Towards Quantum Software Engineering

University of Stuttgart

Benjamin Weder

benjamin.weder@iaas.uni-stuttgart.de

Institute of Architecture of Application Systems

SimTech
Cluster of Excellence

PlanQK

SEQUOIA

Motivation

- Recent advances with more powerful quantum computers
  ➔ New quantum applications are needed

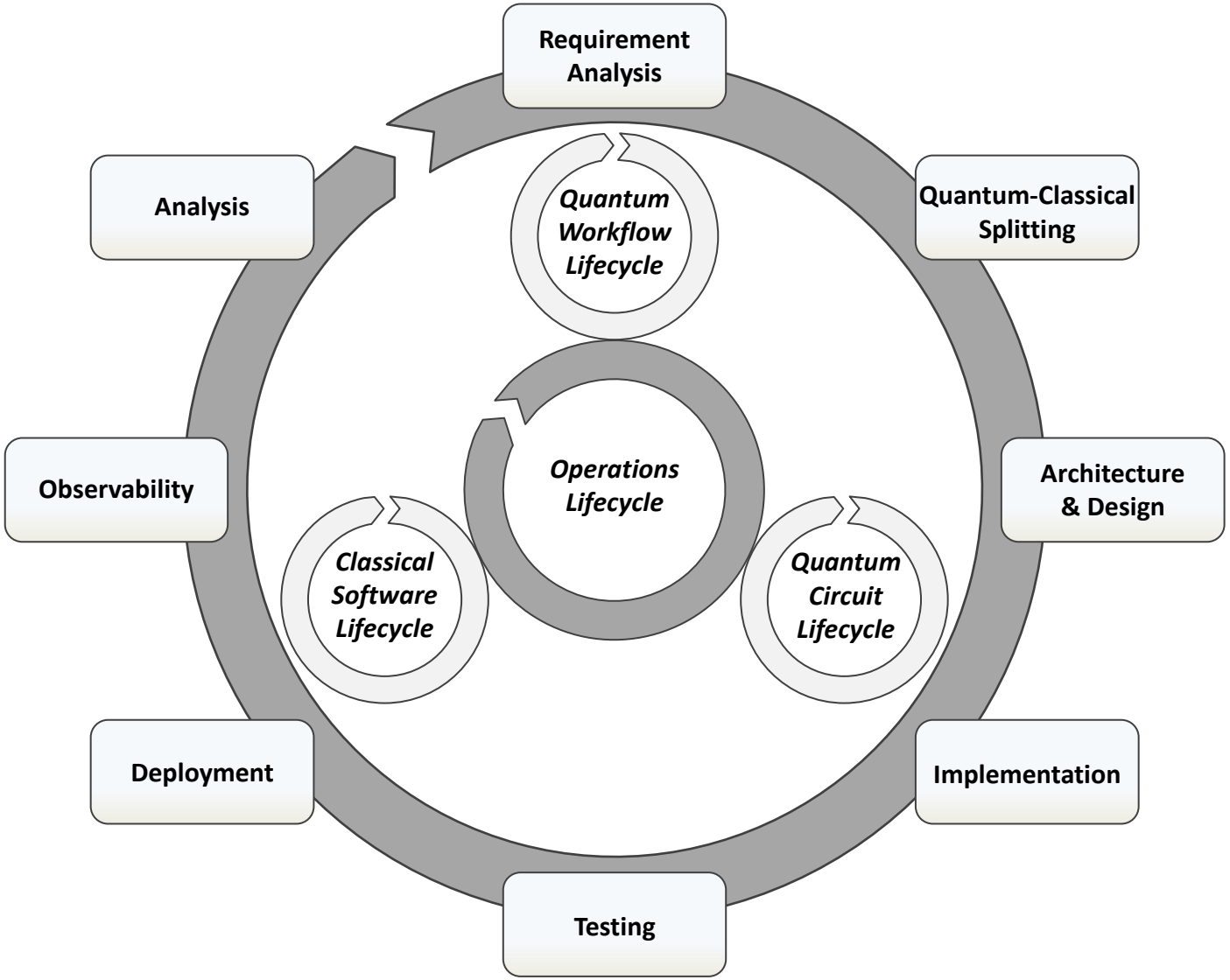
- Development of quantum applications requires expertise from different fields:
  - Computer science
  - Physics
  - Mathematics
  - …

- Common understanding of the development and execution process needed

➔ *Quantum Software Lifecycle*

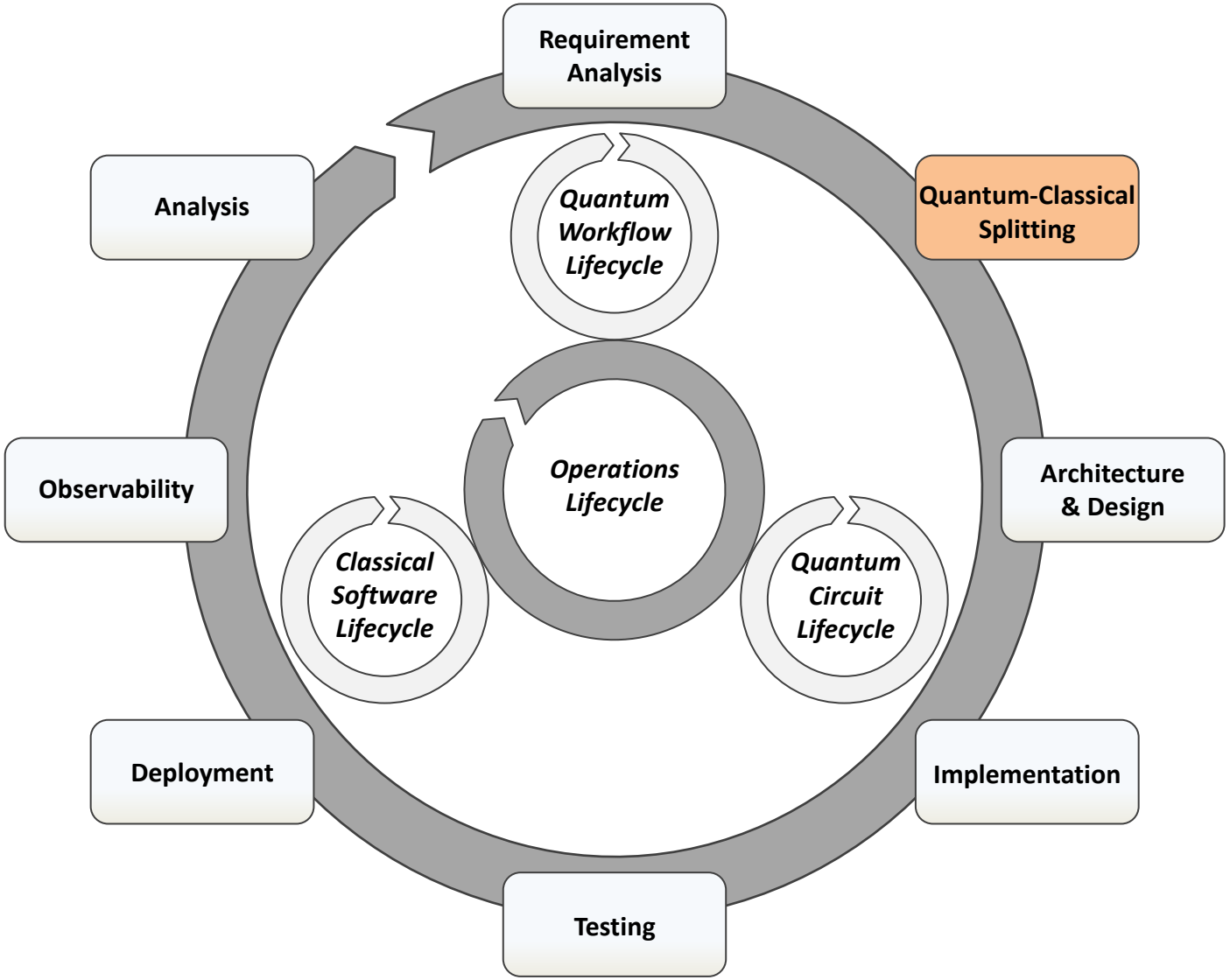
Quantum Software Lifecycle

Quantum Software Lifecycle – Interwoven Lifecycle



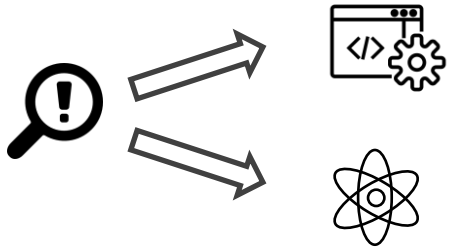
B. Weder, J. Barzen, F. Leymann, and D. Vietz. Quantum Software Development Lifecycle. In: Quantum Software Engineering, Springer, 2022

Quantum Software Lifecycle – Interwoven Lifecycle



Quantum-Classical Splitting

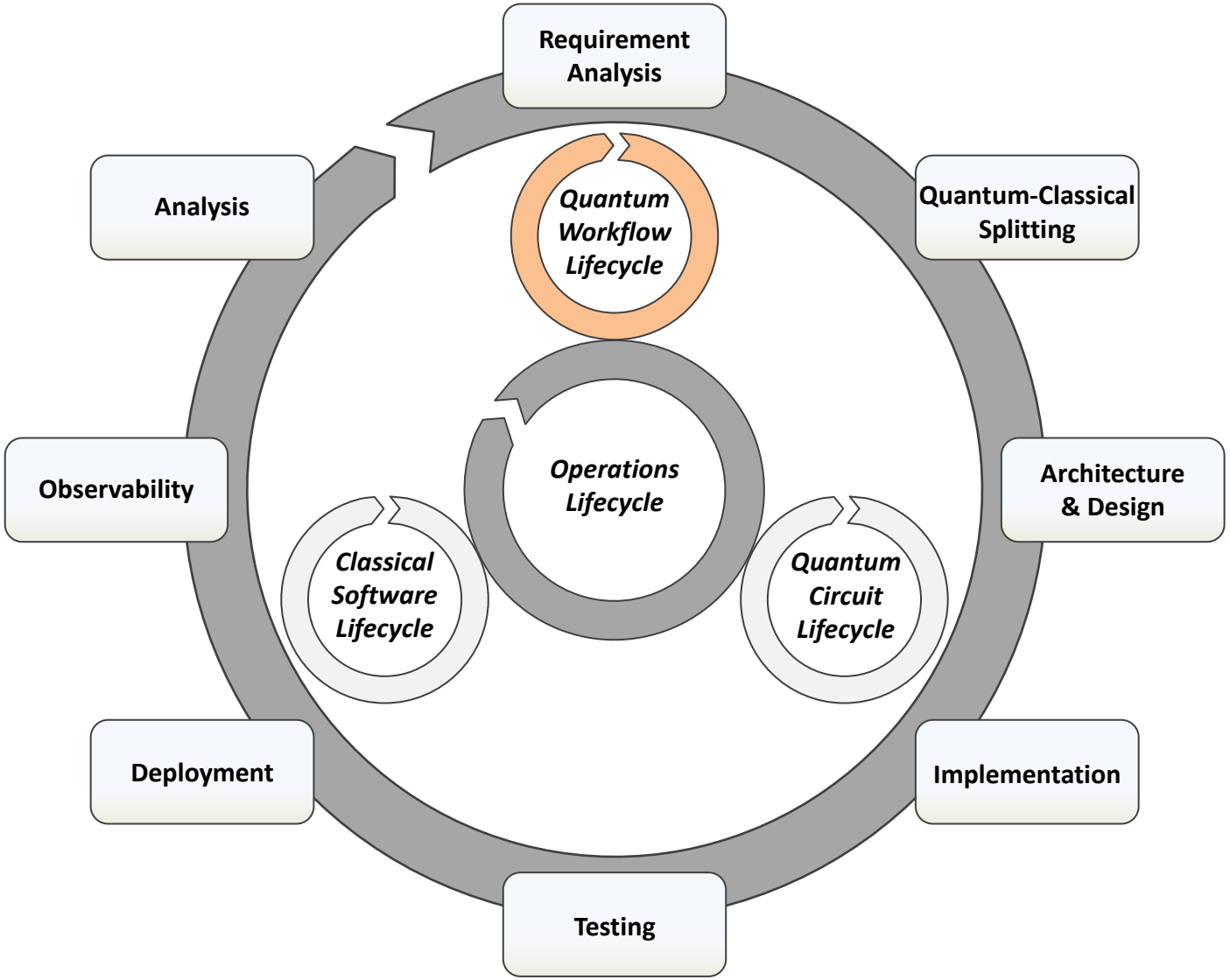
- Entered by the user with the identified requirements
- Split problem to solve into quantum and classical parts:



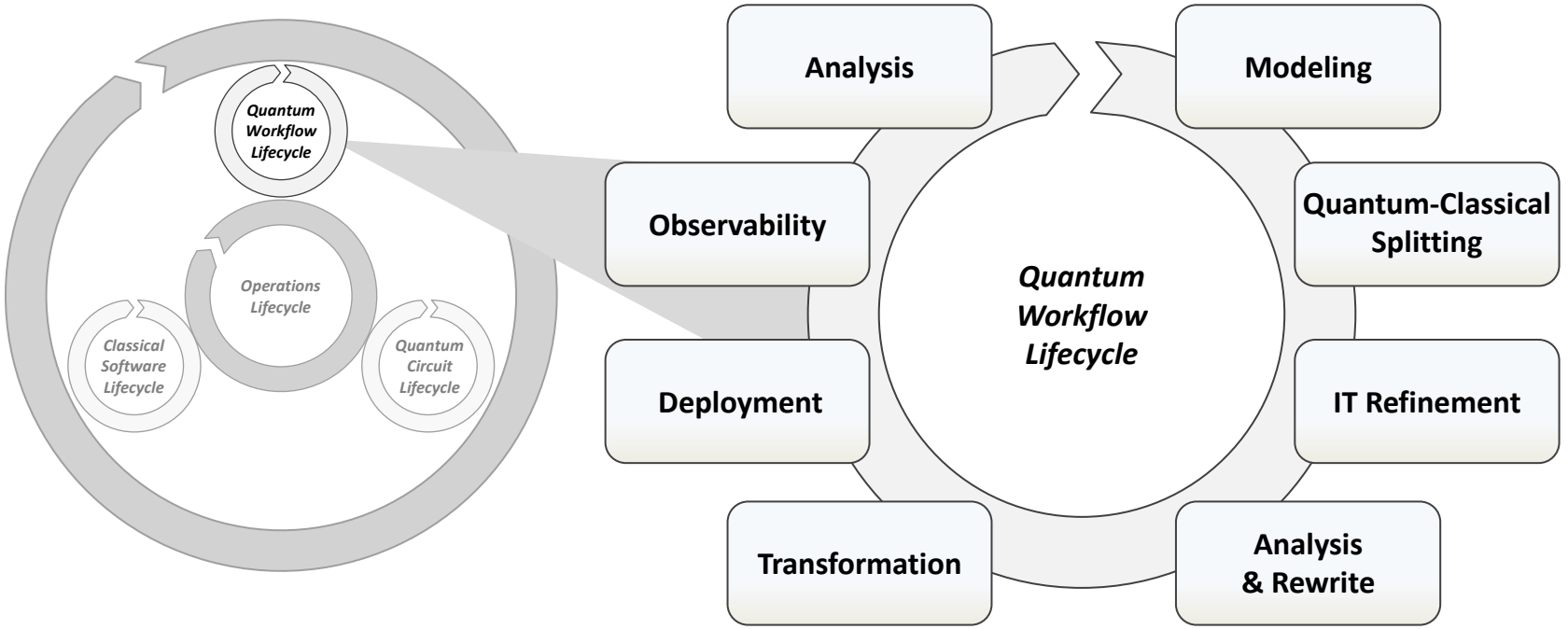
- Different techniques:
  - Manually by experts
  - Comparison of quantum algorithms with classical algorithms ➔ QHAna
  - Automated recommender (based on patterns, provenance, …)

J. Barzen. From Digital Humanities to Quantum Humanities: Potentials and Applications. In: Quantum Computing in the Arts and Humanities

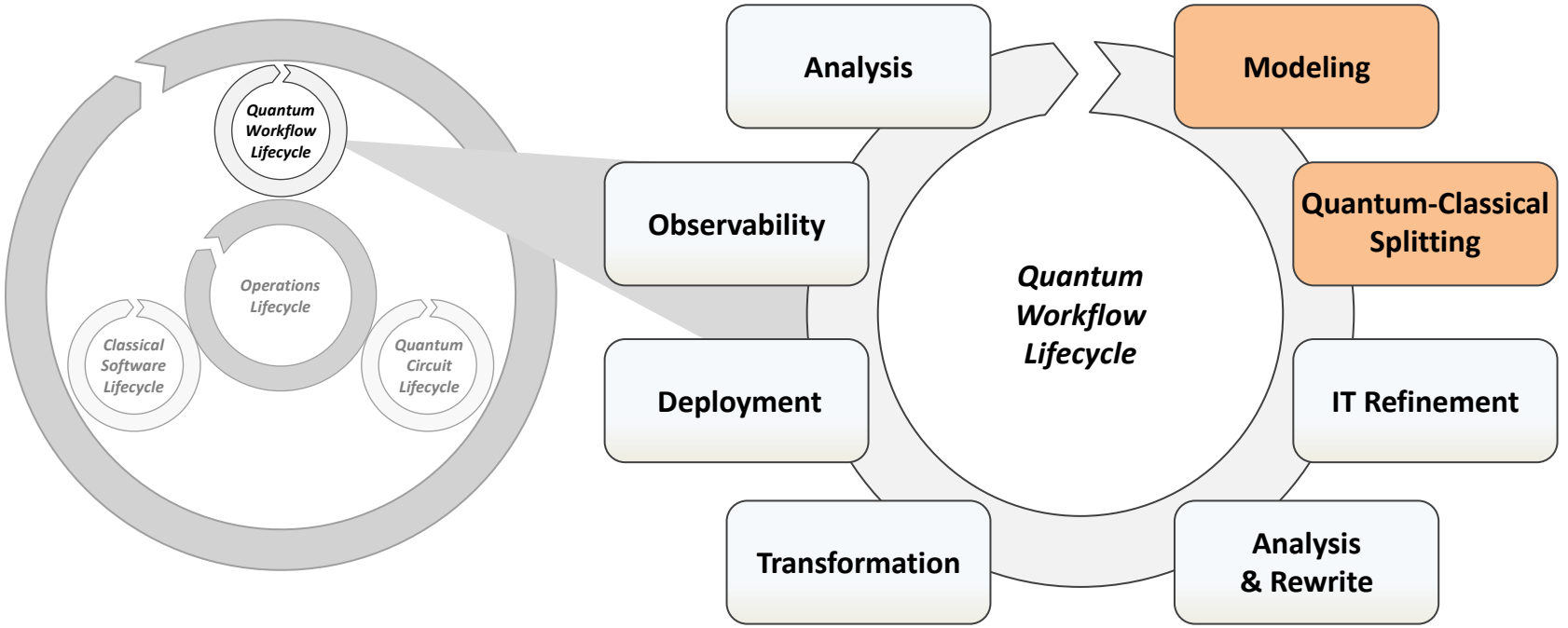
Quantum Software Lifecycle – Interwoven Lifecycle



Detailed View of the Quantum Workflow Lifecycle

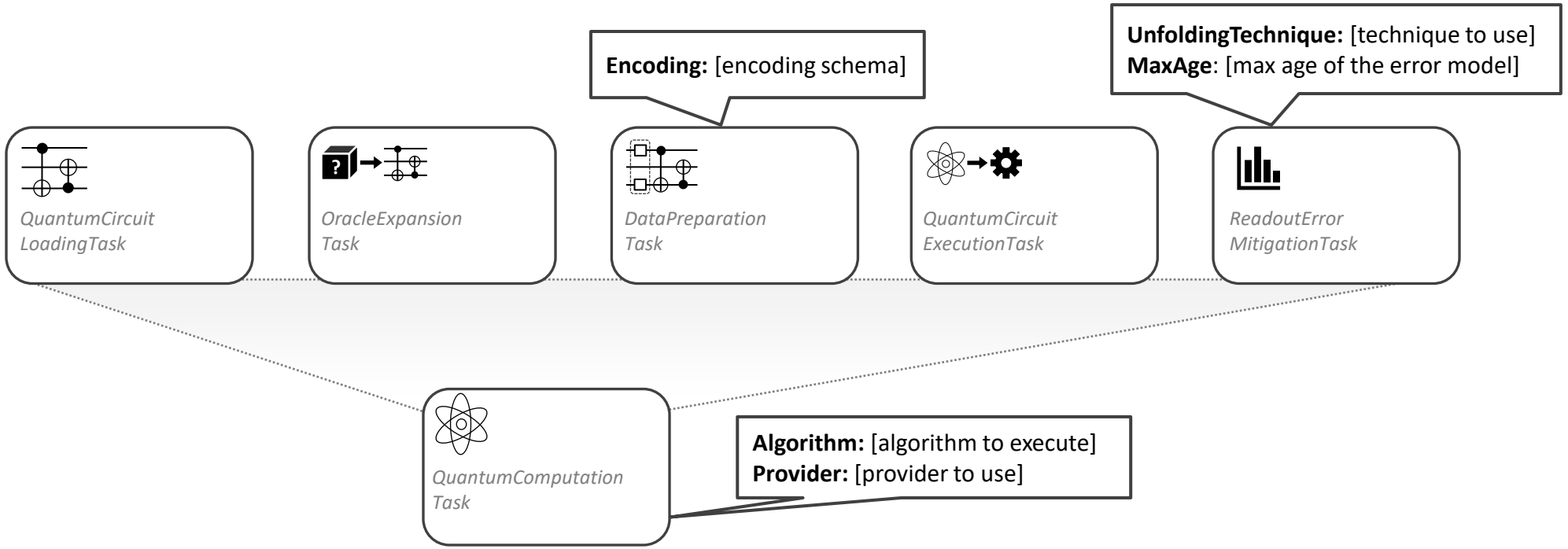


Detailed View of the Quantum Workflow Lifecycle



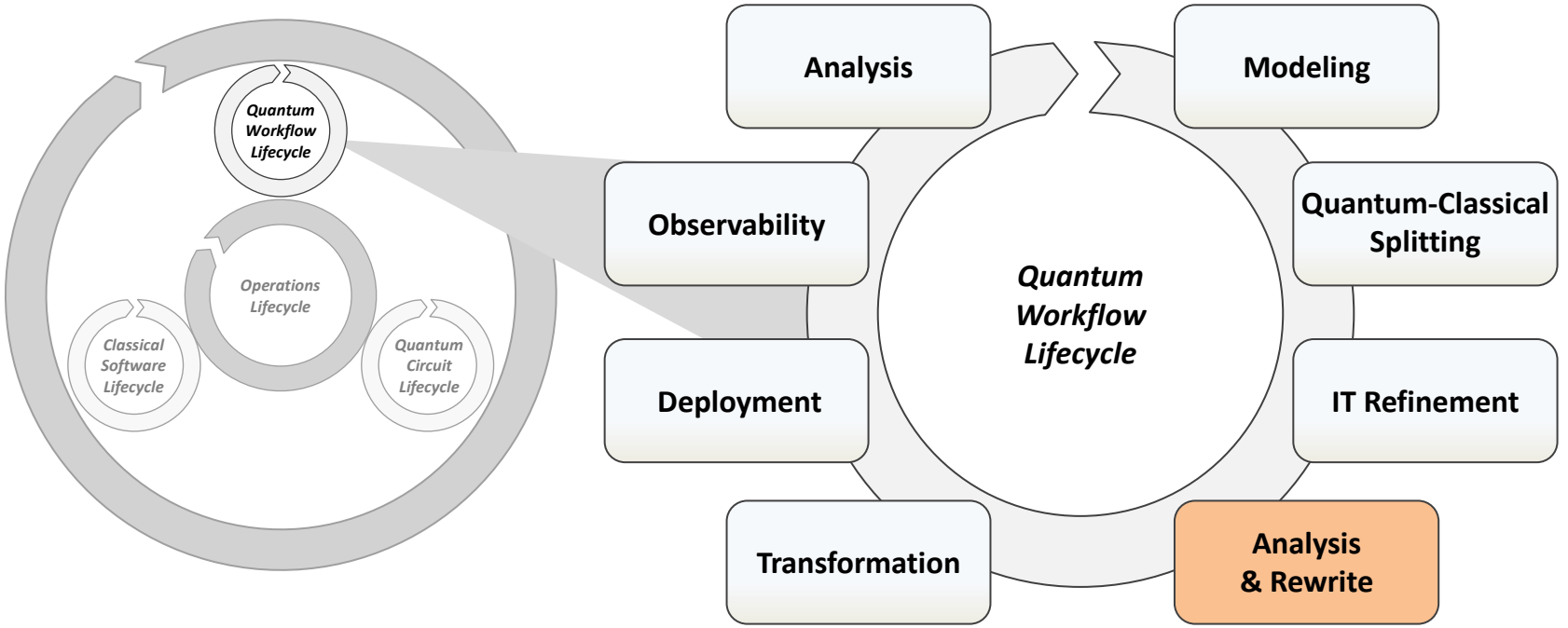
Quantum Modeling Extension (QuantME)

- Guide quantum experts to model the execution of quantum algorithms in workflows
- Hide technical details when invoking predefined quantum algorithms



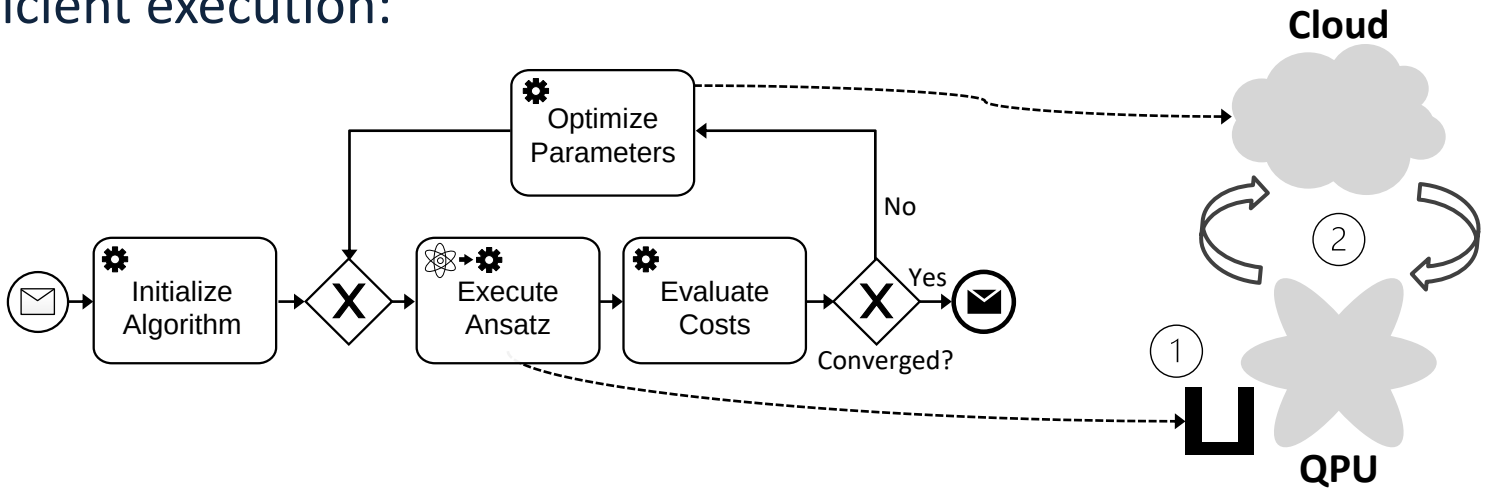
B. Weder, U. Breitenbücher, F. Leymann, and K. Wild. Integrating Quantum Computing into Workflow Modeling and Execution. In: UCC 2020

Detailed View of the Quantum Workflow Lifecycle



Executing Hybrid Loops using Workflows

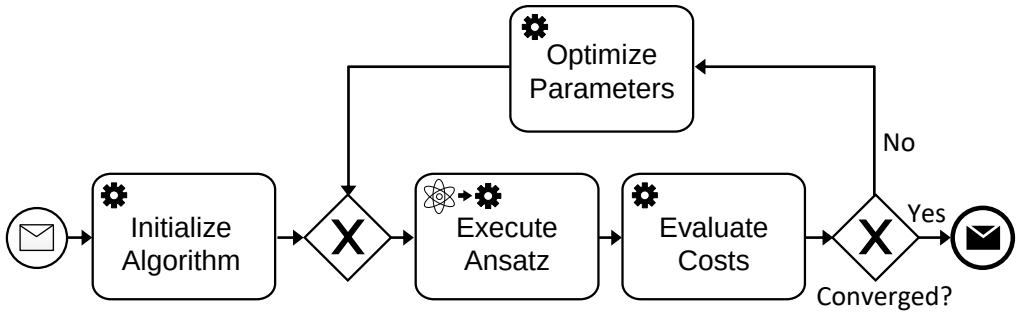
- Inefficient execution:

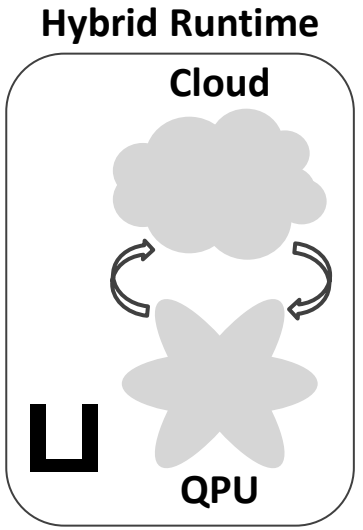


- Problems:
  - 1. Latency due to queue-based access
    ➔ Reserve time slot for complete execution
  - 2. Data transfer between classical and quantum parts inefficient
    ➔ Deploy quantum and classical parts closely together

Executing Hybrid Loops using Workflows

- Inefficient execution:

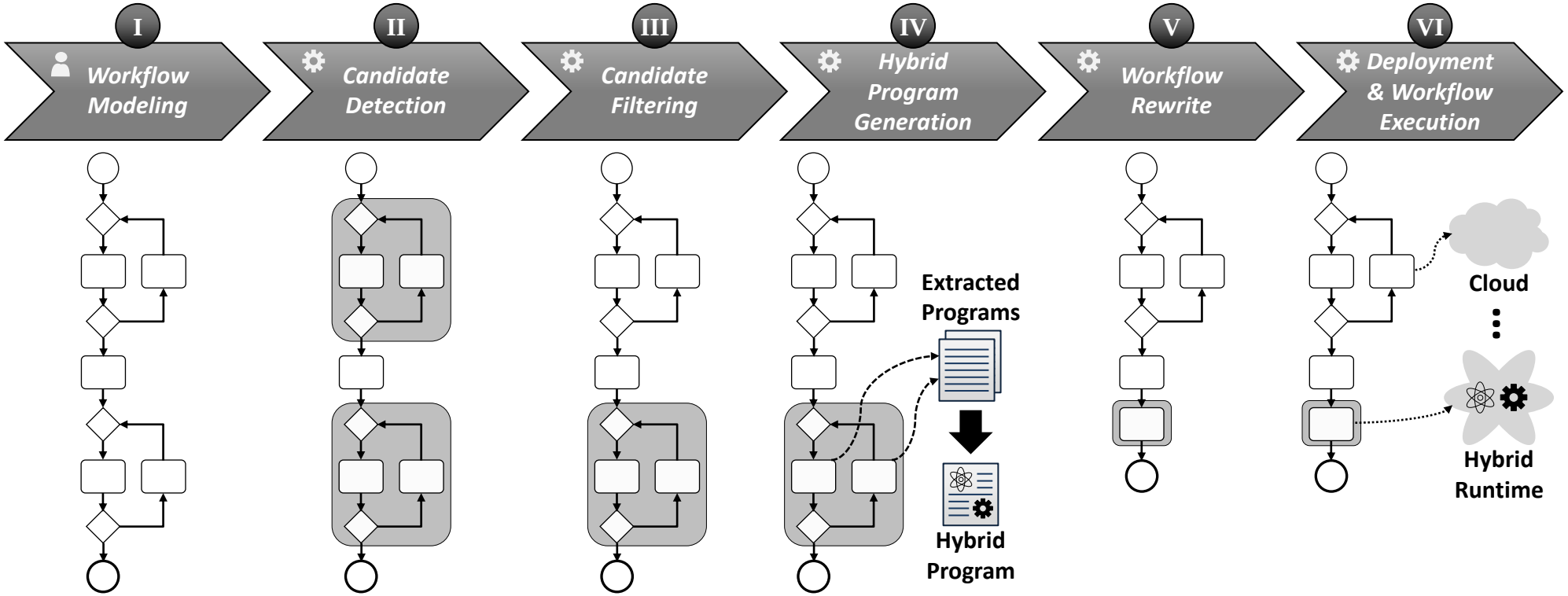




- Problems:
  - 1. Latency due to queue-based access
    ➔ Reserve time slot for complete execution
  - 2. Data transfer between classical and quantum parts inefficient
    ➔ Deploy quantum and classical parts closely together

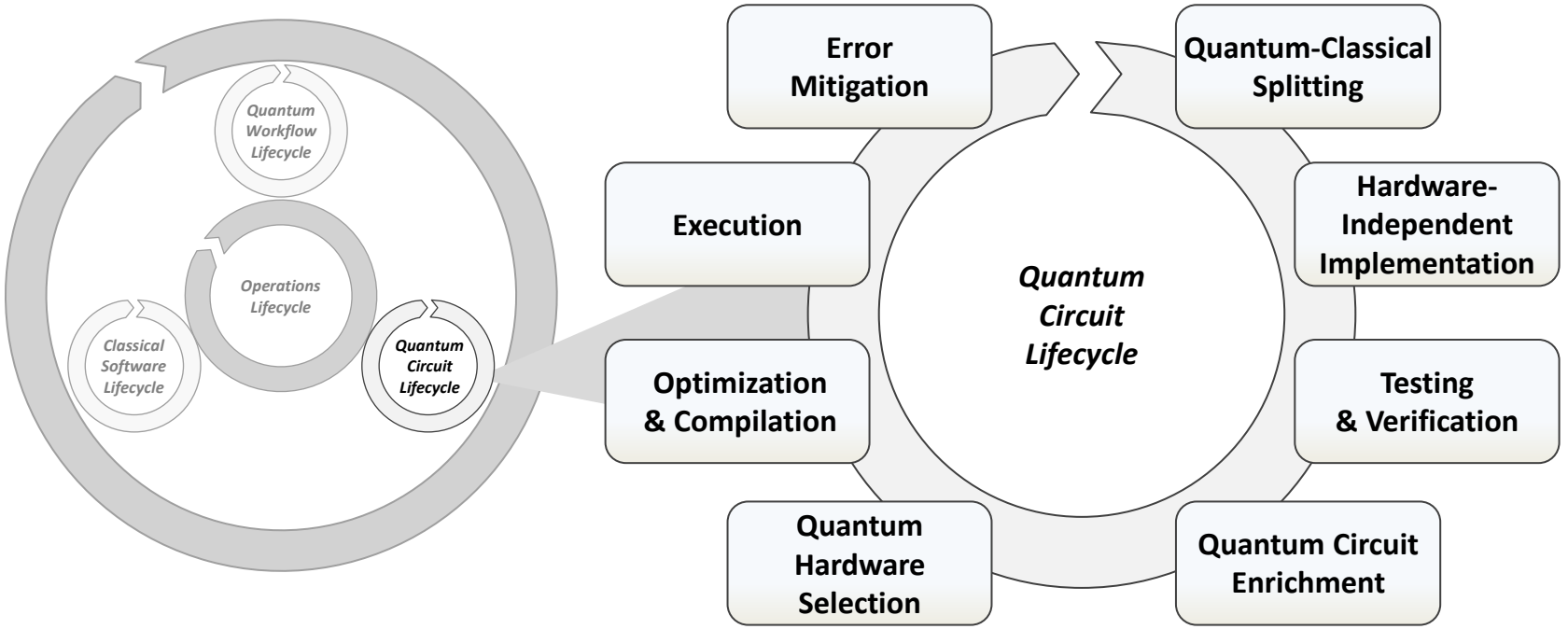
➔ Usage of hybrid runtimes (e.g., Qiskit Runtime) for execution of hybrid loops

Analysis and Rewrite Method

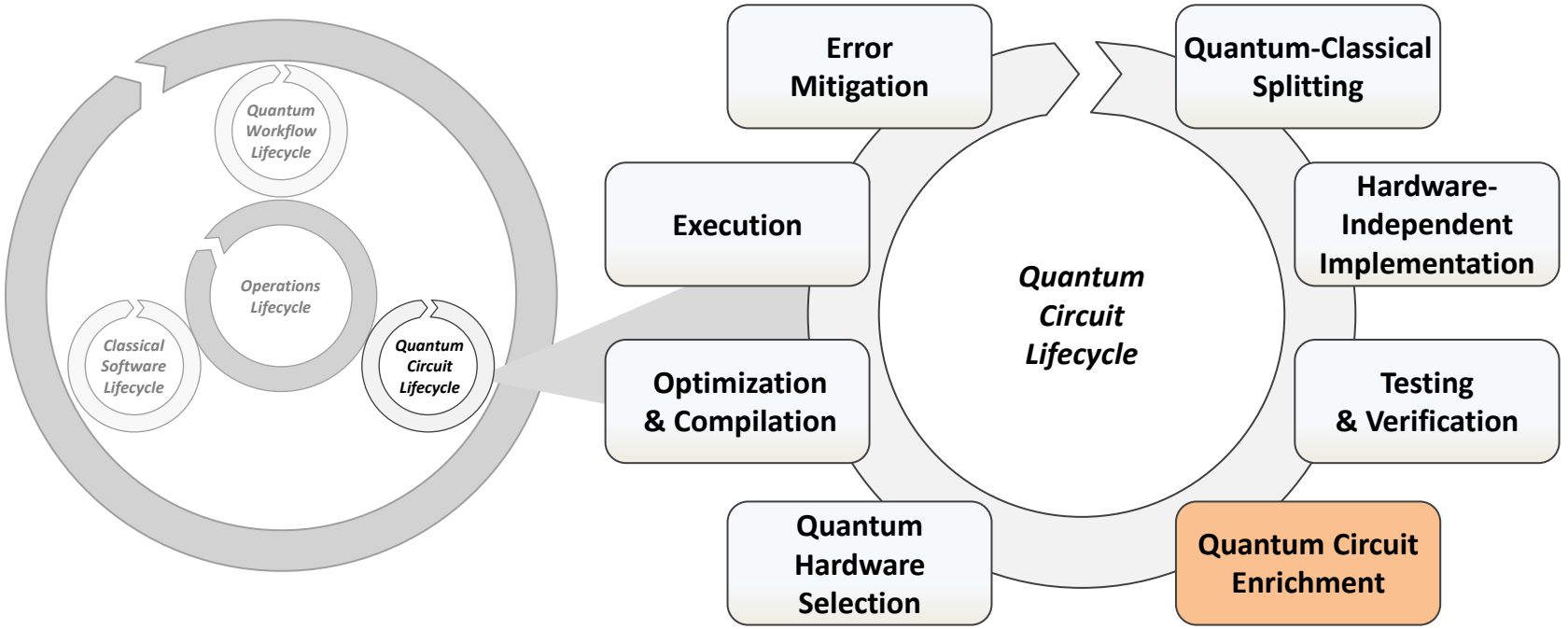


B. Weder, J. Barzen, M. Beisel, and F. Leymann. Analysis and Rewrite of Quantum Workflows: Improving the Execution of Hybrid Quantum Algorithms. In: CLOSER 2022

Detailed View of the Quantum Circuit Lifecycle

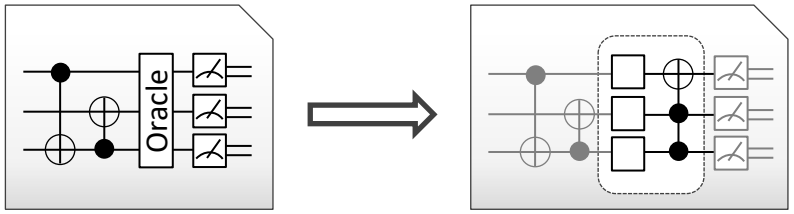


Detailed View of the Quantum Circuit Lifecycle

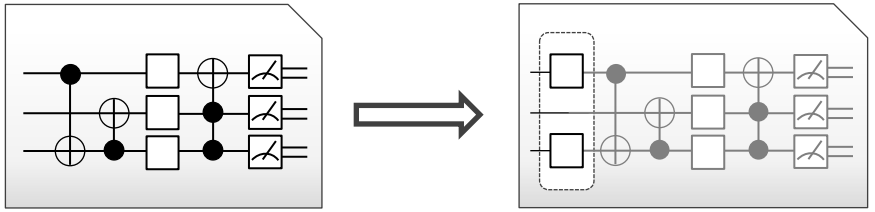


Quantum Circuit Enrichment

- Enrichment with details for a certain problem instance
- Oracle expansion:

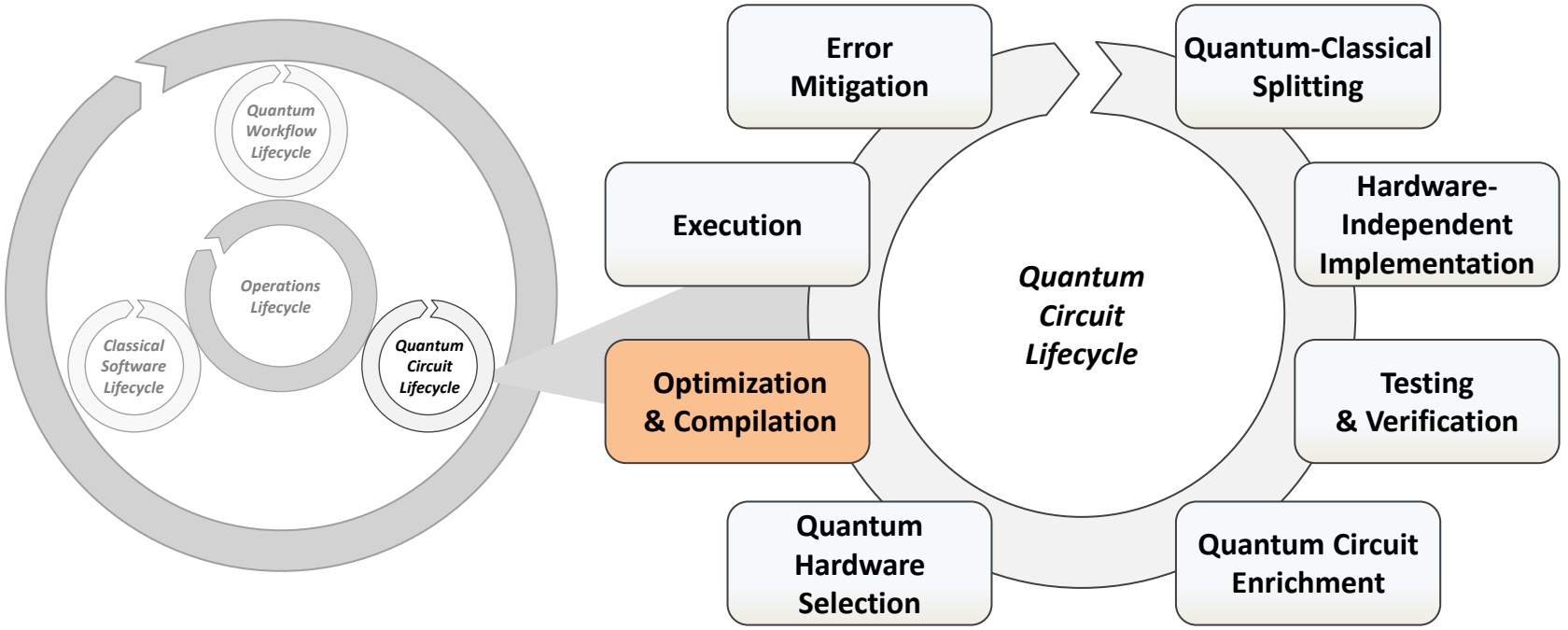


- Data preparation:
  - Adding an initialization circuit to the beginning of the original circuit
  - Different encodings: basis encoding, angle encoding, …



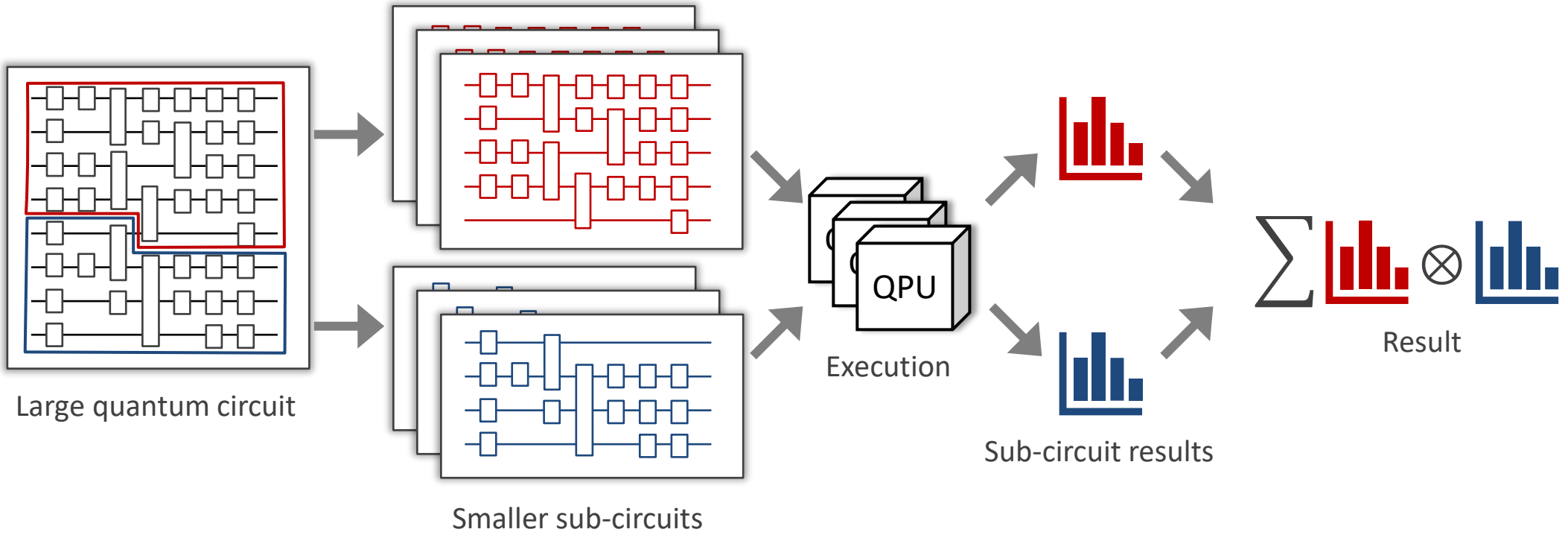
B. Weder, J. Barzen, F. Leymann, M. Salm, and D. Vietz. The Quantum Software Lifecycle. In: APEQS 2020

Detailed View of the Quantum Circuit Lifecycle



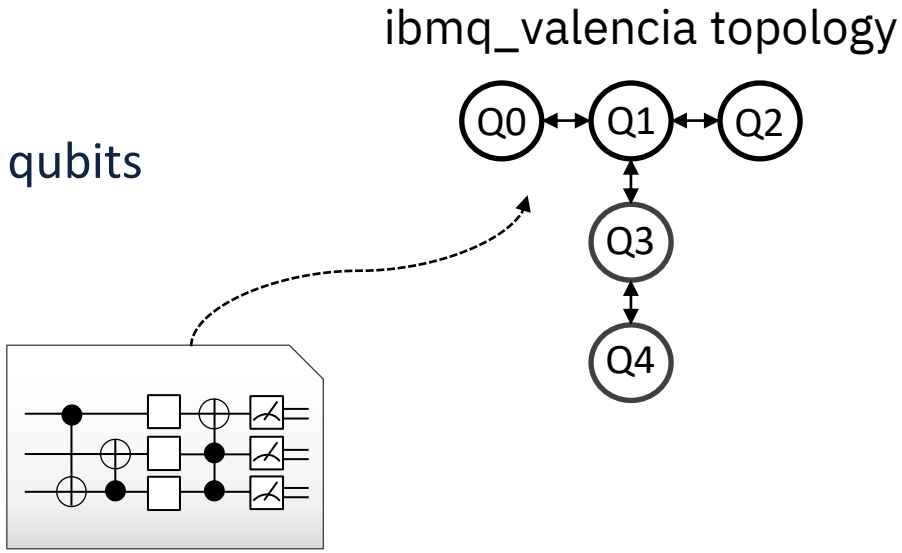
Optimization: Cutting Quantum Circuits

- Quantum circuits might be too large (width, depth) to retrieve good results
- Execute multiple smaller circuits
  ➔ Classical post-processing to combine results

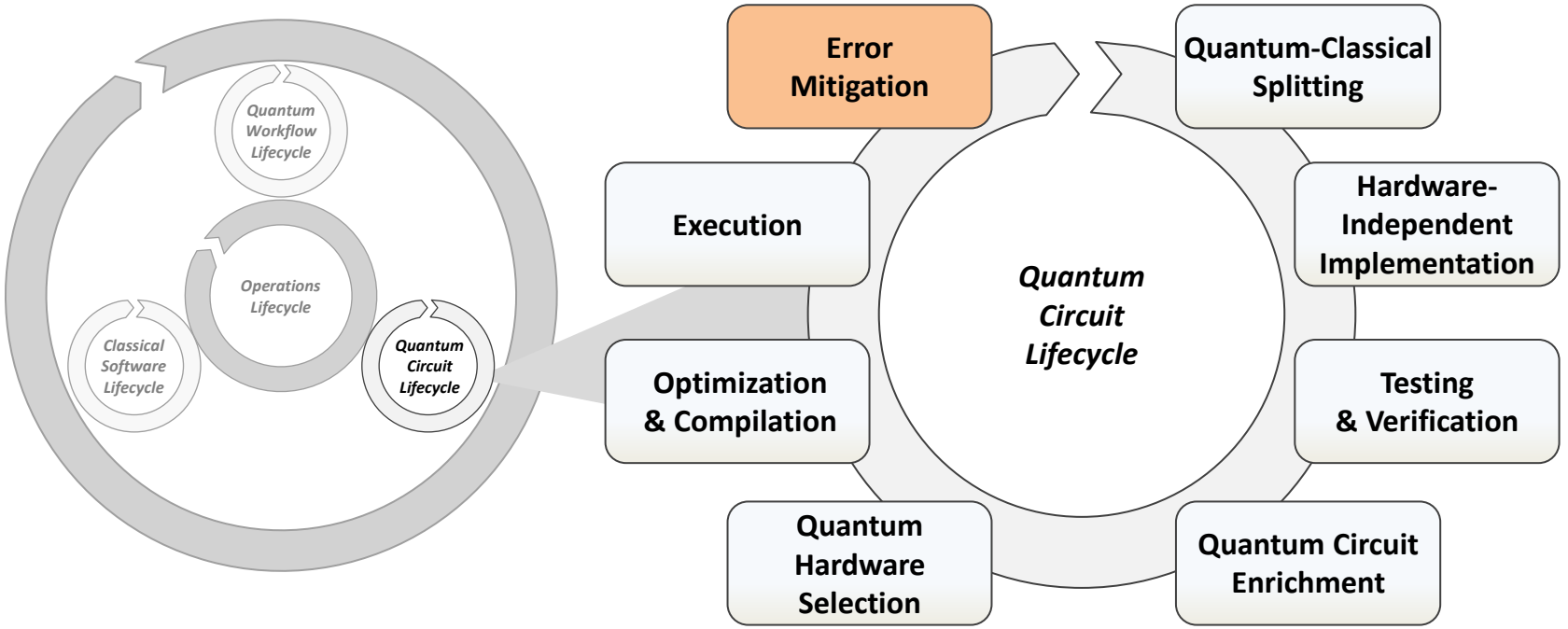


Compilation of Quantum Circuits

- Compilation to machine instructions required
- Replacement of not physically implemented gates
- Qubit allocation on the quantum computer
- Optimization based on:
  - Decoherence times of different qubits
  - Gate fidelities
  - Qubit connectivity

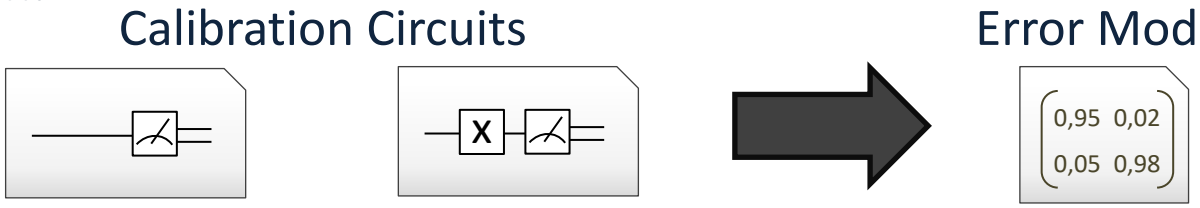


Detailed View of the Quantum Circuit Lifecycle

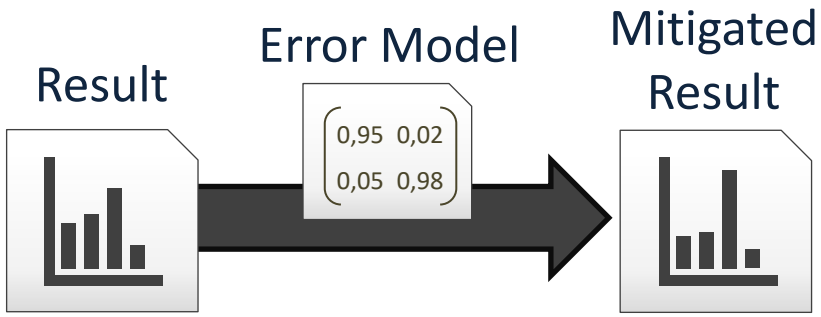


Error-Mitigation

- Reduce impact of errors based on data about the quantum computer
- Example: Readout-error mitigation using the calibration matrix
  - Data collection:

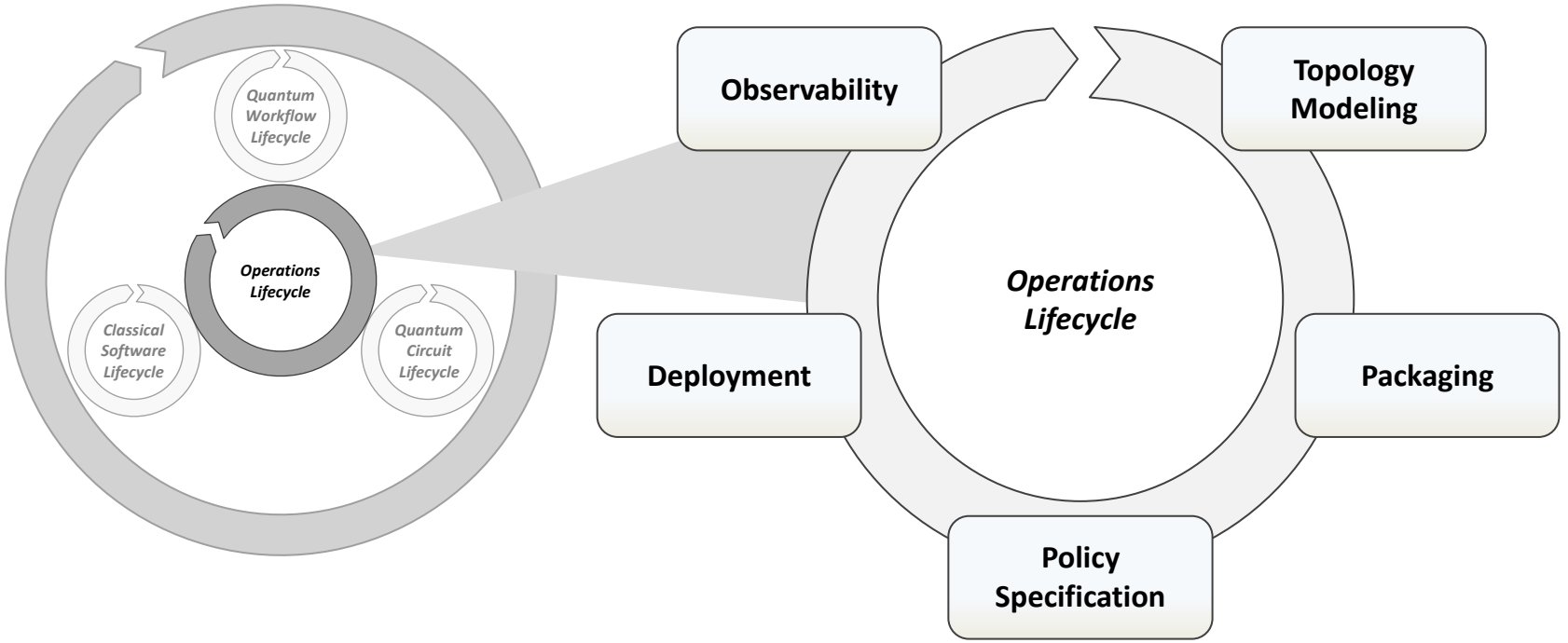


  - Mitigating the result:

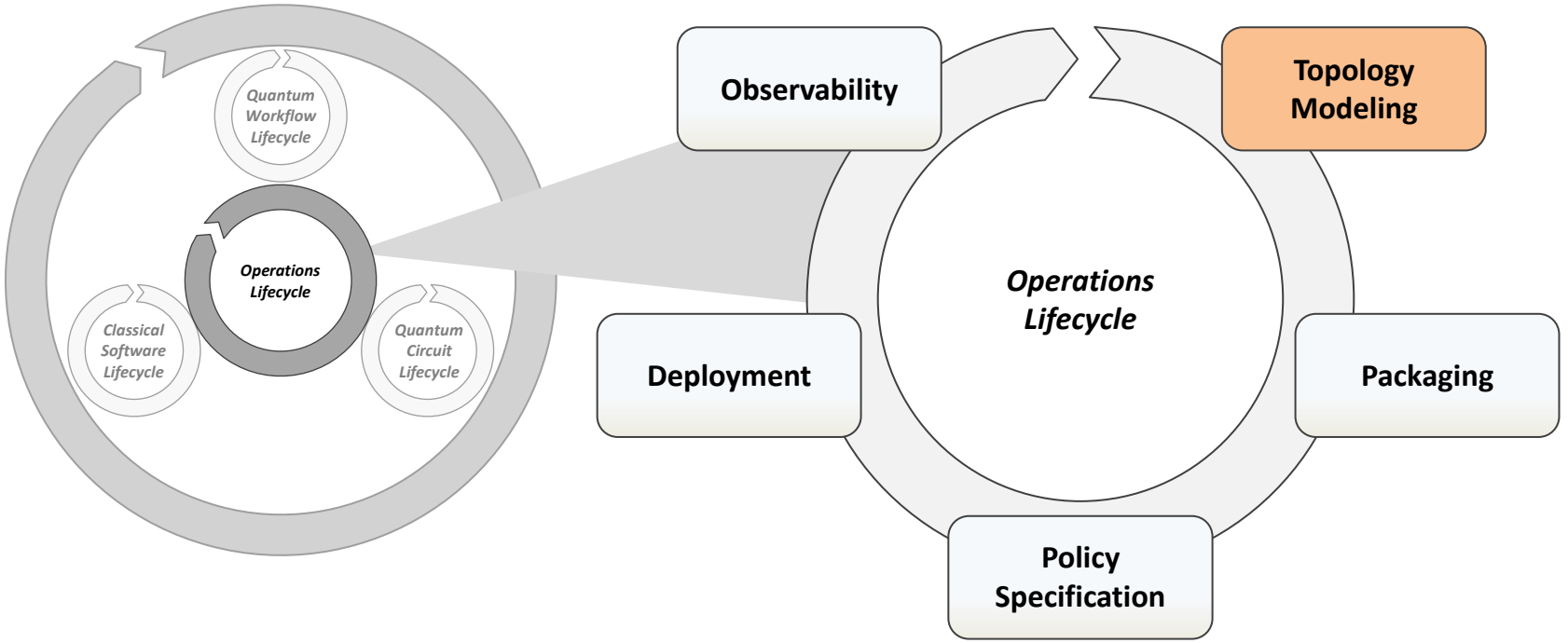


M. Beisel, J. Barzen, F. Leymann, F. Truger, B. Weder, and V. Yussupov. Patterns for Quantum Error Handling. In: PATTERNS 2022

Detailed View of the Operations Lifecycle

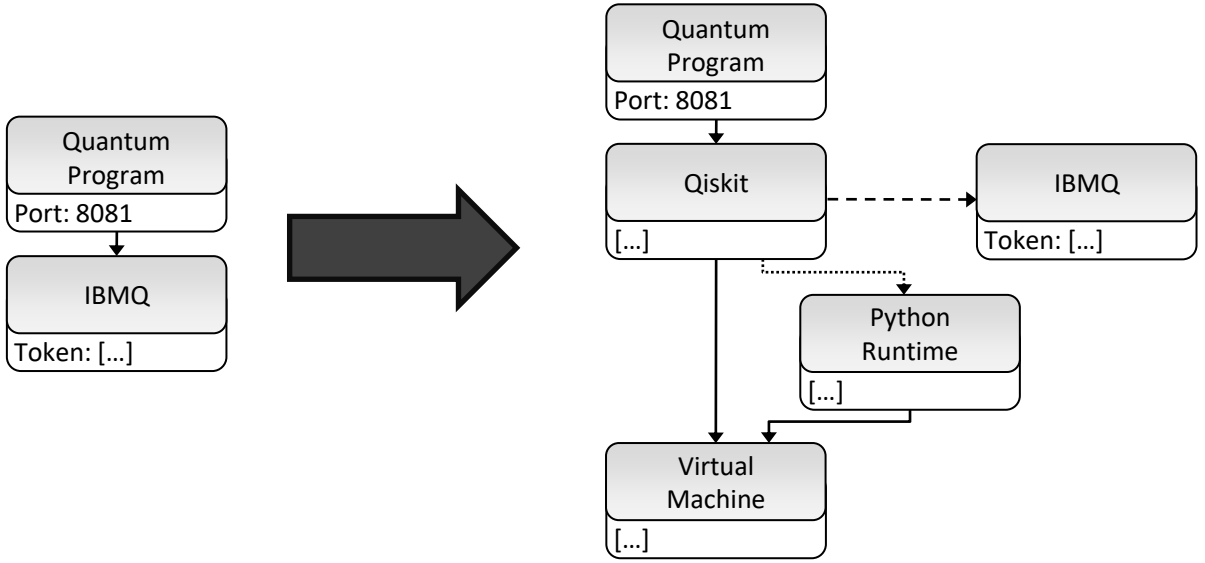


Detailed View of the Operations Lifecycle



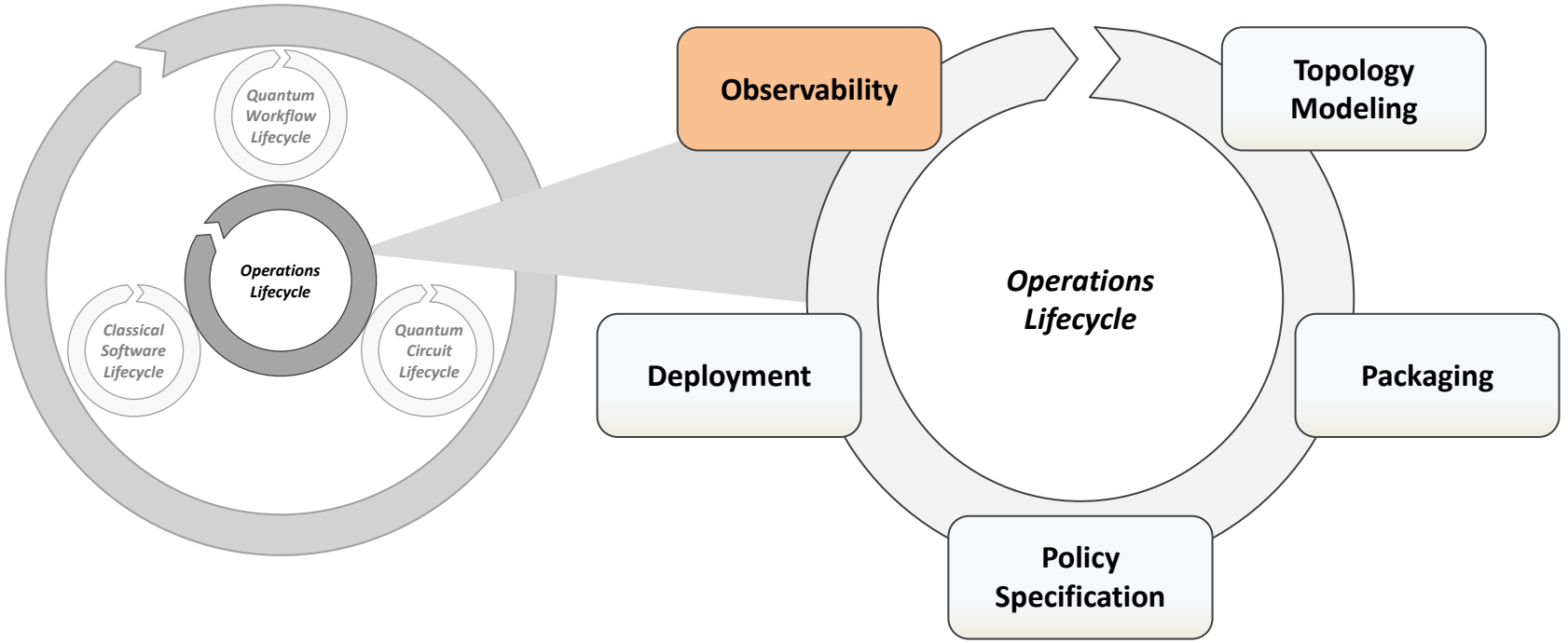
Topology Modeling – TOSCA4QC

- Quantum programs are often deployed when they are invoked
- Common modeling principles do not apply
- *TOSCA4QC*:
  - Introcude two modeling styles for quantum applications
  - Automatic transformation between them



K. Wild, U. Breitenbücher, L. Harzenetter, F. Leymann, D. Vietz, and M. Zimmermann. TOSCA4QC: Two Modeling Styles for TOSCA to Automate the Deployment and Orchestration of Quantum Applications. In: EDOC 2020

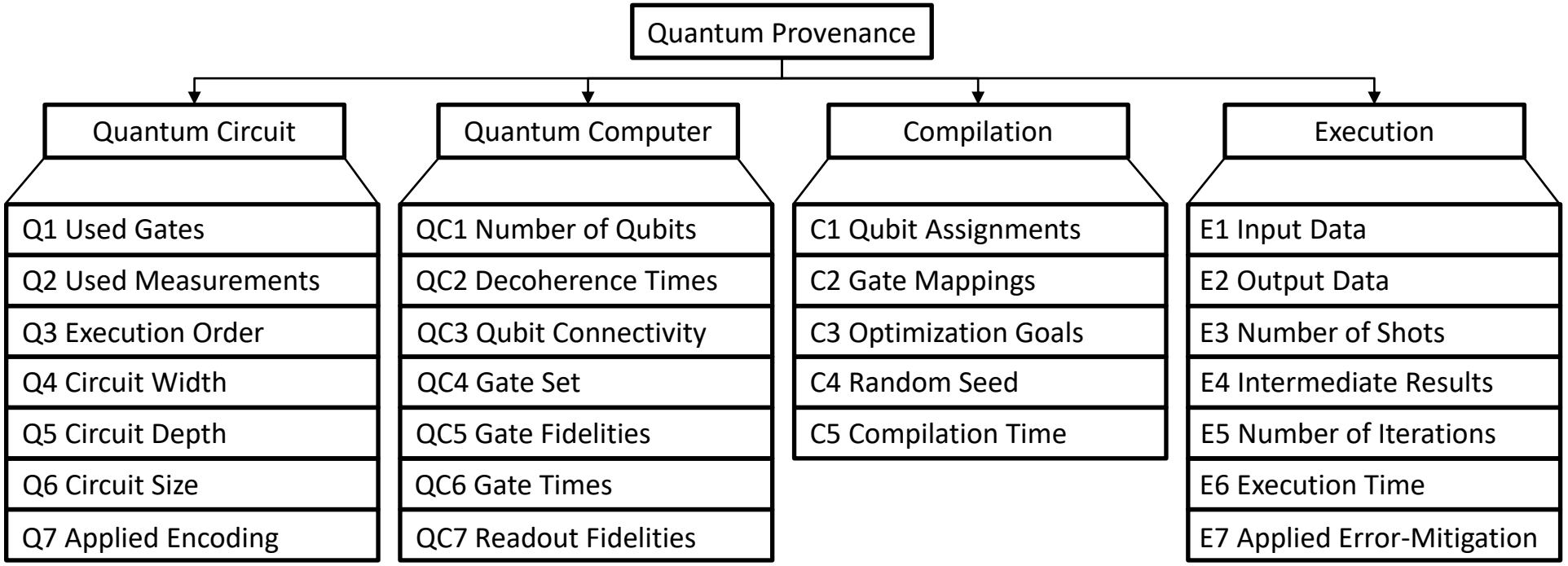
Detailed View of the Operations Lifecycle



Quantum Provenance

- Provenance
  - Information describing a process or computation
  - Goals: reproducibility, understandability, quality
- Especially important for quantum computing
  - Noisy devices (decoherence, gate errors, …)
  - Different realizations (trapped ions, superconducting, …)
- Example use cases:
  - Quantum hardware selection
  - Readout-error mitigation
  - Optimization & compilation

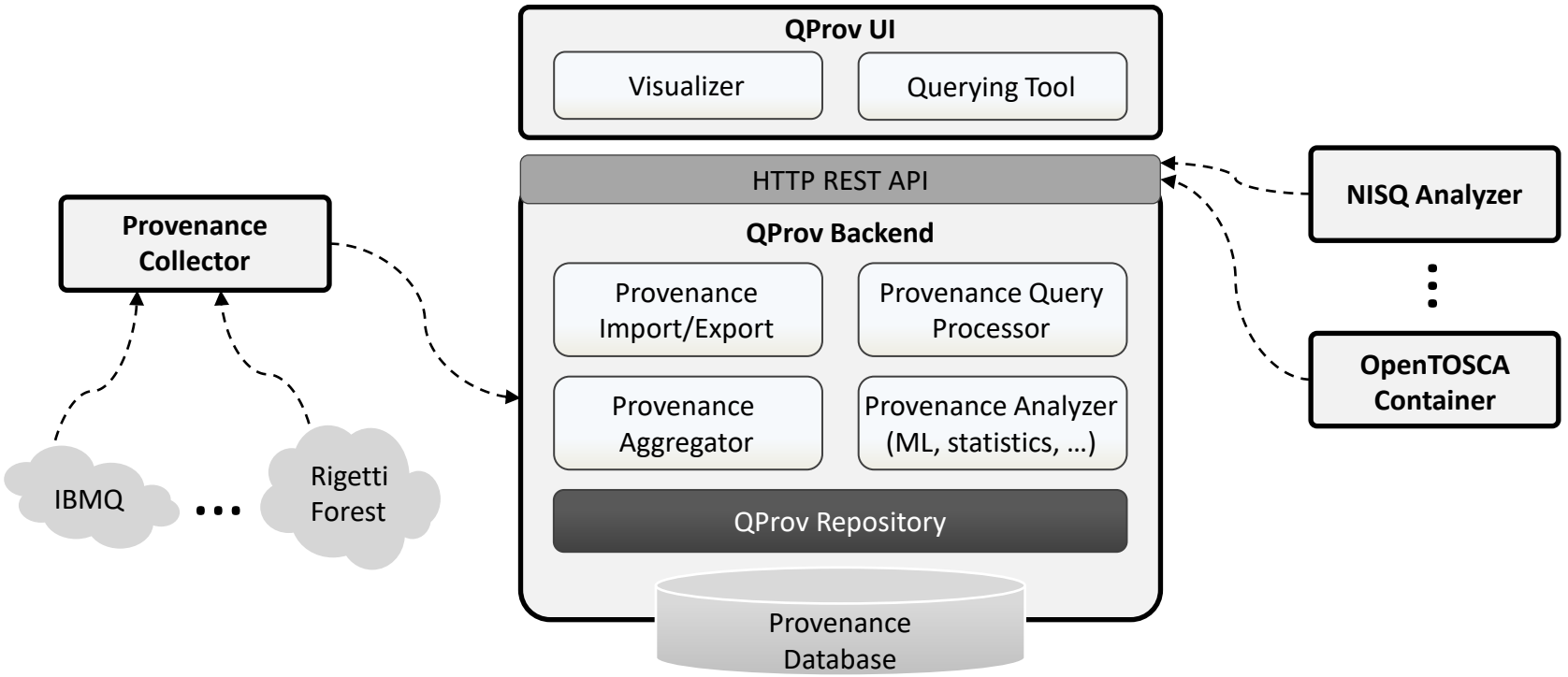
Quantum Provenance Attributes



B. Weder, J. Barzen, F. Leymann, M. Salm, and K. Wild. QProv: A provenance system for quantum computing. In: IET Quantum Communications 2021

The QProv System

- Quantum provenance framework:
  - Continuously gather all required data
  - E.g., through the provider API, by executing calibration circuits, …



B. Weder, J. Barzen, F. Leymann, M. Salm, and K. Wild. QProv: A provenance system for quantum computing. In: IET Quantum Communications 2021

Conclusion & Outlook

Conclusion & Outlook

- Quantum application development is complex and requires experts from different fields
- Common understanding of the various phases and tasks is needed
- *Quantum Software Lifecycle*:
  - Interwoven lifecycles
  - Workflow, classical, quantum circuit, operations lifecycles
- Future work:
  - Many open problems, e.g., how to properly split a problem into quantum and classical parts?
  - Tooling support required (e.g., test, circuit cutting, …)

Thank you for your attention ☺