Fundamental Patterns for Composing Quantum Algorithms

Lavinia Stiliadou[a], Johanna Barzen[b], Martin Beisel[c], Frank Leymann[d] and Benjamin Weder[e]
University of Stuttgart, Institute of Architecture of Application Systems, Universitätsstr. 38, 70569 Stuttgart, Germany

Keywords: Quantum Computing, Pattern Languages, Quantum Algorithms.

Abstract: Designing and implementing quantum algorithms is a time-consuming, complex, and error-prone task. As new quantum algorithms are often published as scientific papers without sufficient documentation and no available software implementations, algorithm designers and software developers have to figure out the missing details or redevelop parts of the quantum algorithm. Similar to classical programs, various quantum algorithms share the same subroutines as building blocks to realize the required functionality. Therefore, these building blocks have to be documented in a structured and easily understandable manner to foster their reuse and speed up development. Patterns are a well-established concept for documenting proven solutions to recurring problems and educating new developers. Hence, a pattern language capturing important concepts in the quantum computing domain was established. It already contains an initial set of patterns documenting common building blocks of quantum algorithms. In this paper, we extend the quantum computing pattern language by introducing five novel patterns, documenting fundamental building blocks to realize quantum algorithms.

1 INTRODUCTION

Quantum devices promise advancements in various domains, such as finance, optimization, and cryptography, by leveraging their ability to solve certain problems faster, with greater precision, or lower energy consumption compared to classical computers (Nielsen and Chuang, 2010; Preskill, 2018). Due to the high complexity of quantum computing, realizing quantum algorithms requires a thorough understanding of different topics, such as the encoding of data for the quantum devices or the subroutines used in quantum algorithms (Leymann and Barzen, 2020). Moreover, there is also a lack of documentation that provides quantum application developers with the knowledge to understand existing quantum algorithms, their building blocks, and corresponding implementations. Thus, the required building blocks for realizing quantum algorithms must be documented in a comprehensive manner (Gilliam et al., 2019).

An established way to capture proven solutions for recurring problems is patterns (Alexander et al., 1977). Patterns abstractly document solution strategies for problems occurring in a specific context. Thereby, patterns provide a technology and problem-independent description for solving problems that can be applied generally. Additionally, they explain why a problem is difficult to solve and what consequences might result from applying a pattern. To document fundamental concepts in the quantum computing domain, Leymann (2019) introduced a pattern language for quantum algorithms. Similar to other pattern languages, the quantum computing pattern language can be used to automatically generate software artifacts utilizing suitable pattern implementations (Vietz et al., 2025).

Since pattern languages continuously evolve, it is important to regularly update and expand the pattern language to reflect the latest advances. Therefore, in this paper, we extend the quantum computing pattern language with five new patterns that capture fundamental building blocks for quantum algorithms.

The remainder of this paper is as follows: In Section 2, fundamentals about quantum algorithms and the used pattern format are introduced. Section 3 documents the newly introduced patterns for building quantum algorithms and Section 4 discusses their usage and potential limitations. In Section 5 related work is presented, and Section 6 concludes the paper.

[a] https://orcid.org/0009-0001-0957-6108
[b] https://orcid.org/0000-0001-8397-7973
[c] https://orcid.org/0000-0003-2617-751X
[d] https://orcid.org/0000-0002-9123-259X
[e] https://orcid.org/0000-0002-6761-6243

Stiliadou, L., Barzen, J., Beisel, M., Leymann, F. and Weder, B.
Fundamental Patterns for Composing Quantum Algorithms.
DOI: 10.5220/0013555600004525
In *Proceedings of the 1st International Conference on Quantum Software (IQSOFT 2025)*, pages 61-69
ISBN: 978-989-758-761-0

2 FUNDAMENTALS

This section briefly introduces the fundamentals of gate-based quantum computing and discusses the pattern format and the applied pattern authoring process.

2.1 Gate-Based Quantum Computing

The currently predominant paradigm for realizing quantum devices is gate-based quantum computing (Nimbe et al., 2021; Pfaendler et al., 2024). To execute quantum algorithms on gate-based quantum devices, they need to be implemented as so-called quantum circuits (Nielsen and Chuang, 2010). These circuits define a sequence of operations, called gates, that manipulate the qubits and thereby modify the state of the quantum system. To extract classical information about a quantum system the state of its qubits must be measured (Nielsen and Chuang, 2010). Measuring a qubit causes its quantum state to collapse, making it an irreversible operation that prevents further computations using the measured qubit's state.

2.2 Pattern Format & Authoring Method

To facilitate the understanding of patterns within a pattern language, typically a uniform pattern format is used (Alexander et al., 1977; Fehling et al., 2014). In this paper, we adopt the pattern format previously used for documenting the quantum computing patterns, which is organized as follows:

Each pattern is identified by a unique *name* within the pattern language. Additionally, each pattern is associated with a mnemonic *icon* to aid in visual recognition. The problem addressed by the pattern is described in the *problem* statement, followed by an outline of the *context* in which the problem occurs. Afterward, the *forces* that complicate solving the problem are discussed. The *solution* section presents a proven strategy for solving the problem, accompanied by a corresponding *solution sketch* that visualizes the essence of the solution. In the *result* section, the consequences of applying the solution are discussed and potential next steps for handling them are suggested. To facilitate the navigation through the pattern language, each pattern is semantically linked to *related patterns*, e.g., to patterns that are commonly used in combination. Lastly, the *known uses* section discusses several real-world applications of the pattern.

In this paper, we focus on patterns related to building blocks for quantum algorithms. They were identified by examining implementations and analyzing the literature for building blocks that were utilized to implement different quantum algorithms. These building blocks were then filtered, documented, and iteratively refined to extract a set of novel patterns that can be used to understand and build quantum algorithms. While different methods for identifying and documenting patterns exist, we follow the approach presented by Fehling et al. (2014). Their approach comprises a pattern identification, authoring, and application phase, which emphasizes the incremental refinement of patterns.

3 PATTERNS FOR BUILDING QUANTUM ALGORITHMS

In this section, we provide an overview of the existing quantum computing pattern language and introduce five novel patterns documenting fundamental building blocks for realizing quantum algorithms.

3.1 Overview of the Quantum Computing Pattern Language

Figure 1 gives an overview of the quantum computing pattern language, showcasing both existing patterns as well as the novel patterns presented in this paper. Each pattern is assigned to one category depending on the related phase in the quantum software development lifecycle (Weder et al., 2022). In the following, the different pattern categories are presented:

First, the *unitary transformations* (Weigold et al., 2021a) patterns document subroutines that can be used as building blocks for realizing different quantum algorithms. The *measurement* patterns (Weigold et al., 2021a) describe techniques for extracting classical data from quantum states, ensuring that results from quantum computations can be interpreted and used effectively. Strategies for distributing computations across quantum and classical hardware are discussed by the *program flow* patterns (Weigold et al., 2021b). The *quantum machine learning* patterns (Stiliadou et al., 2025) provide insights into the development of quantum algorithms that leverage quantum mechanics to enhance machine learning tasks. To encode classical data into quantum circuits, the *data encodings* patterns (Weigold et al., 2021a) detail state preparation routines essential for embedding classical information into quantum states. Techniques to partition large quantum circuits into smaller ones, which can be executed with higher precision on current quantum devices, are documented in the *circuit cutting* patterns (Bechtold et al., 2023).

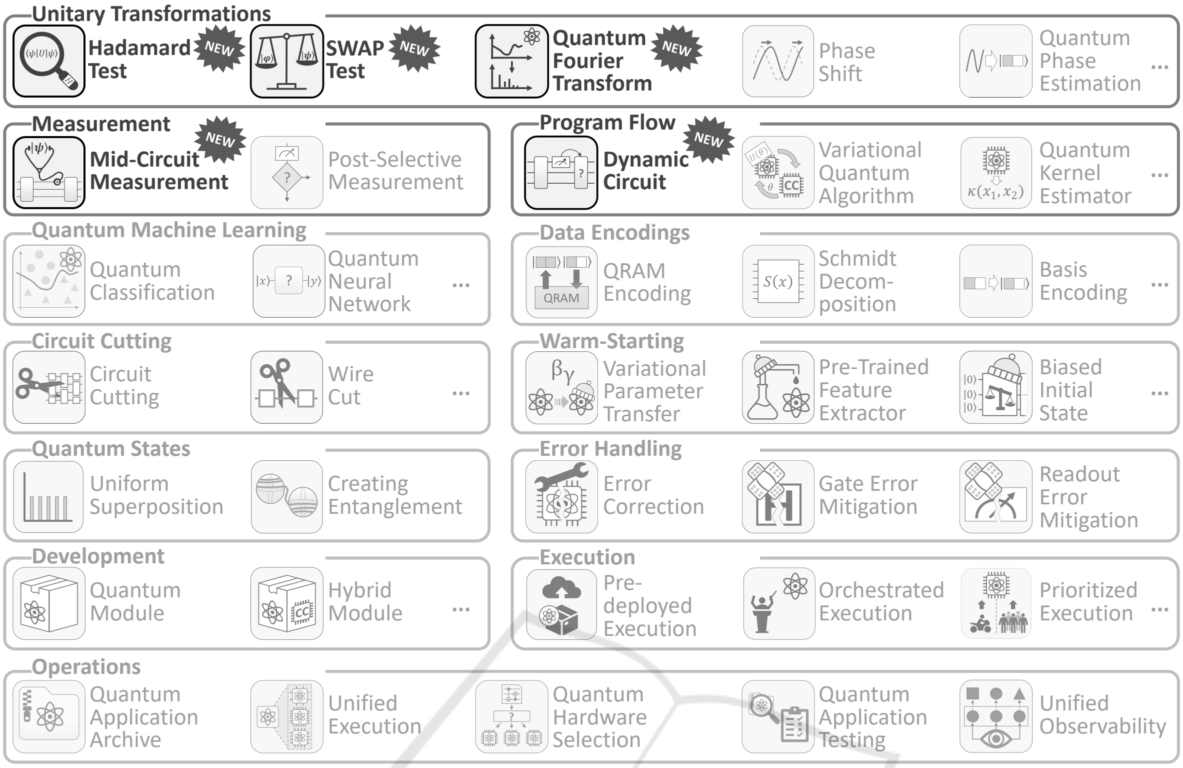


Figure 1: Overview of the quantum computing pattern language with some existing patterns (light gray) and the new patterns introduced in this paper (dark gray).

The *warm-starting* patterns (Truger et al., 2024) showcase techniques to improve the performance of quantum algorithms by initializing them with favorable starting points. To realize quantum algorithms, an understanding of *quantum states* (Leymann, 2019) is necessary. Approaches to mitigate noise in today's quantum devices are summarized in the *error handling* patterns (Beisel et al., 2022). This category describes how they can be created and potential application areas. The *development* patterns (Bühler et al., 2023) summarize best practices for developing hybrid quantum applications. An important aspect of developing such applications is understanding how quantum circuits are executed. These methods are documented in the *execution* patterns (Georg et al., 2023). Further, the *operations* patterns (Beisel et al., 2025b) document how to execute, monitor, and manage quantum applications.

In this work, we introduce five novel patterns describing fundamental building blocks for realizing various quantum algorithms: The HADAMARD TEST pattern focuses on estimating the real and imaginary parts of the expectation value of a unitary operator with respect to a quantum state. Further, the SWAP TEST pattern describes the computation of the similarity of two quantum states. The QUANTUM FOURIER TRANSFORM pattern addresses the challenge of efficiently transforming quantum states from the computational basis to the Fourier basis. The MID-CIRCUIT MEASUREMENT pattern documents an approach for extracting partial information from a quantum device while a circuit execution is still running. Finally, the DYNAMIC CIRCUIT pattern describes how quantum computations can be modified during runtime based on intermediate information about a part of the quantum state.

3.2 Hadamard Test

Problem: How to calculate the expectation value of a unitary operator for a given quantum state?

Context: Given a unitary operator U acting on n qubits, and let $|\psi\rangle$ be a n-qubit quantum state. Then, the expectation value $\langle\psi| U |\psi\rangle$ should be estimated.
Forces: Determining the expectation value classically is computationally expensive and scales exponentially with the number of qubits (Bravyi and Gosset, 2016). Quantum devices enable solving this problem more efficiently (Aharonov et al., 2006). However, using today's quantum devices also leads to additional chal-

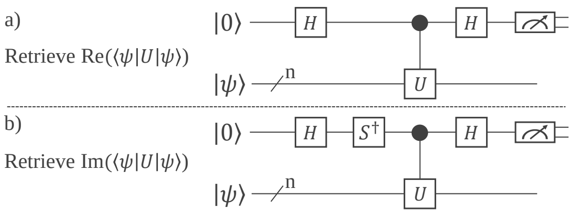


Figure 2: Solution sketch for the HADAMARD TEST pattern.

lenges. For example, large quantum circuits can not be executed successfully, and results are prone to errors (Preskill, 2018).
Solution: Figure 2 gives an overview of the structure of the quantum circuit, which is required to perform a Hadamard test. The quantum circuit requires one ancilla qubit on which a Hadamard gate is applied to bring it into an equal superposition. Subsequently, depending on whether the real (see a) or imaginary part (see b) of $\langle\psi| U |\psi\rangle$ should be retrieved, a $S^\dagger$ gate is added. Next, the unitary operator U is applied in a controlled manner by using the ancilla qubit as the control qubit. Due to a so-called *phase-kickback*, the information is transferred from the target register to the control qubit when entangling the qubits with the controlled U gate (Ossorio-Castillo et al., 2023). Finally, another Hadamard gate is applied to the ancilla qubit enabling the retrieval of the expectation value $\langle\psi| U |\psi\rangle$ by measuring the ancilla qubit.
Result: The real or imaginary part of the expectation value $\langle\psi| U |\psi\rangle$ is the output after measuring the ancilla qubit. Depending on the required upper bound of the absolute error, the number of samples must be increased.
Related Patterns: The HADAMARD TEST can be implemented using the QUANTUM MODULE pattern (Bühler et al., 2023), i.e., separating the inputs from the code generation logic required to create the quantum circuit realizing the Hadamard test. For separable input quantum states $|\psi\rangle$, the SWAP TEST pattern can be emulated by (i) separating the input state into two separate quantum states and by (ii) utilizing the SWAP gate as the unitary operator U within the HADAMARD TEST.
Known Uses: The Hadamard test is utilized within the Aharonov–Jones–Landau algorithm to compute the Jones polynomial (Aharonov et al., 2006). Furthermore, the Hadamard test can also be used for variational quantum algorithms to determine the gradient of the objective function (Harrow and Napp, 2021). Arad and Landau (2010) utilize the Hadamard test in the context of tensor networks. Xu et al. (2022) employ the Hadamard test as the measuring method for a variational quantum support vector machine.

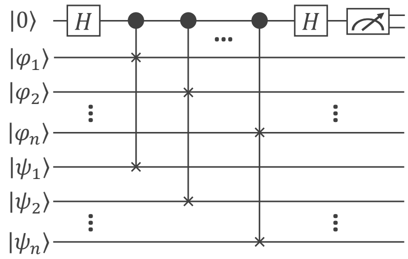


Figure 3: Solution sketch for the SWAP TEST pattern.

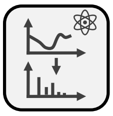
3.3 SWAP Test

Problem: How to evaluate how similar two given quantum states are to each other?

Context: Given two n-qubit quantum states $|\varphi\rangle$ and $|\psi\rangle$. Then, the similarity between these states should be calculated.
Forces: The similarity of two quantum states may influence the processing of a quantum algorithm. Performing classical measurements is unsuitable for comparing quantum states as the states are destroyed and can not be used for further computations.
Solution: Perform the SWAP test to determine the similarity of the two given quantum states $|\varphi\rangle$ and $|\psi\rangle$. The structure of the quantum circuit, which is required to perform a SWAP test, is depicted in Figure 3. It requires one ancilla qubit on which a Hadamard gate is applied. Next, a sequence of controlled SWAP operators is applied to each qubit of the two states using the ancilla qubit as the control qubit. For example, the first controlled SWAP operation is performed between $|\varphi_1\rangle$ and $|\psi_1\rangle$. Finally, another Hadamard gate is applied to the ancilla qubit, leading to the state: $\frac{1}{2}|0\rangle(|\phi\rangle|\psi\rangle + |\psi\rangle|\phi\rangle) + \frac{1}{2}|1\rangle(|\phi\rangle|\psi\rangle - |\psi\rangle|\phi\rangle)$.
Result: After the measurement of the ancilla qubit, the outcome determines the similarity between the states $|\phi\rangle$ and $|\psi\rangle$. If the states are identical, the measurement of the ancilla bit results in 0 with probability 1. In contrast, if the states are orthogonal, the measurement results in 0 or 1 with an equal probability of 0.5.
Related Patterns: The QUANTUM CLASSIFICATION pattern (Stiliadou et al., 2025) can utilize the SWAP TEST to estimate the distances between two points. The SWAP TEST can be realized as a QUANTUM MODULE, and its functionality can be provided via a CLASSICAL-QUANTUM INTERFACE to ease its integration with additional classical functionality (Bühler et al., 2023).
Known Uses: The SWAP test was initially introduced by Buhrman et al. (2001). Gitiaux et al. (2022) show

how to generalize the SWAP test for an arbitrary number m of states to compare using $O(\log(m))$ ancilla qubits. Foulds et al. (2021) adapt the SWAP test to enable checking the presence of entanglement and show how it can be used to distinguish different entanglement classes. Zhao et al. (2019) discuss how quantum neural networks can be built using the SWAP test.

3.4 Quantum Fourier Transform (QFT)



Problem: How to extract frequencies from a function using a quantum device?

Context: Frequencies need to be extracted from function values, which are given at N distinct points to identify characteristics such as periodicity and distribution of the frequencies.
Forces: The best known classical implementation of a Fourier transform, the so-called *Fast Fourier Transform (FFT)* is computationally expensive as it requires $O(N \log N)$ operations for a vector that contains N data points of a function (Camps et al., 2020). While quantum computing enables solving this problem more efficiently, the number of operations executable in sequence on current quantum devices is limited due to high error rates and short decoherence times (Preskill, 2018).
Solution: The QFT extracts frequencies from a n-qubit quantum register $|x\rangle = |x_{n-1},\ldots,x_0\rangle$, where $n = \log N$. x is interpreted as a decimal number and $x_j \in \{0,1\}$ are the binary digits of x. In the QFT, each qubit $|x_j\rangle$ is transformed as follows:

$$|x_j\rangle \mapsto \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i x/2^j}\,|1\rangle\right).$$

The required stages are illustrated in Figure 4. QFT involves n stages, one for each qubit in the quantum register. At stage S_j, operations are performed solely on qubit x_{n-j}, which involves a Hadamard gate, followed by controlled $R_2,\ldots,R_{n+1-j}$ gates, except in stage S_n, where only the Hadamard gate is applied. The R_k gate is controlled by qubit $x_{n+1-j-k}$ to realize the phase $e^{2\pi i/2^k}$. The gate R_k has the form:

$$R_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{pmatrix}, \text{ for } 2 \le k \le n$$

This circuit requires $O(n^2) = O((\log N)^2)$ operations, as each stage S_j involves a single Hadamard operation and $n-1+j$ controlled rotations.
Result: The QFT provides an exponential speedup compared to the FFT. After applying the QFT, the quantum state is transformed into the Fourier basis. If the QFT is applied to a quantum state encoding a periodic function with period p, the resulting state has a high probability of measuring y, where y is a multiple of N/p (Barzen and Leymann, 2022; Dixit and Jian, 2022). Due to the controlled phase gates, QFT requires a high connectivity between the qubits.
Related Patterns: The PHASE SHIFT pattern (Leymann, 2019) is used to implement the modification of the phase. To reduce the depth of the quantum circuit implementing the QFT, the DYNAMIC CIRCUIT pattern can be utilized (Bäumer et al., 2024a). The QUANTUM PHASE ESTIMATION pattern (Weigold et al., 2021a) uses QFT to estimate the eigenvalues of a unitary operator.
Known Uses: Shor's Algorithm extracts the period of a modular exponentiation function using QFT (Shor, 1994). Dixit and Jian (2022) use QFT to estimate frequencies from driving cycles of vehicles, i.e., graphs visualizing the speed of the vehicle over time (Dixit and Jian, 2022). Roncallo et al. (2023) use QFT to compress digital images.

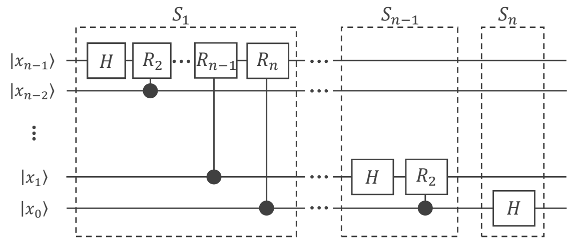
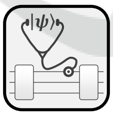


Figure 4: Solution sketch for the QUANTUM FOURIER TRANSFORM pattern.

3.5 Mid-Circuit Measurement



Problem: How to extract partial information from a quantum device while a circuit execution is still running?

Context: During the execution of a quantum circuit, intermediate results, i.e., information about the state of one or multiple qubits, should be extracted before the circuit execution finally terminates.
Forces: As quantum devices only provide a limited number of qubits, these qubits must be utilized efficiently, e.g., by reusing ancilla qubits that are no longer required. When measuring a qubit that is entangled with other qubits, the measurement does not only collapse the state of the measured qubit but also irreversibly impacts the quantum states of the entangled qubits.
Solution: Incorporate so-called mid-circuit measurements into the quantum circuit, i.e., measurements before the quantum circuit execution terminates. As il-

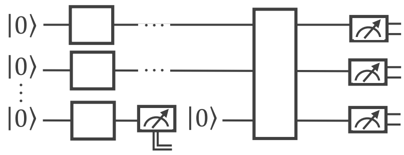


Figure 5: Overview of the solution sketch for the MID-CIRCUIT MEASUREMENT pattern.

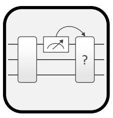
lustrated in Figure 5 various quantum operations are performed on different qubits initialized in the $|0\rangle$ state, resulting in an intermediate quantum state. A subset of m qubits is then measured, yielding a measurement outcome $s \in \{0,1\}^{\otimes m}$, extracting classical information about a part of this intermediate quantum state. Afterwards, the execution of the quantum circuit continues. For example, in Figure 5 a mid-circuit measurement is performed on the last qubit, measuring a classical 0 and collapsing the state of the qubit into the $|0\rangle$ state.

Result: Mid-circuit measurements provide information about the intermediate states of the measured qubits. Each measured qubit collapses into the state $|0\rangle$ or $|1\rangle$, which can, e.g., be used to reset them by applying a controlled X operation after the measurement (Xu et al., 2023). After resetting the qubits, they can be reused for further computations. By performing mid-circuit measurements on a qubit, the states of qubits that are entangled with the measured qubit are influenced. The states of qubits that are not entangled with the measured qubits are preserved.

Related Patterns: The DYNAMIC CIRCUIT pattern uses the MID-CIRCUIT MEASUREMENT pattern to retrieve intermediate information for dynamically modifying quantum computations during runtime.

Known Uses: Mid-circuit measurements can be used to reset a qubit conditionally (Koh et al., 2024). Govia et al. (2023) analyze how mid-circuit measurements affect the states of topologically connected qubits. Smith et al. (2024) present a constant-depth state preparation of matrix product states using mid-circuit measurements and feedforward operations.

3.6 Dynamic Circuit



Problem: How to modify a quantum computation during runtime based on intermediate information about a part of the quantum state?

Context: A quantum circuit must be modified based on intermediate information about a part of the quantum state.

Forces: Measuring a qubit causes its state to collapse and breaks the entanglement between the measured qubit and the entangled qubits. Operations between qubits that are far apart from each other require a lot of intermediate SWAP operations, increasing the depth of the circuit. The low decoherence times of current quantum devices limit the maximum execution time of quantum circuits.

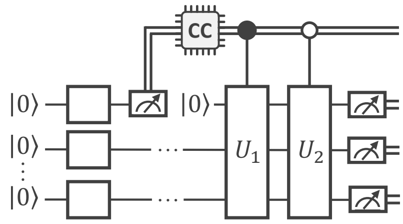


Figure 6: Overview of the solution sketch for the DYNAMIC CIRCUIT pattern.

Solution: To modify a quantum computation during runtime based on intermediate information about a part of the quantum state, define a dynamic quantum circuit that utilizes mid-circuit measurements as well as classical processing. Classical processing can either be a feedforward of the measurement results or a more complex computation that utilizes the measurement results to adaptively apply or skip specific quantum operations. Figure 6 exemplarily showcases a dynamic circuit using feedforward: First, the quantum circuit performs a sequence of operations. Then, a mid-circuit measurement is performed on the first qubit and the measurement result is used for classical processing. Based on the outcome of the classical processing, it is determined if the gate U_1 or U_2 shall be applied. If the outcome of the classical processing is 1, then U_1 is applied; if it is 0, then U_2 is applied.

Result: Dynamic circuits enable the development of algorithms and optimization routines that require intermediate information about a part of the quantum state. The classical processing performed after the mid-circuit measurement must be faster than the decoherence time of the quantum device so that the quantum state is not lost before the quantum computation can be modified and completed. Additionally, feedforward of measurement results leads to constant latency for executing conditional operations, while more complex real-time computations introduce additional variable delays depending on their complexity (Gupta et al., 2024).

Related Patterns: The MID-CIRCUIT MEASUREMENT pattern is used to extract intermediate information about the quantum states during runtime. Dynamic circuits can be utilized with the INITIALIZATION pattern (Leymann, 2019) to reduce the depth of the quantum circuits. Combining the CIRCUIT CUTTING pattern (Bechtold et al., 2023) and the DYNAMIC CIRCUIT pattern can reduce the number of cuts required to split a quantum circuit (Pawar et al., 2023). Further, dynamic circuits can be used as ansatzes for the VARIATIONAL QUANTUM ALGORITHM pattern (Weigold et al., 2021b) as they are free of barren plateaus (Deshpande et al., 2024).

Known Uses: Dynamic circuits can be incorporated into error correction techniques to reduce errors accumulated during the quantum computation (Niu et al., 2024). Another application area is to reduce the depth of QFT circuits by utilizing dynamic circuits (Bäumer et al., 2024a). Distant qubits can be entangled using dynamic circuits, drastically reducing the number of required SWAP operations and, therefore, the depth of the quantum circuit (Bäumer et al., 2024b).

4 DISCUSSION

Patterns are commonly applied to solve real-world problems. Due to the interdisciplinary nature of quantum computing, establishing common knowledge between the different stakeholders is especially important. As stated above, transferring this knowledge into executable quantum applications is a complex, time-consuming, and error-prone task. To facilitate the development of quantum applications, various approaches to automate this process have been introduced: Vietz et al. (2025) present an approach to automate the identification of suitable patterns for a given problem description. These patterns can be used to generate a quantum application to solve this problem. Thus, in this work, we extend the set of usable patterns, enhancing the capabilities of the framework. Beisel et al. (2025a) uses the quantum computing patterns to automatically generate quantum workflows.

While mid-circuit measurements can provide different benefits, such as reducing circuit depth, they are currently not supported by all quantum devices. This limits the general applicability of the MID-CIRCUIT MEASUREMENT pattern. Similar limitations apply to the DYNAMIC CIRCUIT pattern, as it relies on the availability of mid-circuit measurements and real-time classical processing. While dynamic circuits are a promising concept, they are currently only supported by a small number of quantum programming frameworks, such as Qiskit or Pennylane.

5 RELATED WORK

This work introduces patterns for building quantum algorithms that extend the existing quantum computing pattern language as discussed in Section 3.1. Various works have explored different aspects of quantum computing, but they do not follow the pattern format established by Alexander et al. (1977): Baczyk et al. (2024) document patterns that assist in architectural decision-making for quantum applications. Similarly, Khan et al. (2023) identify a range of architecture design patterns for quantum applications through a systematic literature survey. A variety of patterns for building quantum circuits were introduced: Gilliam et al. (2019) present a dictionary for building quantum algorithms, which aims to avoid linear algebra and quantum mechanics. While they include QFT in their dictionary, they do not describe it as a pattern and only mention it as an alternative to classical Fourier transform. Huang and Martonosi (2019) apply quantum programming patterns to find bugs in quantum circuits. Additionally, Guo et al. (2024) present a set of patterns tailored for defining ansätze in variational quantum algorithms. However, they do not include mid-circuit measurements, which are necessary to realize dynamic circuits that have been proven to be barren plateau-free (Deshpande et al., 2024). To evaluate the real-world adoption of the quantum computing pattern language, Pérez-Castillo et al. (2024) investigate quantum software repositories, searching for instances of pattern usage. However, their analysis is limited to only five patterns.

To reduce the manual effort needed to implement the abstractly documented solutions described by patterns, Falkenthal et al. (2017) introduced the concept of so-called concrete solutions. Concrete solutions implement the solution strategy provided by patterns for a specific use case utilizing a certain technology. These concrete solutions can be employed as reusable building blocks for realizing applications.

6 CONCLUSION AND OUTLOOK

Quantum algorithms are often composed utilizing reusable building blocks. However, these building blocks typically lack suitable documentation and proper implementations that can be reused. To overcome these issues, in this paper, we document five novel patterns summarizing fundamental building blocks for quantum algorithms. The patterns are incorporated into the quantum computing pattern language, which is publicly available in the Pattern Atlas (Kipu Quantum, 2025), a tool for authoring and sharing patterns from different pattern languages (Leymann and Barzen, 2021).

New quantum algorithms and corresponding software tools are developed by researchers in industry and academia. Thus, in future work, we plan to analyze new developments to identify best practices that can be documented as patterns and included in the quantum computing pattern language.

ACKNOWLEDGEMENTS

This work was partially funded by the BMWK projects *EniQmA* (01MQ22007B) and *SeQuenC* (01MQ22009B).

REFERENCES

Aharonov, D., Jones, V., and Landau, Z. (2006). A Polynomial Quantum Algorithm for Approximating the Jones Polynomial. In *Proceedings of the 38th annual ACM symposium on Theory of Computing*, pages 427–436.

Alexander, C., Ishikawa, S., and Silverstein, M. (1977). *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press.

Arad, I. and Landau, Z. (2010). Quantum computation and the evaluation of tensor networks. *SIAM Journal on Computing*, 39(7):3089–3121.

Baczyk, M., Pérez-Castillo, R., and Piattini, M. (2024). Towards a Framework of Architectural Patterns for Quantum Software Engineering. In *Proceedings of the 4th International Workshop on Quantum Software Engineering and Technology (Q-SET)*, volume 2, pages 228–233.

Barzen, J. and Leymann, F. (2022). Continued Fractions and Probability Estimations in Shor's Algorithm: A Detailed and Self-Contained Treatise. *AppliedMath*, 2(3):393–432.

Bäumer, E., Tripathi, V., Seif, A., Lidar, D., and Wang, D. S. (2024a). Quantum Fourier transform using dynamic circuits. *Physical review letters*, 133(15):150602.

Bäumer, E., Tripathi, V., Wang, D. S., Rall, P., Chen, E. H., Majumder, S., Seif, A., and Minev, Z. K. (2024b). Efficient long-range entanglement using dynamic circuits. *PRX Quantum*, 5(3).

Bechtold, M., Barzen, J., Beisel, M., Leymann, F., and Weder, B. (2023). Patterns for Quantum Circuit Cutting. In *Proceedings of the 30th Conference on Pattern Languages of Programs (PLoP)*. HILLSIDE.

Beisel, M., Barzen, J., Leymann, F., Stiliadou, L., Vietz, D., and Weder, B. (2025a). Pattern-based Generation and Adaptation of Quantum Workflows. In *Proceedings of the 47th International Conference on Software Engineering (ICSE)*. Springer.

Beisel, M., Barzen, J., Leymann, F., Truger, F., Weder, B., and Yussupov, V. (2022). Patterns for Quantum Error Handling. In *Proceedings of the 14th International Conference on Pervasive Patterns and Applications (PATTERNS)*, pages 22–30. Xpert Publishing Services (XPS).

Beisel, M., Barzen, J., Leymann, F., and Weder, B. (2025b). Operations Patterns for Hybrid Quantum Applications. In *Proceedings of the 15th International Conference on Cloud Computing and Services Science (CLOSER)*. SciTePress.

Bravyi, S. and Gosset, D. (2016). Improved Classical Simulation of Quantum Circuits Dominated by Clifford Gates. *Physical review letters*, 116(25):250501.

Bühler, F., Barzen, J., Beisel, M., Georg, D., Leymann, F., and Wild, K. (2023). Patterns for Quantum Software Development. In *Proceedings of the 15th International Conference on Pervasive Patterns and Applications (PATTERNS)*, pages 30–39. Xpert Publishing Services (XPS).

Buhrman, H., Cleve, R., Watrous, J., and De Wolf, R. (2001). Quantum Fingerprinting. *Physical review letters*, 87(16):167902.

Camps, D., Van Beeumen, R., and Yang, C. (2020). Quantum Fourier transform revisited. *Numerical Linear Algebra with Applications*, 28(1).

Deshpande, A., Hinsche, M., Najafi, S., Sharma, K., Sweke, R., and Zoufal, C. (2024). Dynamic parameterized quantum circuits: expressive and barren-plateau free. *arXiv:2411.05760*.

Dixit, V. and Jian, S. (2022). Quantum Fourier transform to estimate drive cycles. *Scientific Reports*, 12(1):654.

Falkenthal, M., Barzen, J., Breitenbücher, U., and Leymann, F. (2017). Solution Languages: Easing Pattern Composition in Different Domains. *International Journal on Advances in Software*, 10(3):263–274.

Fehling, C., Barzen, J., Breitenbücher, U., and Leymann, F. (2014). A Process for Pattern Identification, Authoring, and Application. In *Proceedings of the 19th European Conference on Pattern Languages of Programs (EuroPLoP)*. ACM.

Foulds, S., Kendon, V., and Spiller, T. (2021). The controlled SWAP test for determining quantum entanglement. *Quantum Science and Technology*, 6(3):035002.

Georg, D., Barzen, J., Beisel, M., Leymann, F., Obst, J., Vietz, D., Weder, B., and Yussupov, V. (2023). Execution Patterns for Quantum Applications. In *Proceedings of the 18th International Conference on Software Technologies (ICSOFT)*, pages 258–268. SciTePress.

Gilliam, A., Venci, C., Muralidharan, S., Dorum, V., May, E., Narasimhan, R., and Gonciulea, C. (2019). Foundational Patterns for Efficient Quantum Computing. *arXiv:1907.11513*.

Gitiaux, X., Morris, I., Emelianenko, M., and Tian, M. (2022). Swap test for an arbitrary number of quantum states. *Quantum Information Processing*, 21(10):344.

Govia, L. C. G., Jurcevic, P., Wood, C. J., Kanazawa, N., Merkel, S. T., and McKay, D. C. (2023). A randomized benchmarking suite for mid-circuit measurements. *New Journal of Physics*, 25(12):123016.

Guo, X., Muta, T., and Zhao, J. (2024). Quantum Circuit Ansatz: Patterns of Abstraction and Reuse of Quantum Algorithm Design. *arXiv:2405.05021*.

Gupta, R. S., Sundaresan, N., Alexander, T., Wood, C. J., Merkel, S. T., Healy, M. B., Hillenbrand, M., Jochym-O'Connor, T., Wootton, J. R., Yoder, T. J., Cross, A. W., Takita, M., and Brown, B. J. (2024). Encoding a magic state with beyond break-even fidelity. *Nature*, 625(7994):259–263.

Harrow, A. W. and Napp, J. C. (2021). Low-Depth Gradient Measurements Can Improve Convergence in Variational Hybrid Quantum-Classical Algorithms. *Physical review letters*, 126(14):140502.

Huang, Y. and Martonosi, M. (2019). Statistical Assertions for Validating Patterns and Finding Bugs in Quantum Programs. In *Proceedings of the 46th International Symposium on Computer Architecture (ISCA)*, pages 541–553. ACM.

Khan, A. A., Ahmad, A., Waseem, M., Liang, P., Fahmideh, M., Mikkonen, T., and Abrahamsson, P. (2023). Software architecture for quantum computing systems — A systematic review. *Journal of Systems and Software*, 201:111682.

Kipu Quantum (2025). PlanQK Pattern Atlas. https://patterns.platform.planqk.de/pattern-languages/af7780d5-1f97-4536-8da7-4194b093ab1d.

Koh, J. M., Koh, D. E., and Thompson, J. (2024). Readout Error Mitigation for Mid-Circuit Measurements and Feedforward. *arXiv:2406.07611*.

Leymann, F. (2019). Towards a Pattern Language for Quantum Algorithms. In *Proceedings of the 1st International Workshop on Quantum Technology and Optimization Problems (QTOP)*. Springer.

Leymann, F. and Barzen, J. (2020). The bitter truth about gate-based quantum algorithms in the NISQ era. *Quantum Science and Technology*, 5(4):1–28.

Leymann, F. and Barzen, J. (2021). Pattern Atlas. In *Next-Gen Digital Services. A Retrospective and Roadmap for Service Computing of the Future*, pages 67–76. Springer.

Nielsen, M. A. and Chuang, I. L. (2010). *Quantum Computation and Quantum Information*. Cambridge University Press.

Nimbe, P., Weyori, B. A., and Adekoya, A. F. (2021). Models in quantum computing: a systematic review. *Quantum Information Processing*, 20(2):80.

Niu, S., Kokcu, E., Mitra, A., Szasz, A., Hashim, A., Kalloor, J., de Jong, W. A., Iancu, C., and Younis, E. (2024). AC/DC: Automated Compilation for Dynamic Circuits. *arXiv:2412.07969*.

Ossorio-Castillo, J., Pastor-Díaz, U., and Tornero, J. M. (2023). A generalisation of the Phase Kick-Back. *Quantum Information Processing*, 22(3):143.

Pawar, A., Li, Y., Mo, Z., Guo, Y., Zhang, Y., Tang, X., and Yang, J. (2023). Integrated Qubit Reuse and Circuit Cutting for Large Quantum Circuit Evaluation. *arXiv:2312.10298*.

Pérez-Castillo, R., Fernández-Osuna, M., Cruz-Lemus, J. A., and Piattini, M. (2024). A preliminary study of the usage of design patterns in quantum software. In *Proceedings of the 5th ACM/IEEE International Workshop on Quantum Software Engineering (QSE)*, pages 41–48.

Pfaendler, S. M.-L., Konson, K., and Greinert, F. (2024). Advancements in Quantum Computing—Viewpoint: Building Adoption and Competency in Industry. *Datenbank-Spektrum*, 24(1):5–20.

Preskill, J. (2018). Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79.

Roncallo, S., Maccone, L., and Macchiavello, C. (2023). Quantum JPEG. *AVS Quantum Science*, 5(4):043803.

Shor, P. W. (1994). Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134. IEEE.

Smith, K. C., Khan, A., Clark, B. K., Girvin, S., and Wei, T.-C. (2024). Constant-depth preparation of matrix product states with adaptive quantum circuits. *PRX Quantum*, 5:030344.

Stiliadou, L., Barzen, J., Beisel, M., Leymann, F., and Weder, B. (2025). Patterns for Quantum Machine Learning. In *Proceedings of the 17th International Conference on Pervasive Patterns and Applications (PATTERNS)*, pages 7–14. Xpert Publishing Services (XPS).

Truger, F., Barzen, J., Beisel, M., Leymann, F., and Yussupov, V. (2024). Warm-Starting Patterns for Quantum Algorithms. In *Proceedings of the 16th International Conference on Pervasive Patterns and Applications (PATTERNS)*, pages 25–31. Xpert Publishing Services (XPS).

Vietz, D., Barzen, J., Beisel, M., Leymann, F., Stiliadou, L., and Weder, B. (2025). Harnessing Patterns to Support the Development of Hybrid Quantum Applications. In *Proceedings of 14th International Conference on Software and Computer Applications (ICSCA '25)*. ACM.

Weder, B., Barzen, J., Leymann, F., and Vietz, D. (2022). Quantum Software Development Lifecycle. In *Quantum Software Engineering*, pages 61–83. Springer.

Weigold, M., Barzen, J., Leymann, F., and Salm, M. (2021a). Encoding patterns for quantum algorithms. *IET Quantum Communication*, 2(4):141–152.

Weigold, M., Barzen, J., Leymann, F., and Vietz, D. (2021b). Patterns for Hybrid Quantum Algorithms. In *Proceedings of the 15th Symposium and Summer School on Service-Oriented Computing (SummerSOC)*, pages 34–51. Springer.

Xu, C., Chen, J., Mi, A., and Szefer, J. (2023). Securing NISQ Quantum Computer Reset Operations Against Higher Energy State Attacks. In *Proceedings of the 30th Conference on Computer and Communications Security (CCS)*, page 594–607. ACM.

Xu, L., Zhang, X.-Y., Liang, J.-M., Wang, J., Li, M., Jian, L., and Shen, S.-q. (2022). Variational quantum support vector machine based on Hadamard test. *Communications in Theoretical Physics*, 74(5):055106.

Zhao, J., Zhang, Y.-H., Shao, C.-P., Wu, Y.-C., Guo, G.-C., and Guo, G.-P. (2019). Building quantum neural networks based on a swap test. *Physical Review A*, 100(1):012334.