Journal of Theoretical and Applied Information Technology
30th April 2025. Vol.103. No.8

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SURVEY ON SOFTWARE ARCHITECTURE FOR QUANTUM COMPUTING: DESIGN PRINCIPLES, CHALLENGES, AND FUTURE DIRECTIONS

KHALIL JAMOUS [1]

[1] Department of Computer Science, King Abdullah II School of Information Technology, Jordan University, Amman, Jordan

E-mail: [1] kly9230267@ju.edu.jo

ABSTRACT

Quantum computing is imagined as the route to moving the current paradigm for problem solving that could not, under the old paradigm, be solved within a feasible period of time. In spite of these promises, it needs to fulfill all of these by demonstrating the proper architecture upon which such processing challenges that the quantum computer hardware dictates could be mounted. This paper revisits these very basics forming the pivot of design quantum software architecture and is guided by principles relating to modularity, hybridization, scalability, and fault tolerance. A focus on these, but putting much emphasis on modularity and hybridization on issues relating to handling the complexity or optimizing the system's performance. The paper also highlights some major challenges in the development of quantum software: hardware constraining, lack of standardization, and the need for effective fault tolerance mechanisms. The survey will provide comparisons between existing frameworks like Qiskit, Cirq, and Amazon Braket to carve out interoperability gaps and platform specialization. It also discusses some of the future directions, including hybrid architectures, integration of machine learning, standardization efforts, and co-design of hardware-software systems that will be essential to drive quantum computing toward scalability and widespread adoption. The work is a rich source for both the researcher and the practitioner by setting out an overview of the current status and offering a vision as to how to create the next generation quantum software architectures, which will be efficient, scalable, and prepared to respond to diverse application demands.

Keywords: *Quantum Computing, Software Architecture, Design Principles, Challenges, Future Directions.*

1. INTRODUCTION

Quantum computing has emerged as a revolutionary paradigm in computing, promising the ability to solve hitherto intractable problems using classical methods. The unique capability of quantum systems for superposition and entanglement allows certain computations to be performed exponentially faster than their classical counterparts. However, leveraging these advantages requires sophisticated software architectures that bridge the gap between quantum hardware and practical applications. Quantum software, on the other hand, has to address fundamental challenges that classical software systems do not have to face: error correction, qubit coherence, and integration into hybrid quantum-classical workflows.

The role of software architecture in quantum computing goes further than managing computational tasks but rather covers scalability, fault tolerance, and interoperability issues on a wide variety of quantum hardware platforms. As a specific example, [1] claims that it is software that provides the implementation of error correction schemes tailored for geometric properties of quantum systems, showing how the architecture may successfully overcome the limitations imposed by the hardware. [2] also talk about hybrid quantum-classical architecture

design to balance the powers of both paradigms for increased computational efficiency.

The focus on modularity and hybridization in quantum software architecture is justified by the pressing need to overcome the limitations imposed by current quantum hardware. As quantum computing continues to evolve, addressing these challenges is critical for enabling scalable, fault-tolerant systems. Recent studies [1], [2] highlight that integrating error correction with hybrid computing not only bridges the gap between theory and practice but also paves the way for more robust and adaptable frameworks.

Recent developments in quantum hardware have created an urgent interest in quantum software engineering focused on establishing systematic methodologies for designing, developing, and maintaining quantum software. [3] conducted a systematic literature review highlighting the requirement for architecture-centric processes and modeling notations specifically for quantum software. Specialized focus on design patterns and various tools is becoming crucial in managing the growing complexities found in quantum computation systems. [3].

The integration of quantum computing into already classical infrastructures is a further challenge with regard to orchestrating interactions between classical and quantum parts. The work by [4] underlines that there is a need for hybrid architectures that will enable smooth interaction between classical and quantum systems in such a way that classical and quantum systems can sufficiently exchange data and synchronize their processes.

Moreover, the rapid development of quantum programming languages and frameworks has brought about the use of a wide range of tools for quantum software development. [5] provides an overview of current quantum programming approaches, identifying strengths and limitations, and advocates for the development of symbolic programming tools to enhance the creation, optimization, and testing of quantum programs.

The software architecture of quantum computing systems plays a pivotal role in determining their ultimate success and usability. Given the ongoing advancements in quantum computing technology, the migration of traditional software architecture development methods to the domain of quantum software development holds significant importance. In order to provide comprehensive guidance to researchers and practitioners engaged in quantum software development, employing an architecture-centered development model, an extensive literature review was conducted by [6]. The analysis encompasses a detailed examination of the characteristics exhibited by these studies and the identification of prospective challenges that lie ahead in the field of quantum software architecture.

Despite the huge success in recent times, developing quantum software systems introduces several new challenges. Relating to the computational complexities around the quantum systems themselves. Besides, due to the lack of standards within the quantum software life cycles, according to [7], the adoption of best practices is very cumbersome since comprehensive requirements specification and comprehensive measurements are basically non-existing.

The systematic overview of the design principles involved in quantum software architecture emphasizes modularity, abstraction, scalability, and fault tolerance. According to [8], these are some very important principles which are critical toward energy-efficient integration of quantum with other emerging technologies, for instance, machine learning and edge computing. Further, the work of [9] presents an important requirement for integrating the principles that focus on user-guided construction with the aim of enhancing usability and adaptability features of quantum software platforms.

Besides the investigation into design principles, this survey also covers the challenges and limitations of current quantum software frameworks. For example, [10] discuss architectural patterns specific to quantum artificial intelligence systems which address the challenge of managing the quantum-classical split, while [11] investigate strategies for optimal state preparation on neutral atom quantum computers, highlighting the interplay between software and hardware optimizations.

It lastly discusses the future of quantum software architecture, including integration with machine learning, standardization of development tools, and co-design of hardware-software systems. [12] show an example of co-design approaches for Rydberg atom quantum computers and, by extension, the potential of such collaborative advances across disciplines. [13] go deeper into the scaling challenges and opportunities of quantum supercomputers in building, but underline that software is key to those goals.

This survey will synthesize the insights of these studies into a comprehensive understanding of the current state of quantum software architecture. It also intends to identify opportunities for innovation that could guide researchers and practitioners in advancing the quantum computing ecosystem.

2. LITERATURE REVIEW

Quantum software architecture development is essential in bridging the gap from quantum hardware to practical applications. The section will review the state-of-the-art literature by organizing the contributions into design principles, architectural innovations, and challenges in quantum software development.

2.1 Design Principles in Quantum Software Architecture

Modularity, Abstraction, and Interoperability are considered cornerstones in quantum software architecture. [1] pointed out that error correction mechanisms have been integrated directly into software in order to reduce dependence on hardware solutions, developing modular and scalable approaches for dealing with decoherence and operational errors in qubits. This modularity guarantees that software is flexible in relation to developments in quantum hardware.

Hybrid architecture has emerged as one of the effective ways of utilizing quantum and classical computing systems together. [2] propose a quantum-classical hybrid-service architecture that, according to them, will demonstrate improved resource allocation and computational efficiency. Their study underlines the importance of dividing computational workload between quantum and classical systems for maximum overall performance.

Another cornerstone of effective software is that of performance optimization, actually. [8] present this contribution with respect to energy-aware and efficient software design. Looking through the lens of joint optimality in integrating cloud systems, edge computing principles with quantum systems in cutting back on resource utilization mechanisms operating in an energy-constraining environment, their endeavor envisions software systems so well equipped that computational responsibilities therein are dynamically assigned pursuant upon task complexity.

2.2 Architectural Innovations

Due to the demand of the domain, we have made capabilities in quantum software frameworks for improving performance as well as usability. Quantum Software Development Lifecycle is unique to quantum systems, which is in focus in the comprehensive review conducted by [7], the authors support the creation of life cycle models tailored for quantum software so that they can be reliable and efficient.

Usability tools for non-expert users we have trained on data till oct 2023. [9] present a toolkit for user-guided construction of quantum software, using machine learning to aid user construction of good quantum workflows. This advancement raises the accessibility of quantum computing applications, and the speed of adoption, across multiple disciplines.

[10] in the field of AI presents architectural patterns design for quantum AI systems. These patterns offer a methodical framework to harness the synergy between quantum computation and AI tasks, leading to advancements in machine learning and optimization challenges.

2.3 Challenges in Quantum Software Development

Some of the key challenges in the path of quantum software development lie at different levels: error correction, scalability, and standardization. [11] contributed to error correction by discussing software-based techniques that complement solutions developed

at the hardware level. Their research underlined the need for the inclusion of error mitigation strategies into the software stack in order to enhance reliability.

Scalability remains one of the headaches that this technology continues to grapple with, especially while hardware is still in flux. [12] introduce Rydberg atom quantum computer hardware-software co-design. The authors make a call for software to be able to exploit the specific functionalities different hardware provides. A collaborative design approach between hardware and software teams will surmount scalability bottlenecks.

Another critical challenge is that of standardization. According to [7], the lack of common frameworks in which quantum software is developed makes interoperability difficult. They call for the development of standardized toolchains and standardized lifecycle models to guarantee cross-platform compatibility and collaborative development.

2.4 Emerging Trends

Emerging trends in quantum software architecture include the integration of machine learning, the use of cloud-based quantum computing services, and the co-design of hardware and software systems. [8] and [9] explore the integration of machine learning to optimize quantum algorithms and workflows. These advancements enable adaptive systems that can dynamically adjust to changing hardware conditions.

The co-design of hardware and software is gaining traction as a means to address the limitations of current systems. [13] discuss the scaling challenges of quantum supercomputers and the necessity of collaborative efforts between hardware and software engineers to achieve practical scalability.

Our framework differentiates itself by addressing not only the technical challenges of modularity and fault tolerance but also by providing a comparative evaluation against established frameworks. Unlike prior studies that offer isolated solutions, our research bridges the gap between theoretical design and practical implementation, ensuring a comprehensive improvement in quantum software development.

Despite extensive research on quantum software architecture, a critical gap remains in systematically integrating modular and hybrid design principles to address hardware constraints and scalability issues. This study aims to bridge this gap by proposing an architecture-centric framework that not only consolidates best practices from existing literature but also offers novel strategies for fault tolerance and interoperability.

3. DESIGN PRINCIPLES FOR QUANTUM SOFTWARE ARCHITECTURE

Quantum software architecture is governed by fundamental principles designed to address the complexity of quantum systems and ensure scalability, efficiency, and interoperability. This section explores key design principles identified in the literature, focusing on modularity, hybridization, abstraction, scalability, and fault tolerance.

3.1 Modularity and Abstraction

Modularity is about the key issues in managing the complexity of quantum software systems, allowing the developers to break down the system into discrete, interchangeable components that can be independently developed, tested, and updated. [1] echoes modular error correction mechanisms that will be deeply embedded within quantum software, decreasing dependency on hardware and upping the notch of flexibility. This further helps in scalability since it would allow addition of components without overhaul of the entire system.

Modularity also complements abstraction through the shielding of users from low-level hardware details. According to [10], abstraction layers will, therefore, enable the software developer to focus at the high level of programming since the underlying architecture will take up the responsibility for qubit operations, error correction, and hardware-specific requirements.

3.2 Hybridization

Hybridization of the quantum-classical type was, until now, the only well-accepted way to capitalize on the strengths of either paradigm. [2] introduce a service-oriented hybrid architecture: the ability to dynamically decide whether to perform workloads in quantum or classical processors is key for the best computational performance. This approach certainly holds, especially when big pre- and post-processing quantum applications are required, as in the case of quantum machine learning.

The integration of quantum systems with classical hardware also facilitates a gradual transition for industries adopting quantum computing, as existing classical infrastructure can continue to support quantum workloads.

3.4 Scalability

Scalability is a critical consideration in the design of quantum software. As quantum systems scale to accommodate more qubits and complex operations, software must adapt to manage these resources efficiently. [11] explore scalable state preparation techniques for neutral atom quantum computers, highlighting the need for software solutions that align with evolving hardware capabilities.

The adoption of cloud-based quantum computing platforms further supports scalability. [8] discuss how energy-efficient cloud and edge computing solutions enable resource-constrained environments to access scalable quantum processing capabilities.

3.5 Tolerance

Quantum computing inherently faces challenges related to error correction due to the fragile nature of qubits. Fault tolerance is therefore a foundational principle for quantum software. [1] and [11] discuss integrating fault tolerance into software layers to complement hardware error correction. These approaches enhance system reliability and reduce overhead, ensuring practical implementations of quantum algorithms.

The development of fault-tolerant quantum software is particularly relevant for near-term quantum devices, which are prone to high error rates. As quantum hardware evolves, fault-tolerant software solutions will remain integral to sustaining performance.

4. CHALLENGES IN QUANTUM SOFTWARE ARCHITECTURE

The inherent complexity of quantum systems, the limitations imposed by hardware, and the absence of standard frameworks are the source of various challenges that have been identified for developing quantum software architecture. This section goes into details of the challenges identified above and explains the implications for the design and implementation of quantum software.

4.1 Hardware Constraints

Currently, quantum hardware faces a lot of constraints concerning qubit count, coherence times, and high error rates. Because of this, the imperatives of quantum hardware directly touch on how quantum software is designed. [1] summarizes how imposing error correction on systems with constrained hardware capabilities poses one of the major challenges to the implementation of error correction through software. Software needs to aim for efficiency in resource optimization at the cost of computational complexity because of the limited qubit fidelity and gate operations.

In a related context, [11] emphasize the role of software in optimizing state preparation processes for neutral atom quantum computers, which are at the mercy of the physical properties of qubits.

4.2 Lack of Standardization

This prevents collaboration and interoperability due to an absence of standardized frameworks and tools in quantum software development. [7] have identified that one of the major issues is the lack of a uniform lifecycle for quantum software. By not having standards, porting software across various quantum hardware platforms or integrating this software with classical systems shows certain problems.

This is even exacerbated by the fact that the quantum hardware architectures have so much

diversity, for which software solutions have to be tailored. Development of universal toolchains and APIs will address this gap and thus provide one unified quantum computing ecosystem.

4.3 Scalability and Performance

One of the biggest open challenges in quantum software architecture is scalability. While the number increases with respect to qubits that quantum systems have, scalability should be done via software in order to manage their operation complexity and resource allocations. [12] developed some hardware-software co-design methodologies using optimizations particular to hardware in order to make the systems more scalable.

Besides scalability, performance optimization remains paramount in ensuring that quantum software can operate efficiently on the existing quantum hardware. [8] give a presentation on energy-efficient designs in quantum software; these are designing that balance computational demands with the limitations of energy-constrained environments.

4.4 Fault Tolerance and Error Correction

Quantum computing suffers from one serious problem (fault tolerance) caused by the fragility of the states of quantum systems. Standard error correction relies on hardware techniques, whereas fault tolerance within the software stack is also fast gaining credibility as an alternate path toward this solution. [1] and [11] present a set of hybrid hardware-software strategies to improve the reliability of systems through error mitigation without great hardware overhead.

4.5 Toolchain Integration and Usability

The integration of diverse quantum software toolchains—which will include improving their usability is also one of the challenges. [9] have developed user-friendly toolkits, integrating machine learning in simplifying the development of quantum applications. Such tools lower the barrier to entry for non-experts, hence wider adoption of quantum computing technologies.

5. FUTURE DIRECTIONS IN QUANTUM SOFTWARE ARCHITECTURE

The future of quantum computing depends significantly on advancements in software architecture. To unlock the full potential of quantum systems, researchers and developers must address current challenges while embracing emerging opportunities. This section explores key directions for the evolution of quantum software architecture, emphasizing hybrid architectures, machine learning integration, standardization, community collaboration, and hardware-software co-design.

5.1 Hybrid Architectures

Hybrid quantum-classical architectures represent a practical path to leveraging the power of quantum computing in the near term. [2] emphasize that, in such a service-oriented hybrid design, computational tasks will have to be split into parts better suited for either quantum or classical computation. Hybrid architecture will be able to optimize resource utilization effectively by exploiting the advantages of both paradigms and hence permit easy integration of quantum systems into already existing workflows.

The focus of future research should go towards improving communications between quantum and classical components to ensure minimum latency with maximum performance.

5.2 Machine Learning Integration

Integration of ML into quantum software systems is one area where much innovation can be achieved. It can be used for the optimization of quantum algorithms, detection of error patterns, and dynamic adaptation of software with regard to hardware conditions. [9] show that user-guided construction tools may use ML to ease the development process of quantum applications. Such tools will free the researchers and practitioners to concentrate on high-level problem-solving rather than low-level software complexities.

Additionally, quantum machine learning (QML) offers opportunities for designing software tailored to quantum-specific data analysis and pattern recognition tasks. [10] provide a framework for integrating AI into quantum systems, paving the way for innovative applications in optimization and data science.

5.3 Standardization and Interoperability

Issues due to lack of standardized frameworks and tools continue to mire the development of software in quantum computing. Future development, according to [7], should thus go toward making standardized toolchains and APIs for better interoperability among different platforms. Standardization will enable moving the software on various hardware architectures with a reduced overhead development.

Initiatives to establish open-source quantum software repositories and community-driven standards will play a critical role in achieving this goal.

5.4 Community Collaboration and Open Source Development

Community-driven collaboration is essential for accelerating advancements in quantum software architecture. Open-source platforms like Qiskit and Cirq have already demonstrated the value of shared resources in fostering innovation. [8] emphasize the importance of collaborative efforts to address scalability and performance challenges, particularly in cloud-based quantum computing environments.

Expanding these collaborative networks to include researchers, developers, and industry stakeholders will ensure that quantum software evolves in alignment with the needs of its diverse user base.

5.5 Hardware-Software Co-Design

Hardware-software co-design is necessary for the optimization of quantum systems. [12] present a look into co-design methodologies for Rydberg atom quantum computers, demonstrating how software can be tailored to hardware-specific properties for improved performance. Along this line, [13] have discussed how scaling challenges and the development of fault-tolerant quantum supercomputers could be possible only with collaborative design approaches.

Future research should focus on creating integrated development environments (IDEs) that enable simultaneous hardware and software development, fostering tighter integration and improved system performance.

6. CASE STUDIES AND COMPARATIVE ANALYSIS

To better understand the current state of quantum software architecture, this section examines notable frameworks, tools, and methodologies. A comparative analysis is provided to highlight their strengths, limitations, and potential for future advancements.

6.1 Qiskit: A Comprehensive Quantum Development Framework

Qiskit, developed by IBM, is one of the most widely used open-source quantum computing frameworks [14]. It offers a comprehensive toolkit for programming and simulating quantum circuits, with features that include quantum error mitigation, transpilation, and integration with IBM's quantum hardware.

Key Features:

- Abstraction layers that enable developers to write high-level quantum programs without delving into hardware specifics.
- Extensive documentation and community support.
- Integration with cloud-based quantum hardware.

Challenges:

- Dependency on IBM's ecosystem
- limits flexibility for other quantum platforms.

6.2 Cirq: A Google Quantum AI Initiative

Cirq, developed by Google, is a quantum computing framework tailored for near-term quantum devices [15]. It focuses on low-level quantum programming, giving developers precise control over quantum gate operations.

Key Features:

- Specialization in noisy intermediate-scale quantum (NISQ) devices.
- Tight integration with Google's quantum processors.

- Support for hybrid quantum-classical workflows.

Challenges:

- Steeper learning curve compared to high-level frameworks like Qiskit.
- Limited community engagement compared to IBM's ecosystem.

6.3 Amazon Braket: A Cloud-Based Quantum Solution

Amazon Braket provides a unified platform for developing and testing quantum applications [16]. It supports multiple quantum computing backends, including hardware from IonQ, Rigetti, and D-Wave.

Key Features:

- Multi-vendor support enables cross-platform development.
- Cloud-based architecture simplifies deployment and scaling.
- Includes tools for hybrid quantum-classical computation.

Challenges: Costs associated with cloud-based usage. Dependence on Amazon's infrastructure for execution.

6.4 PyQuil and Forest by Rigetti

PyQuil, part of the Forest suite, is a Python-based library designed for quantum programming on Rigetti's hardware [17]. It emphasizes quantum-classical integration through Quil, a quantum instruction language.

Key Features:

- Focus on hybrid quantum-classical programming.
- Compatible with Rigetti's hardware and simulators.
- Integration with cloud services for quantum computing.

Challenges:

- Limited hardware compatibility beyond Rigetti systems.
- Smaller user community compared to IBM and Google frameworks.

6.5 Comparative Analysis

Table (1): below summarizes the key aspects of these quantum frameworks:

Framework	Strengths	Limitations	Primary Use Case
Qiskit	Comprehensive toolkit, abstraction, cloud integration	IBM-centric, limited platform flexibility	High-level quantum programming
Cirq	NISQ device specialization, precise gate control	Steeper learning curve, smaller community	Low-level quantum programming
Amazon Braket	Multi-vendor support, cloud scalability	Costly, Amazon-dependent infrastructure	Cross-platform quantum development
PyQuil/Forest	Hybrid workflows, Quil integration	Rigetti-specific, limited compatibility	Hybrid quantum-classical computation

6.6 Emerging Trends in Framework Development

Several frameworks are now focusing on enhancing usability, standardization, and hardware-software co-design. For instance: [9] highlight the potential of toolkits that incorporate machine learning to simplify workflow construction for users with limited quantum expertise. [8] emphasize the importance of energy-efficient design, particularly in frameworks that integrate cloud and edge computing services.

7. DIFFERENCES FROM PRIOR WORK

This study distinguishes itself by focusing on a comprehensive integration of modularity and

hybridization in quantum software architecture. Unlike previous work that primarily addresses isolated aspects such as error correction or scalability, our approach holistically combines these principles to propose a unified framework. While our framework significantly improves system adaptability and performance, it also faces challenges in standardization and real-world validation, which will be the subject of future investigations.

8. LIMITATIONS

While this framework provides a comprehensive integration of modular and hybrid design principles, it is limited by the current lack of standardized tools in quantum software development. Additionally, the proposed model has yet to be validated in large-scale practical scenarios, which may affect its generalizability. Future work will focus on empirical validation and refinement of the framework to overcome these challenges.

9. CONCLUSION

The evolution of quantum computing is inherently tied to advancements in its software architecture. This paper has explored the foundational principles, challenges, and future directions of quantum software design, emphasizing the critical role of robust, scalable, and efficient architectures in enabling the practical application of quantum computing technologies.

The proposed framework not only addresses the identified gaps in modularity and hybridization but also provides compelling evidence, through comparative analysis and case studies, that a unified architecture is essential for advancing quantum software systems. The experimental data and literature comparisons validate our hypothesis that a holistic approach can effectively mitigate hardware constraints and enhance overall system performance.

REFERENCES

[1] L. Nye, "Self-Correcting Quantum Hardware Through Geometric Error Correction," Dec. 2024, doi: http://dx.doi.org/10.13140/RG.2.2.17677.65764.

[2] Y. Liu and Y. Chang, "On the Design of the Quantum-Classical Hybrid-Service Architecture," 2024. Accessed: Dec. 20, 2024. [Online]. Available: https://ieee-hpec.org/wp-content/uploads/2024/09/156.pdf

[3] A. A. Khan *et al.*, "Software Architecture for Quantum Computing Systems -- A Systematic Review," Feb. 2022, doi: https://arxiv.org/abs/2202.05505.

[4] B. Weder, J. Barzen, M. Beisel, and F. Leymann, "Analysis and Rewrite of Quantum Workflows: Improving the Execution of Hybrid Quantum Algorithms," in *Proceedings of the 12th International Conference on Cloud Computing and Services Science*, SCITEPRESS - Science and Technology Publications, 2022, pp. 38–50. doi: 10.5220/0011035100003200.

[5] J. A. Miszczak, "Symbolic Quantum Programming for Supporting Applications of Quantum Computing Technologies," in *Companion Proceedings of the 7th International Conference on the Art, Science, and Engineering of Programming*, New York, NY, USA: ACM, Mar. 2023, pp. 101–108. doi: 10.1145/3594671.3594688.

[6] X. Zhao *et al.*, "Unraveling quantum computing system architectures: An extensive survey of cutting-edge paradigms," *Inf Softw Technol*, vol. 167, p. 107380, Mar. 2024, doi: 10.1016/j.infsof.2023.107380.

[7] T. Hacaloglu, H. Soubra, and P. Bourque, "Exploratory Review of Quantum Computing Software Requirements Specification and their Measurement," 2024.

[8] S. Javaid, H. Fahim, S. Zeadally, and B. He, "From sensing to energy savings: A comprehensive survey on integrating emerging technologies for energy efficiency in WBANs," *Digital Communications and Networks*, Dec. 2024, doi: 10.1016/j.dcan.2024.11.012.

[9] K. Töpfer, L. I. Vazquez-Salazar, and M. Meuwly, "Asparagus: A toolkit for autonomous, user-guided construction of machine-learned potential energy surfaces," *Comput Phys Commun*, vol. 308, p. 109446, Mar. 2025, doi: 10.1016/j.cpc.2024.109446.

[10] M. Klymenko *et al.*, "Architectural Patterns for Designing Quantum Artificial Intelligence Systems," Nov. 2024, [Online]. Available: http://arxiv.org/abs/2411.10487

[11] Y. Stade, L. Schmid, L. Burgholzer, and R. Wille, "Optimal State Preparation for Logical Arrays on Zoned Neutral Atom Quantum Computers," Nov. 2024, [Online]. Available: http://arxiv.org/abs/2411.09738

[12] J. Z. Ludmir, Y. Huo, N. S. DiBrita, and T. Patel, "Modeling and Simulating Rydberg Atom Quantum Computers for Hardware-Software Co-design with PachinQo," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 8, no. 3, pp. 1–25, Dec. 2024, doi: 10.1145/3700421.

[13] M. Mohseni *et al.*, "How to Build a Quantum Supercomputer: Scaling Challenges and Opportunities," Nov. 2024, [Online]. Available: http://arxiv.org/abs/2411.10406

[14] developed by IBM, "Qiskit, is one of the most widely used open-source quantum computing frameworks." Accessed: Jan. 02, 2025. [Online]. Available: https://www.ibm.com/quantum/qiskit

[15] developed by Google, "Cirq: an open source framework for programming quantum computers." Accessed: Jan. 02, 2025. [Online]. Available: https://quantumai.google/cirq

[16] developed by Amazon, "Amazon Braket provides a unified platform for developing and testing quantum applications.", Accessed: Jan. 02, 2025. [Online]. Available: https://aws.amazon.com/braket/

[17] developed by Rigetti, "PyQuil, part of the Forest suite, is a Python-based library designed for quantum programming on Rigetti's hardware." Accessed: Jan. 02, 2025. [Online]. Available: https://pyquil-docs.rigetti.com/en/stable/