fine-tune the survey items, thereby enhancing the relevance and accuracy of the constructs being measured.

8.4 Conclusion validity threats

Conclusion Validity refers to the level of credibility or legitimacy of the study conclusions drawn from the overall study results. The conclusion validity of our study is dependent on the questionnaire design, which was based on previous studies and literature, there is a risk that certain pertinent variables may have been overlooked, which could lead to incorrect inferences. The use of a 9-point scale and the subjective nature of the responses pose a risk of misclassification or overestimation of the strength of associations. Moreover, with a sample size of 104 participants, our study might not have the power to detect subtle but meaningful differences or trends, which could lead to either Type I or Type II errors in our conclusions. To minimize the conclusion validity threats, the survey instrument was thoroughly reviewed by subject matter experts for clarity and coherence and was structured in alignment with established empirical software engineering study designs (Khan et al. 2017; Akbar et al. 2023).

9 Related work

We now summarized the related work focusing on quantum software development with respect to software processes (Sect. 5.1) and probabilistic models developed using the genetic-based algorithm (Sect. 5.2).

9.1 Quantum software development

Quantum software development processes involve designing, implementing, and testing software that runs on QC platforms. This process requires an understanding of quantum algorithms, quantum logic gates, and the unique behavior of quantum bits (qubits). Based on the given characteristics of QC, its challenging to develop quantum-specific software systems using the traditional processes. Few studies have been conducted that focused on defining quantum software processes, methods, and frameworks.

For instance, González(2020) described that in the realm of QC, hardware advancements, characterized by the rapid increase in qubit numbers, maximize the need for the core approaches used to execute the sophisticated algorithms and develop commercial systems. However, programming and engineering methodologies have not kept pace, lacking the specific approaches required for the novel challenges and risks inherent to quantum technology. This paper responds to the need for advanced project management strategies in quantum software development, advocating for a hybrid framework that orchestrates traditional agile methods with the unique aspects of quantum programming. This proposed framework aims to organize the complexities of managing projects that blend classical and quantum development, ensuring readiness for the future landscape of software engineering in the quantum era.

Khan et al. (2022) presented an idea paper and discussed that in realm of HOC and QC, where resource usage is costly, software development demands a nearly flawless product before execution. This paper proposes an agile development methodology tailored for quantum software, where initial stages use interactive, low-cost computations for conceptualization and experimentation. As the project matures, more resource-intensive tasks like algorithm optimization are batch-processed, integrating iterative development with the resource-conscious demands of QC.

Weder et al. (2022) reported that the complexity of developing quantum software applications requires multidisciplinary expertise, which boosts the emergence of quantum software engineering as a field focused on the application of systematic software principles. Hybrid quantum applications, integrating quantum and classical components, need life cycles that address both, plus the orchestration of execution and data exchange. This study introduces a comprehensive quantum software development life cycle, examining phases, tools, and the integration of different software artifacts life cycles. This article outlines hybrid quantum application fundamentals, presents the proposed life cycle, discusses assumptions and limitations, reviews related work, and concludes with future perspectives.

Khan et al. (2023) explore the organization of agile development practices within quantum software development through encapsulating the opinions of practitioners from 10 countries. Findings suggest agile practices are applicable for developing quantum software but also uncover unique challenges in integrating these agile approaches effectively. The research offers insights for adapting agile practices to the specific demands of quantum software engineering, paving the way for the advancement of future quantum software systems and applications.

9.2 Genetic algorithm-based probabilistic models

A genetic algorithm is a search heuristic that uses natural selection and genetic processes such as mutation and crossover, evolving a population of solutions to optimize results. It selects the fittest individuals through a probabilistic fitness function, combining directed search with randomness to efficiently explore complex solution spaces. Various studies have used it to develop cost-based probabilistic models to determine the probable project outcomes.

For instance, Shameem et al. (2023) conducted a study to address the complexity and risks associated with implementing agile methods across globally distributed software development teams. By leveraging a genetic algorithm, the authors have presented a predictive model that assesses the probability of agile project success while considering cost factors. The effectiveness of the model is validated through its ability to anticipate project outcomes by analyzing critical agile project features (factors). It used established prediction techniques, providing insights into the project features that GSD teams should prioritize to enhance the success rates of agile implementations without incurring unnecessary costs.

Alsghaier (2020) presents an innovative approach to software fault prediction by combining the genetic algorithm with the support vector machine (SVM) classifier, enhanced by particle swarm optimization. This methodology aids developers in identifying faulty classes or modules early in the software development lifecycle and directs attention to areas that may require additional refactoring or testing, particularly in mission-critical systems like aircraft or medical devices. Applied to both large-scale NASA datasets and smaller Java open-source projects, the integrated approach showed improved performance in predicting software faults across diverse datasets, indicating a significant step forward in fault detection techniques and overcoming limitations found in previous studies.

Bilgaiyan et al. (2020) address the challenge of effort estimation in agile software development, a significant perspective that can influence the success of a project. While agile methods are favored for their adaptability and efficiency over traditional models like waterfall and spiral, they introduce complexities in estimating costs and efforts due to their dynamic nature. Traditional estimation techniques such as analogy, expert opinion, and disaggregation lack a solid mathematical foundation. To bridge this gap, the authors propose an innovative approach using evolutionary algorithms, specifically a chaotically modified genetic algorithm (CMGA). This approach is based on mathematical morphology and utilizes a hybrid-artificial neuron model, the Dilation-Erosion perceptron (DEP), derived from complete lattice theory (CLT). The DEP-CMGA model was calibrated using data from 21 ASD projects, and its accuracy was assessed using four statistical metrics. The performance of this model was also compared with the best existing models in the literature, highlighting its potential for improving effort estimation in agile environments.

Lester et al. (2020) investigate the concept of design rationale (DR), a repository of decisions and justifications made during the design process, beneficial for guiding future designs and maintaining consistency. Capturing DR is typically avoided due to its perceived cost and effort, leading to an implicit presence in various project documents. The research focuses on automating DR extraction through text mining,

using machine learning to classify document sentences. It highlights the importance of feature selection in text mining and compares two evolutionary algorithms—Ant Colony Optimization (ACO) and Genetic Algorithms (GAs)—for optimizing this selection. The study assesses the effectiveness of these algorithms using the F-1 measure, aiming to determine if and how evolutionary algorithms can enhance the identification of DR from text documents.

10 Summary

Our work presents a novel contribution to the field of quantum software development by presenting a decision-making cost probability prediction model for using the traditional agile approaches to develop quantum software. Where previous studies have established the need for new frameworks and methodologies, such as the quantum agile development framework and agile methodologies tailored for quantum software, our research goes a step further by providing a detailed automated cost prediction framework for adopting traditional agile practices specifically for quantum software systems.

In the domain of using the genetic algorithm, our work stands out by applying these to develop the mentioned proposed cost prediction model, which presents cost prediction and best fitness of a set of features (causes). While other studies have successfully used genetic algorithms for predictive modeling and effort estimation in agile software development, we innovate by tailoring these algorithms to address the complexities and uncertainties of quantum software projects. This significant pivot from conventional software to quantum software represents a rise in applying established traditional computational techniques to an emerging field, potentially setting a benchmark for future research and development in quantum software engineering. Our model not only predicts project cost, best fitness of the features (causes) but also contributes to the understanding of hybrid quantum software development, thus providing a comprehensive toolset for developers and project managers in this cutting-edge domain.

11 Conclusion and future direction

Our study explored Quantum Software Engineering, a field that blends the established practices of classical software engineering with the emerging landscape of quantum computing. This integration is critical in steering architects and developers towards the creation of innovative quantum software applications, thereby advancing the field of quantum computing. The primary aim of our research was to identify key challenges and their causes in agile-based quantum software development projects and to develop a robust Agile-Quantum Software Project Success Prediction Model (AQSSPM). For this purpose, we conducted a survey to gather insights from practitioners in the field. Additionally, we jointly implemented genetic algorithms (GA) with Naive-Bayes Classifiers (NBC), and Logistic Regression (LR) to estimate the success probability of agile-quantum software

projects. We also evaluated the relevance of the identified causes of the challenges in relation to their impact on the success of these projects.

Our results were quite revealing. We found that integrating GA with NBC significantly improved the probability of project success, rising from an initial 53.17% to an impressive 99.68%, and also reduced costs from 0.463 to 0.403%. In a similar vein, when GA was paired with LR, there was a notable increase in the success rate, from 55.52 to 98.99%, with a corresponding reduction in costs from 0.496 to 0.409%. Furthermore, we identified the most critical factors (causes) for the success of agile-quantum projects, which included domain-specific knowledge, effective cross-disciplinary integration, availability of resources, adaptability to technological paradigm shifts, and growing market interest. These findings are expected to provide the research and practitioner communities with robust analytical tools and strategic insights, significantly enhancing the success rates of agile-quantum projects.

Looking towards the future, we plan to test the proposed model (AQSSPM) across various quantum computing projects to evaluate its adaptability and robustness. Moreover, based on the study findings, we will enhance the model with additional predictive algorithms to improve its accuracy. Additionally, we will conduct longitudinal studies to monitor the AQSSPM's long-term performance in the fast-evolving field of quantum computing would be beneficial. Finally, we plan to develop and incorporate a structured set of guidelines specifically designed for practitioners applying the proposed model in their projects. These guidelines will include practical steps, best practices, and illustrative examples, enhancing the practical value of the model and making its accessible and actionable for real-world implementation.

Availability of data and materials The codes and data are available under request from the authors.

Declarations

Ethical approval The survey data was collected based on the concent of the respondents. The sample survey questionnaire is provided at: https://tinyurl.com/43r7uxx6.

References

Ahmad, A., Khan, A. A., Waseem, M., Fahmideh, M. and Mikkonen, T.: "Towards process centered architecting for quantum software systems". In: 2022 IEEE international conference on quantum software (QSW), 2022: IEEE, pp. 26–31

Akbar, M.A., Khan, A.A., Mahmood, S., Mishra, A.: SRCMIMM: the software requirements change management and implementation maturity model in the domain of global software development industry. Inform. Technol. Manag. **24**(3), 195–219 (2023)

Akbar, M.A., Khan, A.A., Rafi, S.: A systematic decision-making framework for tackling quantum software engineering challenges. Autom. Softw. Eng. **30**(2), 22 (2023)

Akbar, M. A., Khan, A. A., Mahmood, S., and Rafi, S.: "Quantum Software Engineering: A New Genre of Computing," arXiv preprint arXiv:2211.13990, 2022

Ali, S., Yue, T., Abreu, R.: When software engineering meets quantum computing. Commun. ACM **65**(4), 84–88 (2022)

Alsghaier, H., Akour, M.: Software fault prediction using particle swarm algorithm with genetic algorithm and support vector machine classifier. Softw. Pract. Exp. **50**(4), 407–427 (2020)

Beck, K., et al.: Manifesto for agile software development (2001). https://ai-learn.it/wp-content/uploads/2019/03/03_ManifestoofAgileSoftwareDevelopment-1.pdf

Berrar, D.: Bayes' theorem and naive Bayes classifier. Encyclop. Bioinform. Comput. Biol. ABC Bioinform. **403**, 412 (2018)

Bilgaiyan, S., Panigrahi, P. K. and Mishra, S.: "Chaos-based modified morphological genetic algorithm for effort estimation in agile software development". J. Towards Bio-inspired Tech. Softw. Eng. pp. 89–102, 2020.

Carleton, A., Shull, F., Harper, E.: Architecting the future of software engineering. Computer **55**(9), 89–93 (2022)

Cerpa, N., Bardeen, M., Astudillo, C.A., Verner, J.: Evaluating different families of prediction methods for estimating software project outcomes. J. Syst. Softw. **112**, 48–64 (2016)

De Stefano, M., Pecorelli, F., Di Nucci, D., Palomba, F., De Lucia, A.: Software engineering for quantum programming: how far are we? J. Syst. Softw. **190**, 111326 (2022)

De Winter, J.C., Gosling, S.D., Potter, J.: Comparing the Pearson and Spearman correlation coefficients across distributions and sample sizes: a tutorial using simulations and empirical data. Psychol. Methods **21**(3), 273 (2016)

Fitzgerald, B., Hartnett, G., Conboy, K.: Customising agile methods to software practices at Intel Shannon. Eur. J. Inf. Syst. **15**(2), 200–213 (2006)

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

Hernández González, G. J., and Paradela, C. A.: "Quantum agile development framework". In: International conference on the quality of information and communications technology, 2020: Springer, pp. 284–291

Holland, J.H.: Genetic algorithms. Sci. Am. **267**(1), 66–73 (1992)

Hu, P., Pan, J.-S., Chu, S.-C.: Improved binary grey wolf optimizer and its application for feature selection. Knowl. Based Syst. **195**, 105746 (2020)

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

Khan, A.A., et al.: Software architecture for quantum computing systems—a systematic review. J. Syst. Softw. **201**, 111682 (2023)

Khan, A.A., Akbar, M.A.: Systematic literature review and empirical investigation of motivators for requirements change management process in global software development. J. Softw. Evolut. Process **32**(4), e2242 (2020)

Khan, A.A., Keung, J., Niazi, M., Hussain, S., Ahmad, A.: Systematic literature review and empirical investigation of barriers to process improvement in global software development: client–vendor perspective. Inf. Softw. Technol. **87**, 180–205 (2017)

Khan, A.A., Shameem, M., Nadeem, M., Akbar, M.A.: Agile trends in Chinese global software development industry: fuzzy AHP based conceptual mapping. Appl. Soft Comput. **102**, 107090 (2021)

Khan, A. A. et al.: "AI ethics: an empirical study on the views of practitioners and lawmakers". IEEE Trans. Comput. Soc. Syst. 2023.

Khan, A. A. et al.: "Embracing iterations in quantum software: a vision. In: Proceedings of the 1st international workshop on quantum programming for software engineering, 2022, pp. 11–14

Khan, A. A. et al.: Agile practices for quantum software development: practitioners' perspectives. In: 2023 IEEE international conference on quantum software (QSW), 2023: IEEE, pp. 9–20

Komaki, G., Kayvanfar, V.: Grey Wolf Optimizer algorithm for the two-stage assembly flow shop scheduling problem with release time. J. Comput. Sci. **8**, 109–120 (2015)

Kotsiantis, S.B., Zaharakis, I.D., Pintelas, P.E.: Machine learning: a review of classification and combining techniques. Artif. Intell. Rev. **26**(3), 159–190 (2006)

Kumar, A., Nadeem, M., Banka, H.: Nature inspired optimization algorithms: a comprehensive overview. Evol. Syst. **14**(1), 141–156 (2023)

Lester, M., Guerrero, M., Burge, J.: Using evolutionary algorithms to select text features for mining design rationale. AI EDAM **34**(2), 132–146 (2020)

Mahmoodabadi, M.J., Safaie, A.A., Bagheri, A., Nariman-Zadeh, N.: A novel combination of particle swarm optimization and genetic algorithm for Pareto optimal design of a five-degree of freedom vehicle vibration model. Appl. Soft Comput. **13**(5), 2577–2591 (2013)

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

Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey wolf optimizer. Adv. Eng. Softw. **69**, 46–61 (2014)

Mirjalili, S., and Mirjalili, S.: "Genetic algorithm," Evolut. Algorithms Neural Netw. Theor. Appl., pp. 43–55, 2019

Mitchell, M.: An Introduction to Genetic Algorithms. MIT press, UK (1998)

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

Myers, L. and Sirois, M. J.: "Spearman correlation coefficients, differences between". Encyclop. Stat. Sci., vol. 12, 2004.

Piattini, M., Serrano, M., Perez-Castillo, R., Petersen, G., Hevia, J.L.: Toward a quantum software engineering. IT Professional **23**(1), 62–66 (2021)

Rieffel, E.G., Polak, W.H.: Quantum Computing: A Gentle Introduction. MIT Press, UK (2011)

Shameem, M., Nadeem, M., Zamani, A.T.: Genetic algorithm based probabilistic model for agile project success in global software development. Appl. Soft Comput. **135**, 109998 (2023)

Wang, X., Arcaini, P., Yue, T., Ali, S.: Generating failing test suites for quantum programs with search. In: International symposium on search based software engineering, pp. 9–25. Springer (2021)

Wang, X., Yu, T., Arcaini, P., Yue, T., and Ali, S.: "Mutation-based test generation for quantum programs with multi-objective search". In: Proceedings of the genetic and evolutionary computation conference, 2022, pp. 1345–1353

Weder, B., Barzen, J., Leymann, F., and Vietz, D.: "Quantum software development lifecycle". In: Quantum software engineering: Springer, 2022, pp. 61–83

What is sustainable software development (2023). https://www.linkedin.com/advice/0/what-best-practices-sustainable-software-development#:~:text=Sustainable%20software%20development%20is%20the,user%20needs%20and%20quality%20standards

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in Software Engineering. Springer Science & Business Media, Cham (2012)

Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**(1), 67–82 (1997)

Zhao, J.: "Quantum Software Engineering: Landscapes and Horizons," arXiv preprint arXiv:2007.07047, 2020

Authors and Affiliations

Arif Ali Khan[1] · Muhammad Azeem Akbar[2] · Valtteri Lahtinen[3] · Marko Paavola[4] · Mahmood Niazi[5,6] · Mohammed Naif Alatawi[7] · Shoayee Dlaim Alotaibi[8]

✉ Arif Ali Khan
   arif.khan@oulu.fi

   Muhammad Azeem Akbar
   azeem.akbar@lut.fi

   Valtteri Lahtinen
   valtteri.lahtinen@quanscient.com

   Marko Paavola
   marko.paavola@vtt.fi

   Mahmood Niazi
   mkniazi@kfupm.edu.sa

   Mohammed Naif Alatawi
   alatawimn@ut.edu.sa

   Shoayee Dlaim Alotaibi
   s.alotaibi@uoh.edu.sa

[1] M3S Empirical Software Engineering Research Unit, University of Oulu, 90014 Oulu, Finland

[2] Software Engineering Department, Lappeenranta-Lahti University of Technology, 53851 Lappeenranta, Finland

[3] Quanscient Oy, Tampere, Finland

[4] VTT Technical Research Ctr of Finland Ltd, Oulu, Finland

[5] Department of Information and Computer Science, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia

[6] Interdisciplinary Research Centre for Intelligent Secure Systems, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia

[7] Information Technology Department, Faculty of Computers and Information Technology, University of Tabuk, Tabuk, Saudi Arabia

[8] Department of Artificial Intelligence and Data Science, College of Computer Science and Engineering, University of Hail, Hail, Saudi Arabia