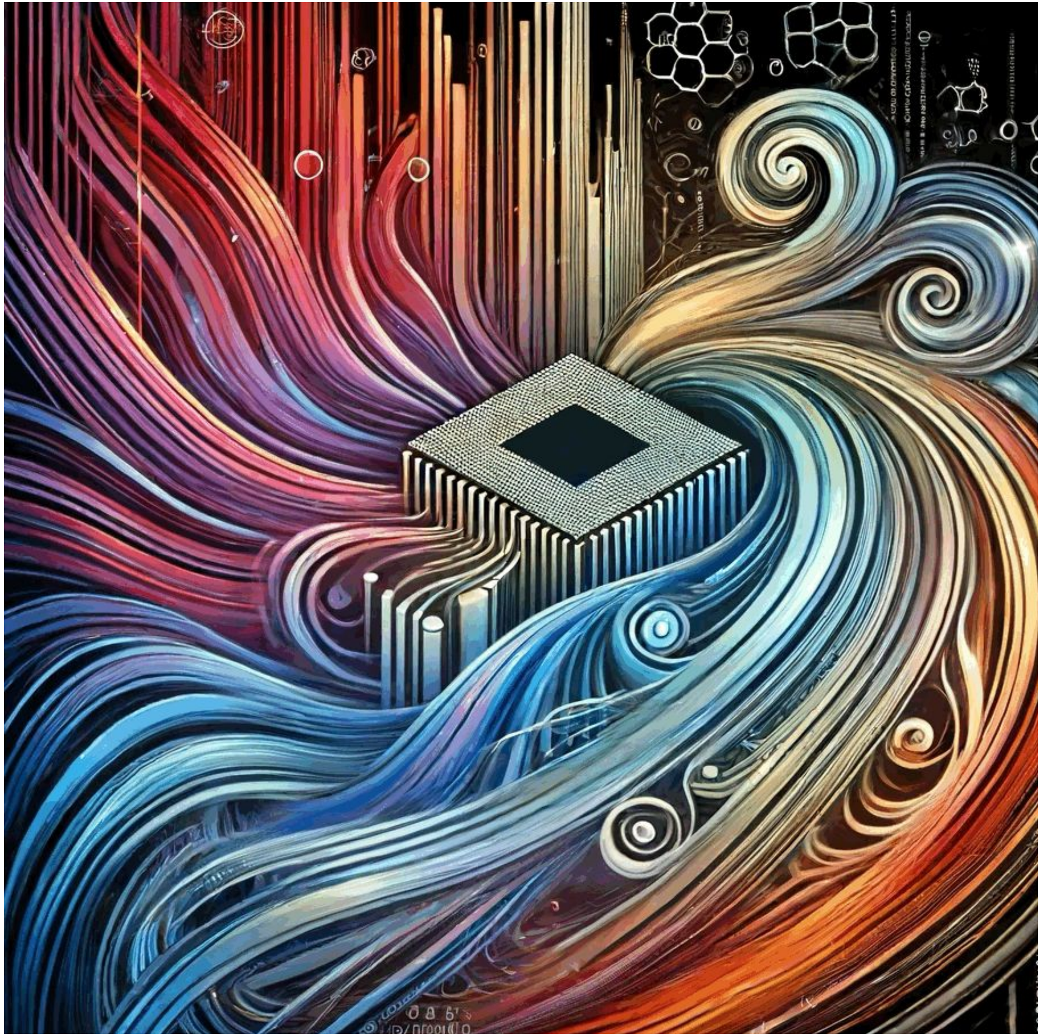
Patterns for Quantum Software Engineering

Michal Baczyk, Ricardo Pérez-Castillo, Mario Piattini

ReAQCT 2024, Budapest, 19.07.2024

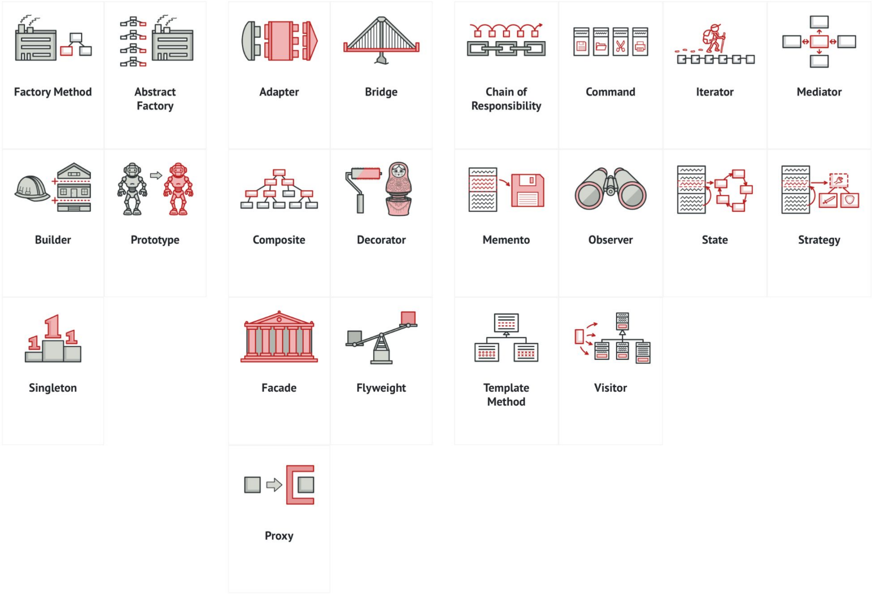


What is a pattern?

Why are these useful or even necessary in software engineering?

“Patterns are often confused with algorithms, because both concepts describe typical solutions to some known problems.

While an algorithm always defines a clear set of actions that can achieve some goal, a pattern is a more high-level description of a solution. The code of the same pattern applied to two different programs may be different.”



References: https://refactoring.guru/design-patterns/what-is-pattern

TABLE I

TABLE PRESENTS THE PROPOSED ORTHOGONAL TAXONOMY FOR CATEGORIZING STRUCTURES: PATTERNS AND ALGORITHMS USED ACROSS CLASSICAL, QUANTUM AND HYBRID SOFTWARE SYSTEMS DEVELOPMENTS. WHILE THE ALGORITHMS LAYER CANNOT BE CLASSIFIED AS PATTERNS AND, BECAUSE OF ITS DIRECT RELEVANCE TO THE DEVELOPMENT OF QUANTUM COMPUTING IT IS INCLUDED TO COMPLETE THE PICTURE.

	Classical Software Systems	**Quantum Software Systems**	**Hybrid Software Systems**
Design patterns	e.g. Singleton Pattern, Observer Pattern, Factory Method Pattern [14]–[17]	Quantum Circuits design patterns [34]–[37]	N/A
Algorithms families	N/A	Deutsch Jozsa, Grover, QFT, HHL [25]–[28]	VQA, QAOA [30]–[32]
Architectural patterns	e.g. Model View Controller (MVC), Microservices Architecture, Layered (n-tier) Architecture [18]	N/A	Orchestration of hybrid workflows, cloud computing with quantum resources [39], [40], [40], [41]

Framework for Quantum Software Patterns

- **Orchestration and execution of the quantum counterpart in a hybrid system**, which is one of the primary challenges in QSE. Unlike classical computing, where the execution model is well-established and standardized, quantum computing introduces new complexities due to the inherent probabilistic nature of quantum operations and the limited availability of quantum resources [42].

- **Maintainability** is another critical aspect of quantum software systems that can benefit from architectural patterns. As quantum computing technology evolves rapidly, software architectures must be designed to accommodate changes and updates in quantum hardware, algorithms, and programming models. Architectural patterns that promote modularity, loose coupling, and separation of concerns can significantly enhance the maintainability of quantum software systems, allowing for easier updates and modifications as the technology matures [45].

- **Efficiency** is a paramount concern in quantum computing, given the current limitations of quantum hardware and the high cost of quantum operations. Architectural patterns can help optimize the utilization of quantum resources by minimizing the number of required qubits and quantum gates, reducing the depth of quantum circuits, and optimizing the classical-quantum interaction.

Goals for Architectural Quantum Software Patterns

Advice for practitioners

To help practitioners navigate this emerging field, we highlight a few key areas where innovation in software patterns can make a significant impact:

- Quantum-Classical Integration: Practitioners should focus on design patterns that facilitate seamless integration between quantum and classical components, which is especially important in the context of Quantum Machine Leaning which depends on hybrid resources allocation.
- Quantum Algorithm Composition: Design patterns that promote the modular composition of quantum algorithms can help practitioners build more complex and powerful quantum applications.
- Quantum Performance Optimization: Optimizing the performance of quantum software is essential for achieving the desired computational advantages which may have far-reaching consequences for example for cybersecurity where Y2Q event is a topic on all national security agendas worldwide.

Thank you for your attention!

Questions?