Self-Adaptive Quantum and Classical Software Systems

Antonio Brogi
University of Pisa
Pisa, Italy
antonio.brogi@unipi.it

Schahram Dustdar
Distributed Systems Group
TU Wien
Vienna, Austria
dustdar@dsg.tuwien.ac.at

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

Hausi Muller
Faculty of Engineering
University of Victoria
Victoria, Canada
hausi@uvic.ca

Juan Manuel Murillo
Universidad de Extremadura
Cáceres, Spain
juanmamu@unex.es

Ricardo Pérez-Castillo
Faculty of Social Sciences & IT
University of Castilla-La Mancha
Talavera de la Reina, Spain
ricardo.pdelcastillo@uclm.es

Antonio Ruiz-Cortés
Lab, I3US Institute
Universidad de Sevilla
Seville, Spain
aruiz@us.es

Shinobu Saito
NTT Computer and
Data Science Laboratories
Tokyo, Japan
shinobu.saito@ntt.com

Ulrike Stege
Faculty of Engineering
University of Victoria
Victoria, Canada
ustege@uvic.ca

***Abstract*—Quantum computing has the potential to revolutionize various industries by solving complex problems beyond the capabilities of classical systems. However, the practical realization of these advancements depends on robust hybrid software capable of adapting to highly dynamic execution environments. Fluctuating resource availability, quantum noise, and hardware constraints in distributed systems present significant challenges, making self-adaptation mechanisms essential for optimizing performance, ensuring reliability, and maintaining scalability. Traditional static approaches to software design are insufficient in such unpredictable settings, requiring runtime adaptation strategies that continuously monitor and adjust system behavior. Frameworks such as autonomic computing models and dynamic software architectures provide viable solutions for managing uncertainty and improving computational efficiency. By integrating self-adaptive capabilities, hybrid quantum software can dynamically respond to execution constraints, enhance system resilience, and support real-world applications, ensuring that quantum computing can effectively complement classical computing to address the growing demands of high-performance computation.**

***Index Terms*—Self-adaptive systems, Quantum software, Hybrid software systems, Distributed computing.**

I. Introduction

Quantum computing is set to revolutionize industries by solving complex problems in finance, logistics, and cybersecurity, where classical systems struggle [1]. Its ability to optimize, simulate, and accelerate computations will drive breakthroughs in many sectors [2], [3]. However, realizing these promising applications depends on the development of robust quantum software [4], which is essential to harness quantum hardware efficiently, design scalable algorithms, and integrate quantum and classical systems. Thus, quantum computing represents the next frontier, complementing classical software systems to meet the demands of high-performance computing. As investment and research grow, hybrid (quantum-classical) software solutions will bridge current limitations, paving the way for practical applications and future innovations that redefine what is computationally possible.

As computing systems continue to evolve towards greater heterogeneity, hybrid software systems have emerged as a key paradigm for integrating and exploiting the strengths of diverse computational models [5], [6]. However, these systems must operate in highly dynamic environments, where resource availability, computational limitations, and execution conditions vary unpredictably, posing significant challenges to their efficient and reliable operation [7]. Distributed quantum computing further amplifies these challenges, as computations must be coordinated across multiple quantum and classical nodes, introducing additional complexity in managing resource constraints and execution variability [8]. Moreover, the gap between academic research and industry practices further complicates hybrid and quantum software development. Although academic efforts in debugging and testing primarily address a narrow set of quantum-specific challenges, industry practitioners face a broader spectrum of issues, including software integration, interoperability, and deployment in real-world infrastructures [9].

To address these challenges, self-adaptation mechanisms [10] must be integrated into hybrid software systems to enable dynamic performance optimization, reliability enhancement, and scalability. Given the fluctuating availability of quantum resources, unpredictable noise levels, and evolving software-hardware interactions, among other challenges, adaptive frameworks can help mitigate these uncertainties by continuously monitoring system conditions and adjusting execution strategies accordingly.

A self-adaptive system (SAS) [11], [12] is a system capable of modifying its behavior at runtime in response to changes in its context or environment to continuously achieve its goals. This adaptability is essential in hybrid distributed computing, where the increasing functionality and interrelationship of systems make it impractical to pre-define all possible behaviors at design time. In the context of hybrid quantum software, self-adaptation mechanisms [10] are crucial to optimize performance, improve reliability, and ensure scalability. Given the fluctuating availability of quantum resources, unpredictable noise levels, and evolving software-hardware interactions, adaptive frameworks provide a means to dynamically monitor system conditions and adjust execution strategies accordingly, ensuring robustness in rapidly changing computational environments.

Although there are some preliminary works addressing some of these challenges [13], [14], this position paper advocates the necessity of self-adaptation in hybrid quantum-classical software systems, emphasizing that dynamic adaptation is essential to cope with fluctuating execution conditions in quantum, distributed systems, quantum resource constraints, and reliability challenges. Thus, future hybrid quantum systems should integrate self-adaptive mechanisms, such as MAPE-K (Monitor, Analyze, Plan, Execute, and Knowledge) [15], [16] and DSPL (Dynamic Software Product Lines) [17], to ensure efficiency, scalability, and robustness in dynamically changing environments. Given the inherent variability in quantum resources, fluctuating execution conditions, and reliability challenges, the adoption of structured adaptation frameworks is essential to optimize performance and resource allocation.

This article originates from the ideas developed within the working group on Self-Adaptive Hybrid Systems at the Dagstuhl Seminar on Quantum Software Engineering [18]. This paper serves as a call to action for researchers to explore adaptive architectures, AI-driven optimizations, and new frameworks for hybrid quantum computing. Moreover, the ideas presented herein are intended not only to contribute to the academic research community but also to provide insights and potential directions for the industry, where self-adaptive hybrid quantum systems can drive advancements in real-world applications.

The remainder of this paper is structured as follows. Section II discusses factors that make self-adaptiveness necessary and explains different drivers for managing self-adaptive hybrid software. Section III introduces how existing approaches could be adapted to design self-adaptive hybrid software systems, as well as some limitations in this context. Finally, Section IV draws main conclusions and implications.

II. Factors and drivers for adaptation in Hybrid Software Systems

Hybrid, distributed quantum systems operate in dynamic and resource-constrained environments, requiring continuous adaptation to maintain efficiency, scalability, and reliability. Adaptation in these systems can be categorized into three primary dimensions (see Fig. 1): quality-driven, resource-driven, and cost-driven adaptation (see subsections II-A to II-C). Understanding the interplay between these factors is crucial for designing effective self-adaptive quantum software (cf. subsection II-D).

A. Quality-Driven Adaptation

Quality-driven adaptation ensures that quantum computations maintain accuracy and reliability despite hardware limitations. Since quantum systems are inherently noisy, adaptation techniques must mitigate errors and enhance result precision. Approaches to quality-driven adaptation include:

- Selecting error-mitigation techniques based on real-time noise profiling [19].
- Adjusting measurement strategies to optimize readout fidelity [20].
- Dynamically reconfiguring quantum circuits to minimize decoherence effects [21].
- Adapting quantum error correction techniques based on available qubits and noise levels [22].
- Employing post-processing strategies to filter out erroneous results [19].

B. Resource-Driven Adaptation

Resource-driven adaptation addresses the constraints imposed by the availability and capacity of quantum and classical resources. Given the limited number of high-fidelity qubits, coherence times, and quantum-classical communication bandwidth, adaptation mechanisms must allocate resources dynamically to maximize system performance. Key strategies include:

- Adapting quantum circuit execution based on qubit availability and error rates [23].
- Load balancing between quantum processors and classical co-processors to optimize hybrid execution [24].
- Dynamically selecting noise-aware compilation strategies to improve circuit fidelity [25].
- Adjusting execution schedules to optimize shared quantum resources for multiple users [26].
- Selecting optimal qubit mappings to reduce cross-talk and gate errors [21].

C. Cost-Driven Adaptation

Cost-driven adaptation focuses on optimizing the financial and computational expenses associated with quantum computing. Access to quantum resources is costly, particularly when utilizing cloud-based quantum computing services. Adaptive mechanisms must balance the trade-offs between execution time, the number of quantum operations (gates), and overall economic efficiency. Examples of cost-driven adaptation include:

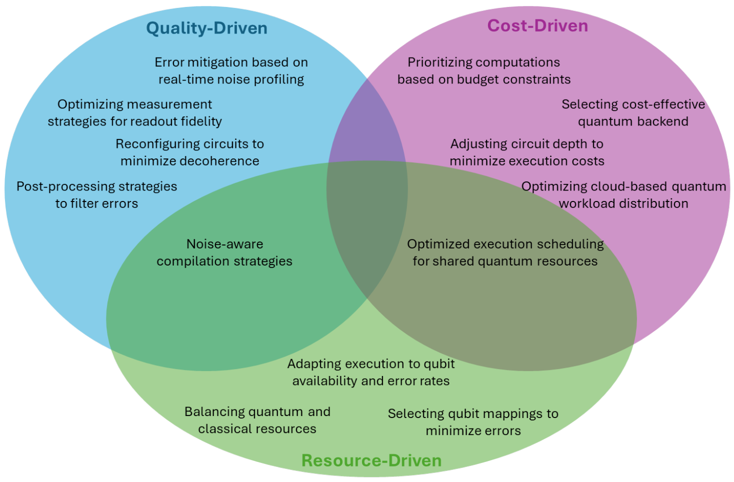
Quality-Driven
Error mitigation based on real-time noise profiling
Optimizing measurement strategies for readout fidelity
Reconfiguring circuits to minimize decoherence
Post-processing strategies to filter errors
Cost-Driven
Prioritizing computations based on budget constraints
Selecting cost-effective quantum backend
Adjusting circuit depth to minimize execution costs
Optimizing cloud-based quantum workload distribution
Noise-aware compilation strategies
Optimized execution scheduling for shared quantum resources
Adapting execution to qubit availability and error rates
Balancing quantum and classical resources
Selecting qubit mappings to minimize errors
Resource-Driven

Fig. 1. Dimensions and critical factors for self-adaptiveness in hybrid software systems.

- Selecting the most cost-effective quantum backend based on pricing and availability [27].
- Dynamically adjusting circuit depth to minimize execution costs.
- Efficiently distributing quantum workloads to optimize cloud service expenses [28].
- Prioritizing computations based on budget constraints.
- Reducing redundant computations to minimize unnecessary quantum executions [29].

D. Interdependencies Between Quality, Resources, and Cost in Adaptive Quantum Systems

While resource-driven and cost-driven adaptation may seem similar, they address distinct aspects of quantum computing. Resource-driven adaptation focuses on the physical and computational limitations of available hardware, ensuring optimal utilization of quantum and classical components. In contrast, cost-driven adaptation emphasizes financial efficiency, aiming to minimize execution expenses while maintaining computational feasibility. While both dimensions influence each other, they may sometimes require different optimization strategies. For instance, using fewer qubits can reduce costs but may lead to longer execution times due to increased circuit depth.

These trade-offs become evident in complex quantum-classical problems. In order to illustrate it, let us think of a well-known logistic problem, the Travelling Salesperson Problem (TSP) [30], where a quantum algorithm seeks the shortest path among multiple cities. As the problem size increases, adaptation mechanisms must balance quality, resource allocation, and cost. A resource-driven approach might partition the problem into smaller subproblems using classical clustering techniques and then solve them on available quantum hardware, optimizing qubit utilization while maintaining computational feasibility. However, if quality is prioritized, the system might select a quantum backend with a higher number of qubits and lower noise levels, ensuring greater accuracy but at a higher cost. Conversely, a cost-driven approach might favor executing the problem on less expensive, lower-fidelity hardware, accepting potential accuracy reductions to minimize expenses.

A potential solution to balancing these trade-offs is the shot-wise distribution of quantum computations [8], where measurement shots are dynamically allocated across different quantum backends. This approach enables the system to adjust execution strategies in real time, optimizing computational quality while controlling costs and resource usage. By implementing self-adaptive mechanisms capable of continuously evaluating and adjusting these dimensions, hybrid quantum software can achieve optimal efficiency, reliability, and economic viability in real-world applications.

III. Potential Approaches for Designing Self-Adaptive Hybrid Software Systems

In addition to the factors drivers presented in Section II, the hybrid nature of quantum-classical systems presents unique challenges that require integrated adaptation mechanisms. Hybrid adaptation considers the interactions between quantum and classical components, ensuring that both paradigms operate efficiently together. Important aspects of hybrid adaptation include adapting data transfer strategies between quantum and classical processors to reduce bottlenecks; dynamically selecting classical approximation methods when quantum resources are limited; or managing hybrid workflows to ensure seamless execution across different computational platforms; among others.

This section first explores MAPE-K as the traditional self-adaptive solution, and then discusses its limitations and challenges for hybrid software systems. These challenges are then related to the investigation of potential solutions in the design and implementation of self-adaptive hybrid systems.

A. MAPE-K as traditional self-adaptive solution

The MAPE-K [15], [16] cycle is a well-established framework for self-adaptive systems that provides a structured way

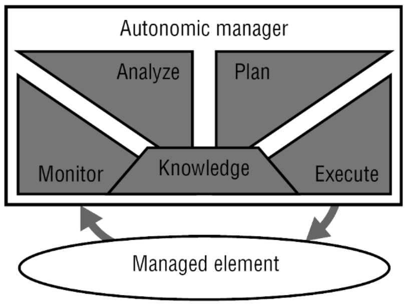

Fig. 2. The MAPE-K Cycle as defined in [11]

to dynamically adjust system behavior based on environmental changes. It consists of four key phases (see Fig. 2):

- **Monitor**: Collects data from the system and its execution environment, including performance metrics, resource usage, and quantum processor availability.
- **Analyze**: Evaluates the collected data to detect anomalies, predict trends, or identify potential optimizations, such as adjusting workload distribution between quantum and classical resources.
- **Plan**: Based on the analysis, the system determines adaptation strategies to optimize performance, cost, or resource utilization. This may include selecting an alternative quantum backend, adjusting circuit depth, or redistributing computational tasks.
- **Execute**: Applies the planned adaptation actions by modifying system configurations, adjusting quantum job scheduling, or reallocating resources.
- **Knowledge**: Serves as a shared repository of system insights, past adaptation actions, and learned strategies to enhance future decision-making.

Applying the MAPE-K cycle to hybrid quantum-classical systems enables continuous self-adaptation, ensuring that the system dynamically responds to fluctuations in resource availability, execution conditions, and cost constraints. Key benefits of using MAPE-K in this context include:

- Automated adaptation: Enables real-time reconfiguration of quantum and classical workflows to optimize performance and reliability.
- Optimized resource utilization and cost efficiency: Dynamically adjusts computational load distribution between quantum and classical processors based on current constraints while managing quantum resource expenses by selecting the most cost-effective execution strategies.
- Enhanced system resilience: Detects and mitigates potential issues, such as high error rates in quantum computations, by adapting execution strategies accordingly.
- Scalability: Provides a structured and modular approach that can be extended as hybrid quantum computing infrastructures evolve.

By leveraging the MAPE-K cycle, hybrid software systems can achieve higher adaptability, robustness, and efficiency, ensuring that they remain responsive to the dynamic nature of quantum and classical computing environments.

B. Limitations of the traditional self-adaptive approach for Hybrid Software Systems

While MAPE-K provides a structured approach to self-adaptation, its direct application to hybrid quantum-classical systems presents several limitations that must be addressed.

a) ***Rigidity of the adaptation cycle:*** MAPE-K follows a fixed sequential cycle (Monitor $\rightarrow$ Analyze $\rightarrow$ Plan $\rightarrow$ Execute), which can be inefficient in scenarios requiring rapid or flexible adaptation. In dynamic environments such as distributed quantum systems, strict adherence to this cycle may introduce delays or prevent proactive adaptation. Alternative models such as event-driven adaptation and just-in-time adaptations can allow for more flexible decision-making.

b) ***Latency in decision-making:*** The sequential nature of MAPE-K can introduce high latency in responding to changes. Quantum computing operates on extremely short timescales, where decoherence effects and noise fluctuations require rapid adaptation. Hybrid approaches combining offline planning (precomputed strategies) with online adaptation (real-time adaptations) can help improve responsiveness.

c) ***Complexity in knowledge management:*** The Knowledge component is crucial for improving adaptation over time, but MAPE-K does not specify how this knowledge should be structured, stored, or leveraged. This limitation becomes particularly evident in *Quantum DevOps* [31], where the operational complexity of hybrid quantum-classical systems demands efficient management of adaptation knowledge, including quantum execution metadata, error correction strategies, and hybrid workflow optimization. Efficient knowledge management requires techniques such as machine learning, reinforcement learning, and big data analytics, which are not inherently included in MAPE-K. Moreover, the rigidity of traditional knowledge structures can hinder adaptation in dynamic and distributed quantum-classical environments. Another promising approach to overcoming these limitations is the use of Dynamic Software Product Lines (DSPLs) [17], [32], which enable runtime reconfiguration by dynamically selecting and composing adaptation strategies based on system changes. Integrating DSPLs alongside AI-driven decision-making and continuous learning techniques can enhance adaptation effectiveness, ensuring more flexible and scalable self-adaptation in hybrid quantum-classical systems.

d) ***Lack of scalability in distributed systems:*** MAPE-K was originally designed for monolithic systems, making its direct application to distributed environments challenging. In distributed quantum computing, managing multiple MAPE-K loops across different quantum and classical resources may lead to communication overhead and synchronization issues. Additionally, coordinating multiple self-adaptive agents requires decentralized or hierarchical control mechanisms. Implementing multi-agent MAPE-K architectures, where different agents cooperate without relying on a single control point, and adopting hierarchical adaptation, where local MAPE-K loops handle low-level adaptations while a higher-level control optimizes the system globally, can address these challenges. Quantum, distributed systems introduce additional complexity due to the need for coordinating computations across geographically dispersed quantum and classical nodes [8]. Ensuring synchronization, managing network-induced latencies, and balancing workloads efficiently become crucial challenges that require enhanced adaptation mechanisms beyond what MAPE-K traditionally offers [33].

*e) **Difficulty in integrating modern adaptation techniques**:* MAPE-K does not inherently support modern adaptation techniques, such as AI-based optimization, big data analytics, or cloud-native architectures. In hybrid quantum-classical systems, adaptation strategies must integrate both quantum and classical decision-making processes. Extending MAPE-K with AI-driven self-learning adaptation policies that dynamically adjust based on real-time data can improve its applicability.

*f) **Sensitivity of monitoring:**:* The monitoring step of the MAPE-K loop must be performed with care in quantum systems. While final probabilistic outcomes of quantum computations can be collected and analyzed, the observation of intermediate states introduces side effects.

*g) **Latency in decision-making:**:* Reducing the latency of the MAPE-K loop is, however, not an immediately urgent limitation to overcome, given the high queuing delays in currently available NISQ devices.

Table I summarizes the challenges presented in this section. For MAPE-K to be effectively applied in distributed quantum computing, these limitations must be addressed. Possible improvements include making the cycle more dynamic by allowing parallel execution of some phases to speed up adaptation, reducing decision-making latency by combining precomputed strategies with real-time adjustments, enhancing knowledge management by leveraging machine learning to refine adaptation policies over time, and implementing distributed architectures to enable decentralized control and coordination of multiple self-adaptive agents. By addressing these challenges, MAPE-K can serve as a viable foundation for adaptive hybrid quantum-classical systems, ensuring efficient resource utilization and improved computational reliability.

IV. Conclusions

The increasing integration of quantum and classical computing within hybrid systems necessitates robust self-adaptation mechanisms to address execution variability, resource constraints, and noise fluctuations in quantum hardware. Traditional static approaches to software design and management are insufficient in such dynamic, distributed environments, requiring real-time monitoring and adaptive strategies to optimize performance, reliability, and scalability, among other relevant properties in a quality-, resource-, and cost-driven manner.

Hybrid software systems require real-time workload adaptations to adapt to the mentioned factors. Although there exist some traditional approaches, such as MAPE-K, for managing self-adaptiveness, the inherent latency in the decision-making process may hinder the responsiveness needed for hybrid software execution. Real-time monitoring, predictive analytics, and dynamic adaptation policies are necessary to address these constraints.

Several challenges remain open in self-adaptive hybrid software. Reducing decision-making latency, integrating AI-driven adaptation strategies, ensuring scalability in distributed quantum computing, and balancing computational efficiency with cost constraints are key areas that require further research. Overcoming these challenges is crucial to improving the adaptability and efficiency of hybrid systems, allowing quantum computing to be leveraged more effectively in real-world applications. This paper aims to stimulate discussion and further investigation in this field, encouraging new approaches and solutions for the development of self-adaptive hybrid software.

Acknowledgment

The authors thank the organizers and participants of Dagstuhl Seminar on Quantum Software Engineering (https://www.dagstuhl.de/24512) for the stimulating discussions, especially within the Self-Adaptive Hybrid Systems working group. R. Pérez-Castillo is supported by the project SMOOTH (PID2022-137944NB-I00) funded by MICIU/AEI/10.13039/501100011033 / PRTR, EU.

References

[1] Y. Kim, A. Eddins, S. Anand, K. X. Wei, E. Van Den Berg, S. Rosenblatt, H. Nayfeh, Y. Wu, M. Zaletel, K. Temme *et al.*, “Evidence for the utility of quantum computing before fault tolerance,” *Nature*, vol. 618, no. 7965, pp. 500–505, 2023.

[2] A. Bayerstadler, G. Becquin, J. Binder, T. Botter, H. Ehm, T. Ehmer, M. Erdmann, N. Gaus, P. Harbach, M. Hess *et al.*, “Industry quantum computing applications,” *EPJ Quantum Technology*, vol. 8, no. 1, p. 25, 2021.

[3] A. I. S. Alsalman, “Accelerating quantum computing readiness: Risk management and strategies for sectors,” *Journal of Quantum Information Science*, vol. 13, no. 2, pp. 33–44, 2023.

[4] J. M. Murillo, J. Garcia-Alonso, E. Moguel, J. Barzen, F. Leymann, S. Ali, T. Yue, P. Arcaini, R. Pérez-Castillo, I. García Rodríguez de Guzmán, M. Piattini, A. Ruiz-Cortés, A. Brogi, J. Zhao, A. Miranskyy, and M. Wimmer, “Quantum software engineering: Roadmap and challenges ahead,” *ACM Trans. Softw. Eng. Methodol.*, Jan. 2025, just Accepted. [Online]. Available: https://doi.org/10.1145/3712002

[5] B. Weder, J. Barzen, M. Beisel, and F. Leymann, “Provenance-preserving analysis and rewrite of quantum workflows for hybrid quantum algorithms,” *SN Computer Science*, vol. 4, no. 3, p. 233, 2023. [Online]. Available: https://doi.org/10.1007/s42979-022-01625-9

[6] R. Pérez-Castillo, M. A. Serrano, and M. Piattini, “Software modernization to embrace quantum technology,” *Advances in Engineering Software*, vol. 151, p. 102933, 2021. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0965997820309790

[7] S. Sepúlveda, R. Pérez-Castillo, and M. Piattini, “A software product line approach for developing hybrid software systems,” *Information and Software Technology*, vol. 178, p. 107625, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950584924002301

[8] G. Bisicchia, J. García-Alonso, J. M. Murillo, and A. Brogi, “Distributing quantum computations, by shots,” in *Service-Oriented Computing*, F. Monti, S. Rinderle-Ma, A. Ruiz Cortés, Z. Zheng, and M. Mecella, Eds. Cham: Springer Nature Switzerland, 2023, pp. 363–377.

[9] J. Zappin, T. Stalnaker, O. Chaparro, and D. Poshyvanyk, “Bridging the quantum divide: Aligning academic and industry goals in software engineering,” 2025. [Online]. Available: https://arxiv.org/abs/2502.07014

TABLE I
SUMMARY OF MAPE-K LIMITATIONS IN SELF-ADAPTIVE HYBRID QUANTUM SYSTEMS

Limitation/Challenge	Impact on Hybrid Quantum Software	Potential Solution
Rigidity of the adaptation cycle	Sequential cycle limits proactive adaptation in dynamic quantum environments.	Use event-driven adaptation and just-in-time adaptations.
Latency in decision-making	High response latency conflicts with rapid adaptation needs in quantum computing.	Combine offline planning with real-time adaptation.
Complexity in knowledge management	Lacks structured knowledge management, critical in Quantum DevOps.	Employ Dynamic Software Product Lines (DSPLs) and AI-driven learning.
Lack of scalability in distributed systems	Challenges in scalability, synchronization, and workload balancing in distributed systems.	Implement multi-agent architectures and hierarchical adaptation.
Difficulty in integrating modern adaptation techniques	Does not support AI-based optimization or modern adaptation techniques.	Integrate AI-driven self-learning policies for dynamic adjustment.

[10] T. Wong, M. Wagner, and C. Treude, "Self-adaptive systems: A systematic literature review across categories and domains," *Information and Software Technology*, vol. 148, p. 106934, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950584922000854

[11] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[12] P. Oreizy, N. Medvidovic, and R. Taylor, "Architecture-based runtime software evolution," in *Proceedings of the 20th International Conference on Software Engineering*, 1998, pp. 177–186.

[13] Z. Jiang, Z. Du, S. Ruan, J. Chen, Y. Wang, L. Cheng, R. Buyya, and Y. Mao, "Resource-efficient and self-adaptive quantum search in a quantum-classical hybrid system," 2024. [Online]. Available: https://arxiv.org/abs/2405.04490

[14] A. Setty, R. Abdusalamov, and F. Motzoi, "Self-adaptive physics-informed quantum machine learning for solving differential equations," *Machine Learning: Science and Technology*, vol. 6, no. 1, p. 015002, Jan. 2025. [Online]. Available: http://dx.doi.org/10.1088/2632-2153/ada3ab

[15] M. Hachicha, R. B. Halima, and A. H. Kacem, "Modeling and verifying self-adaptive systems: A refinement approach," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2016, pp. 003 967–003 972.

[16] J. Oh, C. Raibulet, and J. Leest, "Analysis of mape-k loop in self-adaptive systems for cloud, iot and cps," in *Service-Oriented Computing – ICSOC 2022 Workshops*, J. Troya, R. Mirandola, E. Navarro, A. Delgado, S. Segura, G. Ortiz, C. Pautasso, C. Zirpins, P. Fernández, and A. Ruiz-Cortés, Eds. Cham: Springer Nature Switzerland, 2023, pp. 130–141.

[17] B. I. Moritani and J. Lee, "An approach for managing a distributed feature model to evolve self-adaptive dynamic software product lines," in *Proceedings of the 21st International Systems and Software Product Line Conference - Volume B*, ser. SPLC '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 107–110. [Online]. Available: https://doi.org/10.1145/3109729.3109743

[18] S. Ali, J. Barzen, A. Delgado, H. A. Müller, and J. M. Murillo, "Dagstuhl seminar 24512: Quantum software engineering," Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024, dagstuhl Seminar 24512, December 15–20, 2024. [Online]. Available: https://www.dagstuhl.de/24512

[19] K. Temme, S. Bravyi, and J. M. Gambetta, "Error mitigation for short-depth quantum circuits," *Phys. Rev. Lett.*, vol. 119, p. 180509, Nov 2017. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.119.180509

[20] F. B. Maciejewski, Z. Zimborás, and M. Oszmaniec, "Mitigation of readout noise in near-term quantum devices by classical post-processing based on detector tomography," *Quantum*, vol. 4, p. 257, Apr. 2020. [Online]. Available: https://doi.org/10.22331/q-2020-04-24-257

[21] P. Murali, J. M. Baker, A. Javadi-Abhari, F. T. Chong, and M. Martonosi, "Noise-adaptive compiler mappings for noisy intermediate-scale quantum computers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 1015–1029. [Online]. Available: https://doi.org/10.1145/3297858.3304075

[22] S. S. Tannu and M. K. Qureshi, "Not all qubits are created equal: A case for variability-aware policies for nisq-era quantum computers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 987–999. [Online]. Available: https://doi.org/10.1145/3297858.3304007

[23] P. D. Nation, H. Kang, N. Sundaresan, and J. M. Gambetta, "Scalable mitigation of measurement errors on quantum computers," *PRX Quantum*, vol. 2, p. 040326, Nov 2021. [Online]. Available: https://link.aps.org/doi/10.1103/PRXQuantum.2.040326

[24] A. McCaskey, E. Dumitrescu, D. Liakh, and T. Humble, "Hybrid programming for near-term quantum computing systems," in *2018 IEEE international conference on rebooting computing (ICRC)*. IEEE, 2018, pp. 1–12.

[25] G. Li, Y. Ding, and Y. Xie, "Tackling the qubit mapping problem for nisq-era quantum devices," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 1001–1014. [Online]. Available: https://doi.org/10.1145/3297858.3304023

[26] I. Henao, J. P. Santos, and R. Uzdin, "Adaptive quantum error mitigation using pulse-based inverse evolutions," *npj Quantum Information*, vol. 9, no. 1, p. 120, 2023. [Online]. Available: https://doi.org/10.1038/s41534-023-00785-7

[27] J. Alvarado-Valiente, J. Romero-Álvarez, E. Moguel, J. García-Alonso, and J. M. Murillo, "Technological diversity of quantum computing providers: a comparative study and a proposal for api gateway integration," *Software Quality Journal*, vol. 32, no. 1, pp. 53–73, 2024. [Online]. Available: https://doi.org/10.1007/s11219-023-09633-5

[28] S. Bahrani, R. D. Oliveira, J. M. Parra-Ullauri, R. Wang, and D. Simeonidou, "Resource management and circuit scheduling for distributed quantum computing interconnect networks," 2024. [Online]. Available: https://arxiv.org/abs/2409.12675

[29] H. Singh, "Optimised hybrid classical-quantum algorithm for accelerated solution of sparse linear systems," *arXiv preprint arXiv:2410.02408*, 2024.

[30] V. Padmasola, Z. Li, R. Chatterjee, and W. Dyk, "Solving the traveling salesman problem via different quantum computing architectures," 2025. [Online]. Available: https://arxiv.org/abs/2502.17725

[31] I.-D. Gheorghe-Pop, N. Tcholtchev, T. Ritter, and M. Hauswirth, "Quantum devops: Towards reliable and applicable nisq quantum computing," in *2020 IEEE Globecom Workshops (GC Wkshps*, 2020, pp. 1–6.

[32] T. Nakanishi, H. Furusho, K. Hisazumi, and A. Fukuda, "Dynamic spl and derivative development with uncertainty management for devops," in *2016 5th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)*, 2016, pp. 244–249.

[33] J. Alvarado-Valiente, J. Romero-Álvarez, J. Casco-Seco, E. Moguel, J. Garcia-Alonso, and J. M. Murillo, "Circuit scheduling policies on current qpus: Qcraft scheduler," in *Service-Oriented Computing*, W. Gaaloul, M. Sheng, Q. Yu, and S. Yangui, Eds. Singapore: Springer Nature Singapore, 2025, pp. 195–209.