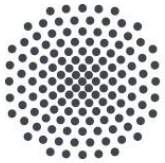
Quantum Software Development Lifecycle



University of Stuttgart

Benjamin Weder

benjamin.weder@iaas.uni-stuttgart.de

Institute of Architecture of Application Systems

PlanQK

SeQuenC

EniQmA

Tutorial Structure

- Session 1 (09:00 - 10:30): An Introduction to Quantum Computing
- **Session 2 (11:00 - 12:30): Quantum Software Engineering**
  - Quantum software development lifecycle
  - Quantum hardware selection
  - Q/A session
  - Outlook to the afternoon sessions
- Session 3 (14:00 - 15:30): Quantum Workflows
- Session 4 (16:00 - 17:30): Operation of Hybrid Quantum Applications

Tutorial Structure

- Session 1 (09:00 - 10:30): An Introduction to Quantum Computing
- **Session 2 (11:00 - 12:30): Quantum Software Engineering**
  - **Quantum software development lifecycle**
  - Quantum hardware selection
  - Q/A session
  - Outlook to the afternoon sessions
- Session 3 (14:00 - 15:30): Quantum Workflows
- Session 4 (16:00 - 17:30): Operation of Hybrid Quantum Applications

Motivation

Motivation

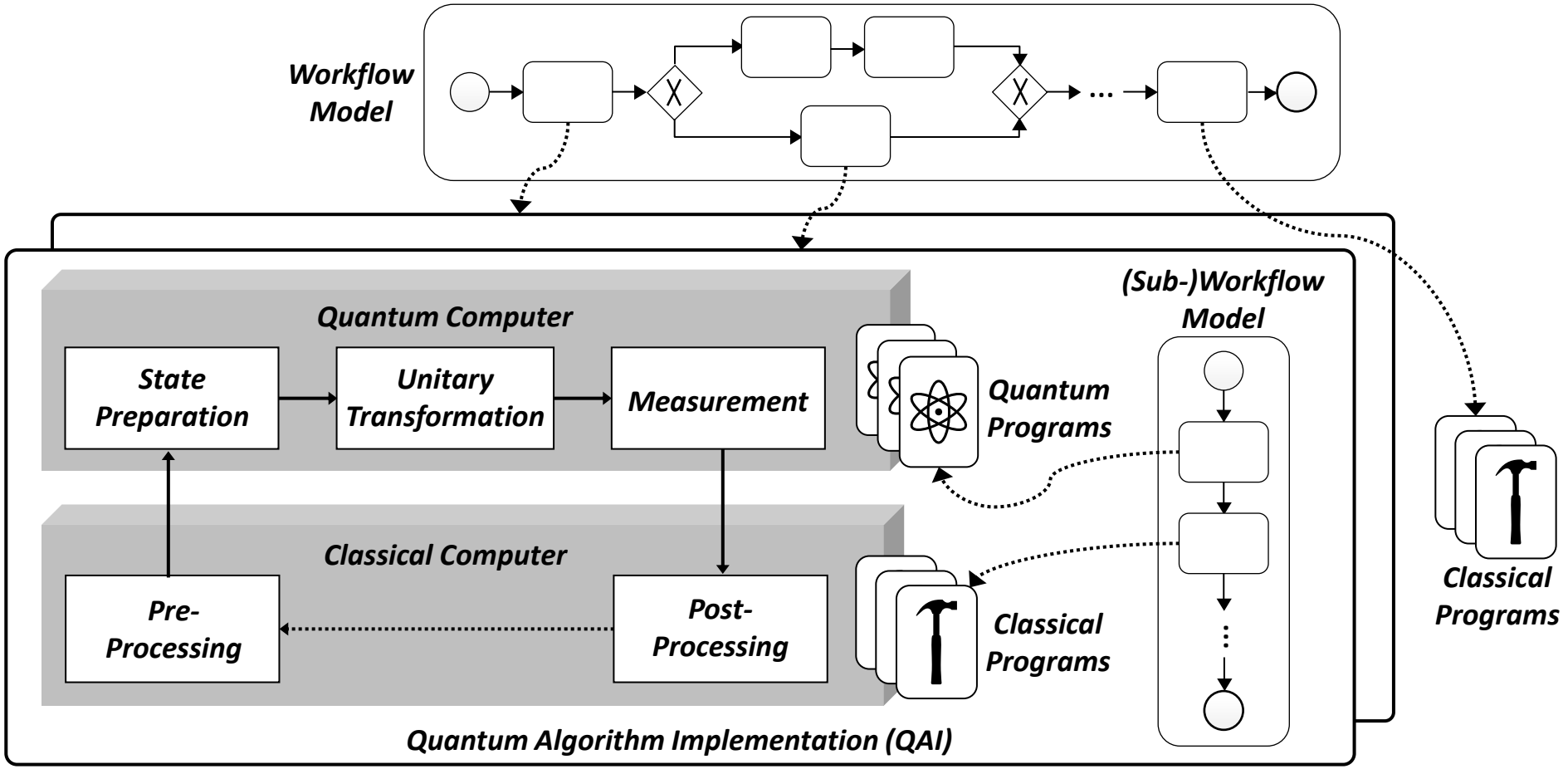
- Recent advances with more powerful quantum computers
  ➔ New quantum applications are needed

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

- Common understanding of the development and execution process needed

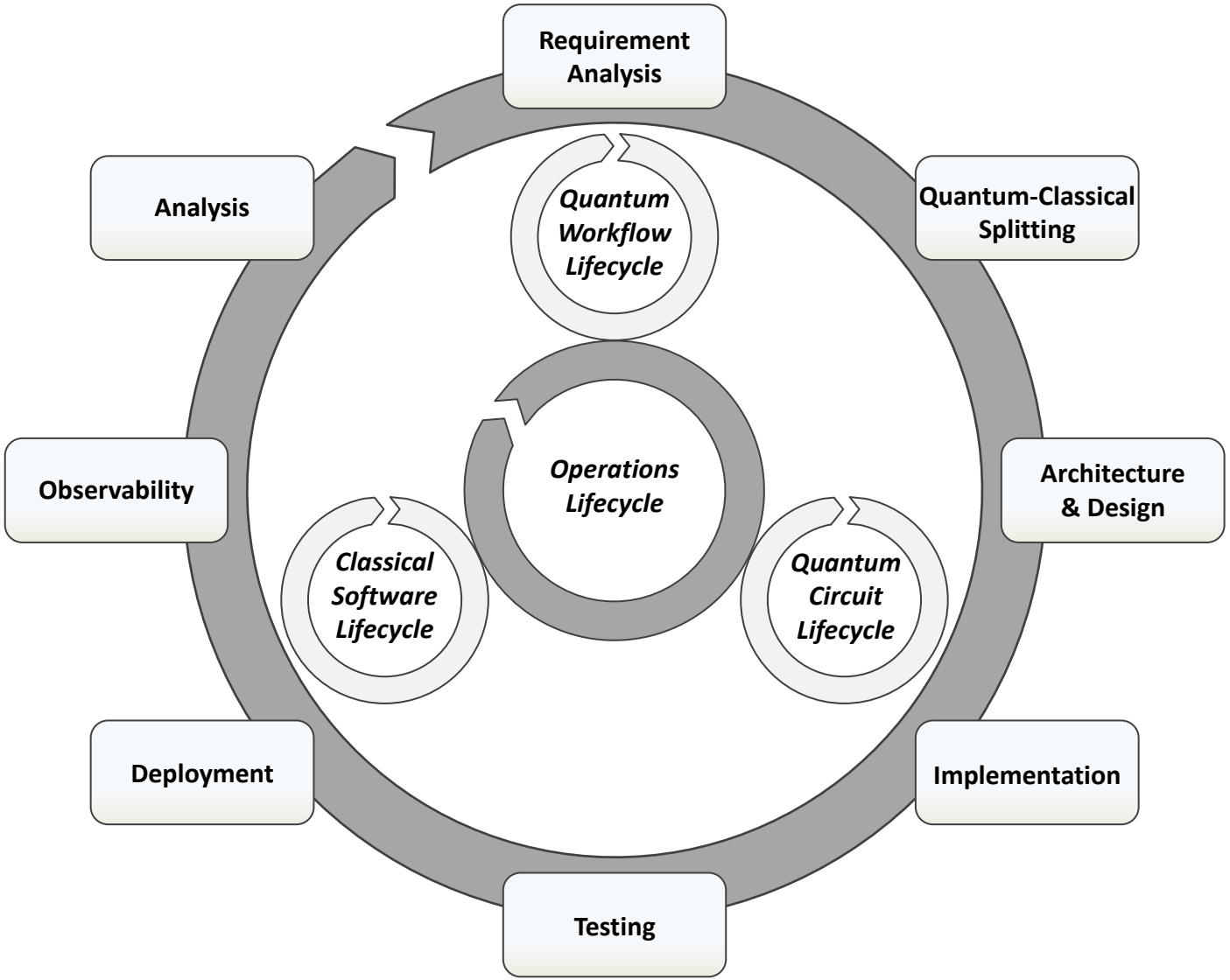
➔ *Quantum Software Development Lifecycle*

Structure of a Hybrid Quantum Application

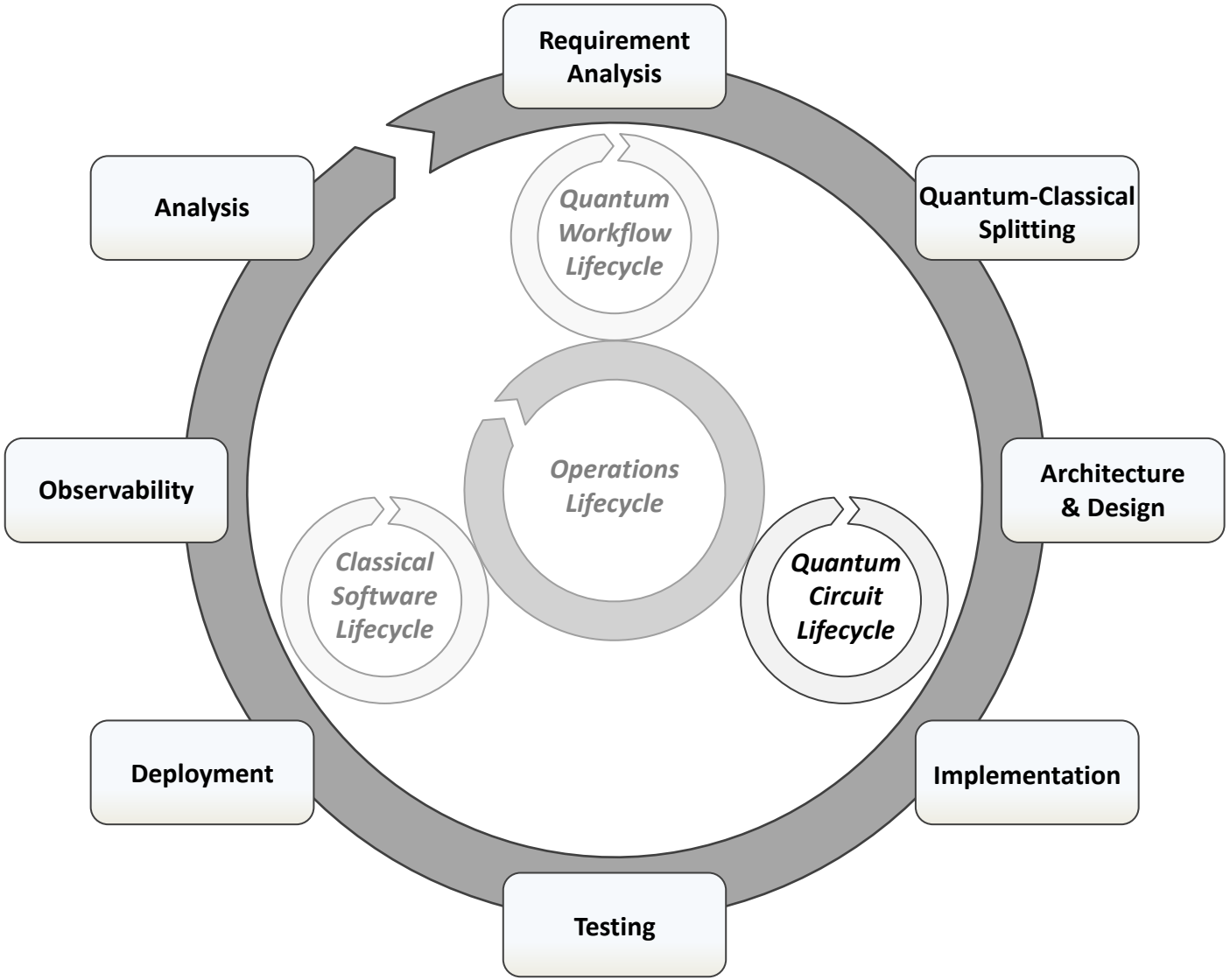


Quantum Software Development Lifecycle

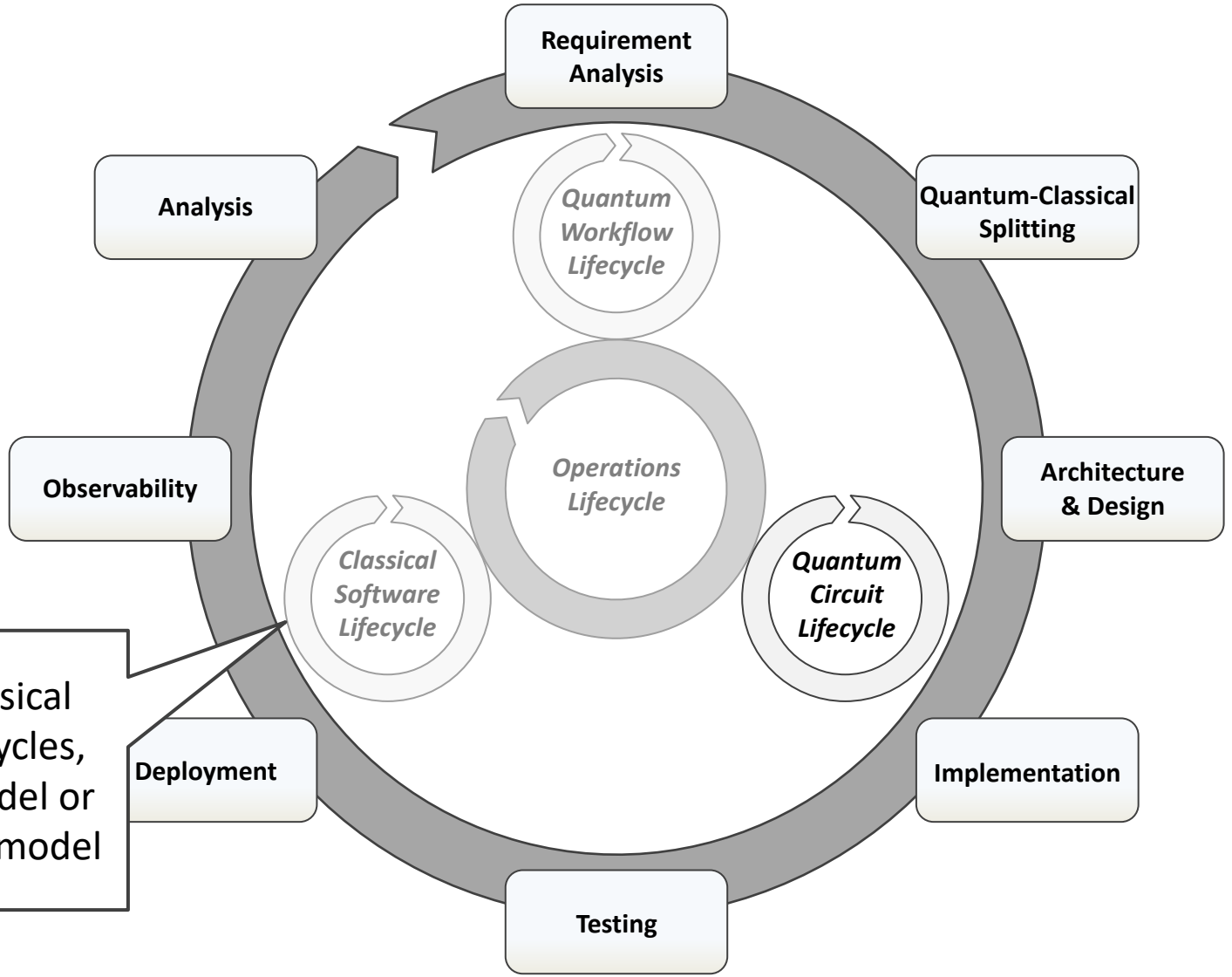
Quantum Software Lifecycle – Interwoven Lifecycle



Focus of this Session

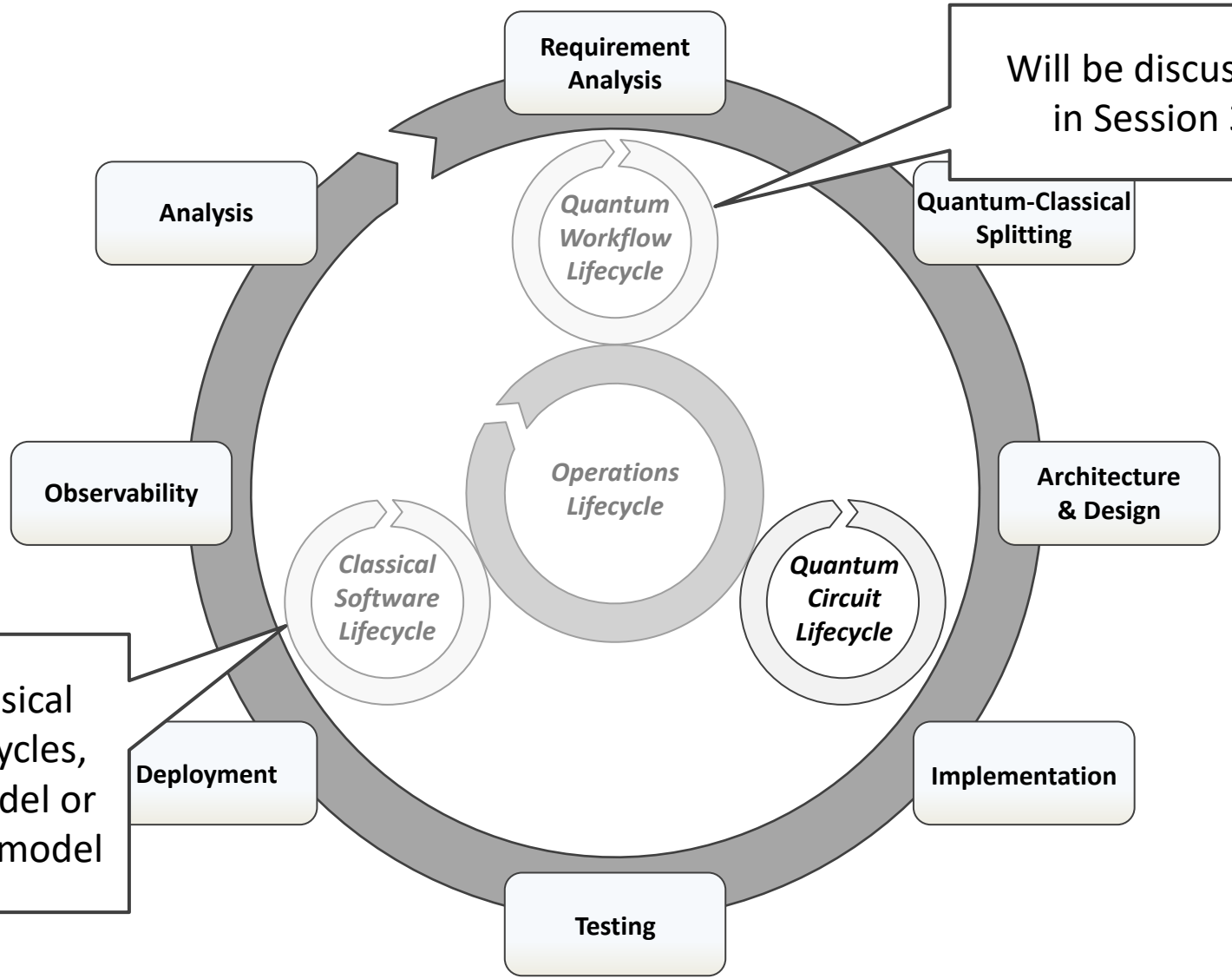


Focus of this Session



Usage of classical software lifecycles, e.g., the V-model or the prototype model

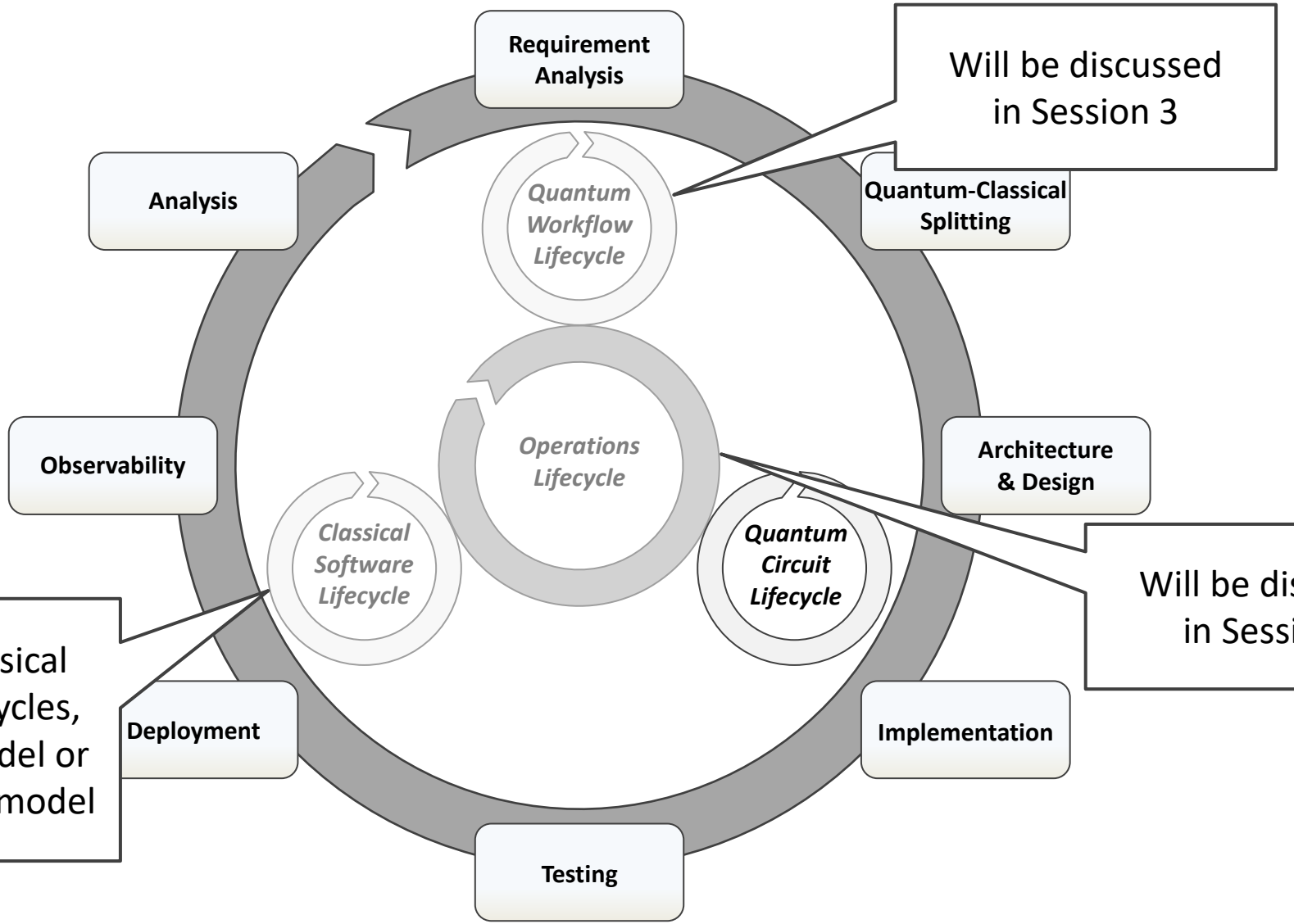
Focus of this Session



Will be discussed in Session 3

Usage of classical software lifecycles, e.g., the V-model or the prototype model

Focus of this Session



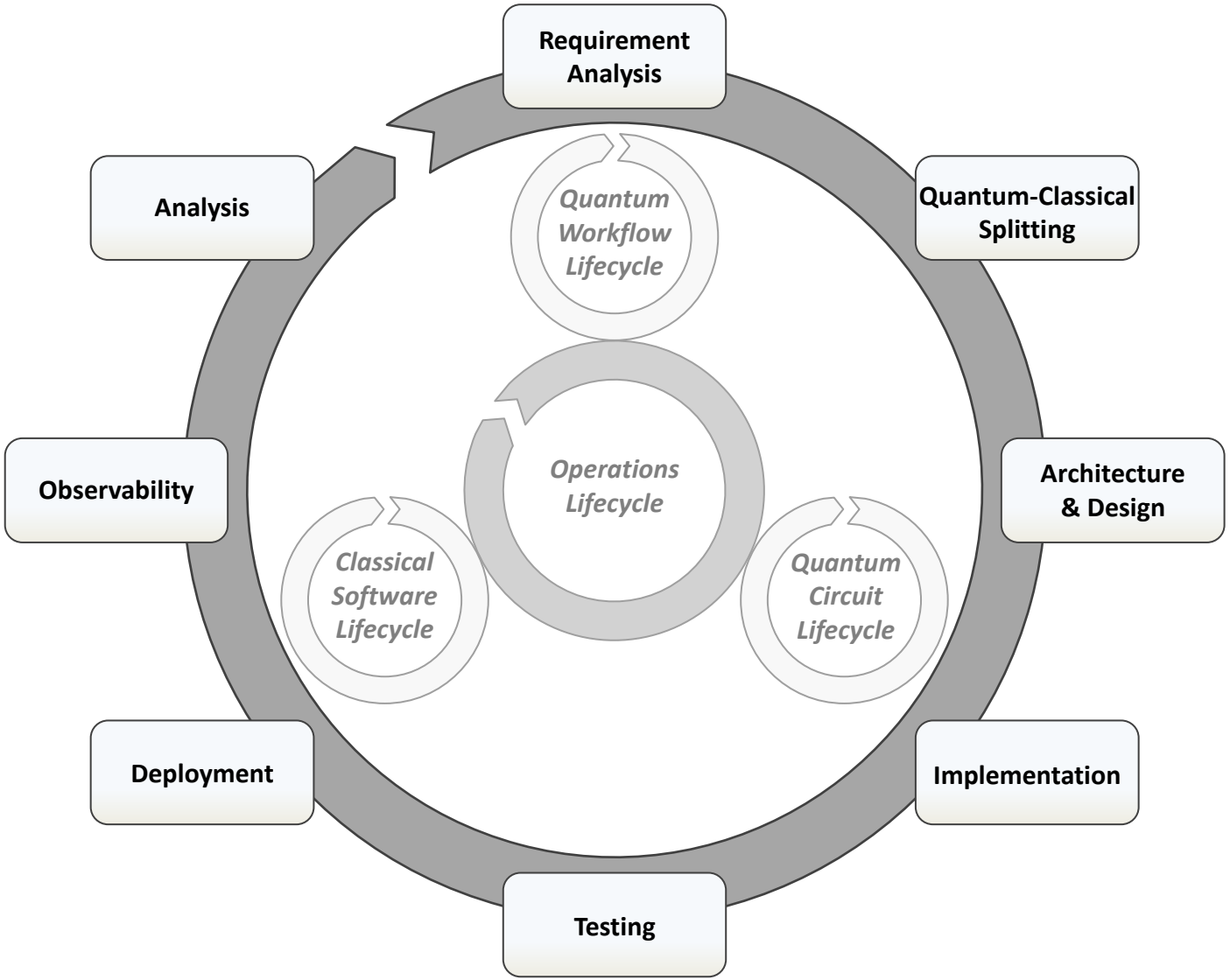
Will be discussed in Session 3

Will be discussed in Session 4

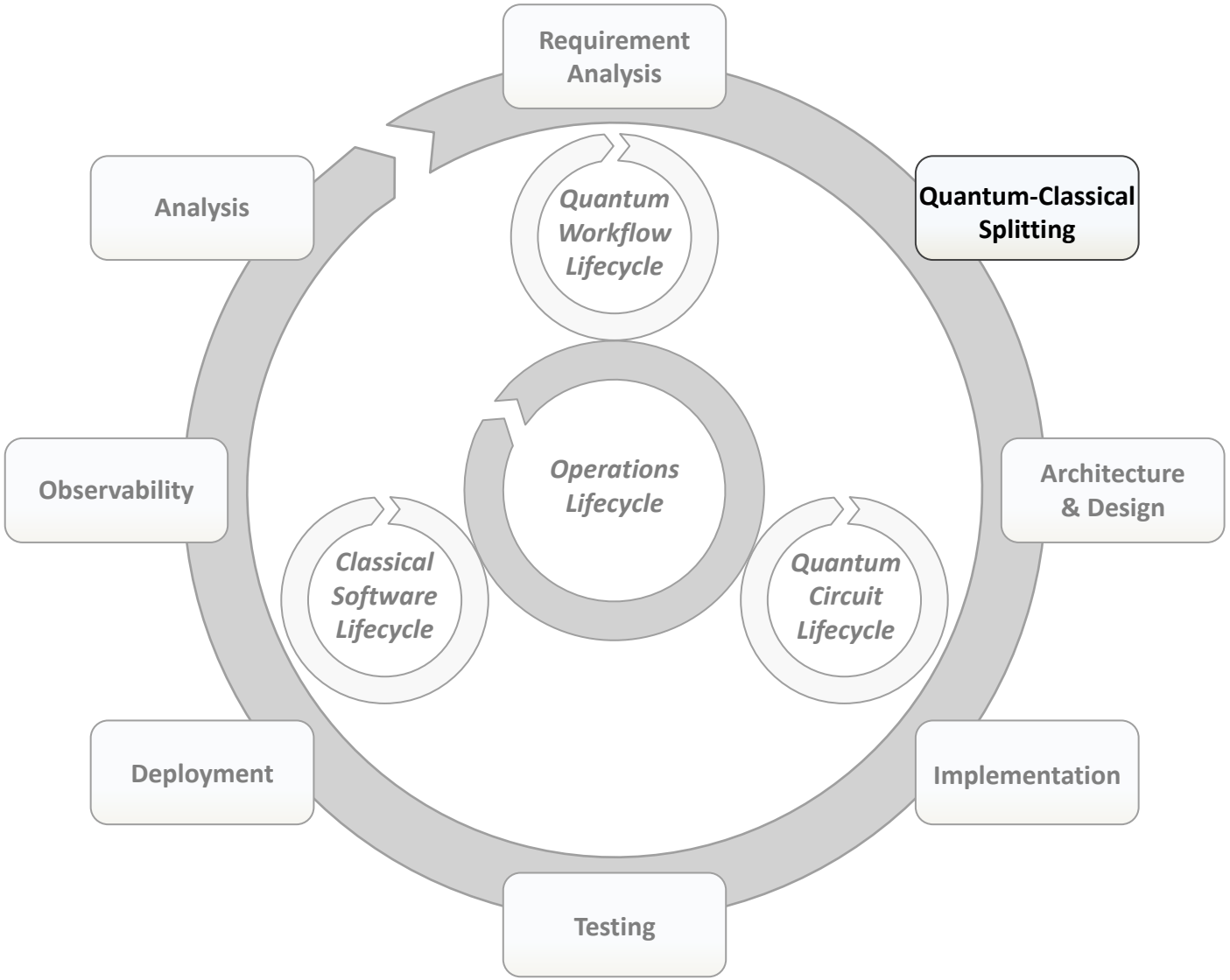
Usage of classical software lifecycles, e.g., the V-model or the prototype model

Enclosing Lifecycle

Enclosing Lifecycle

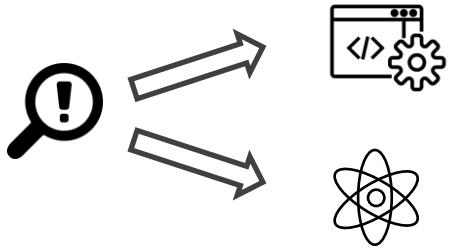


Enclosing Lifecycle



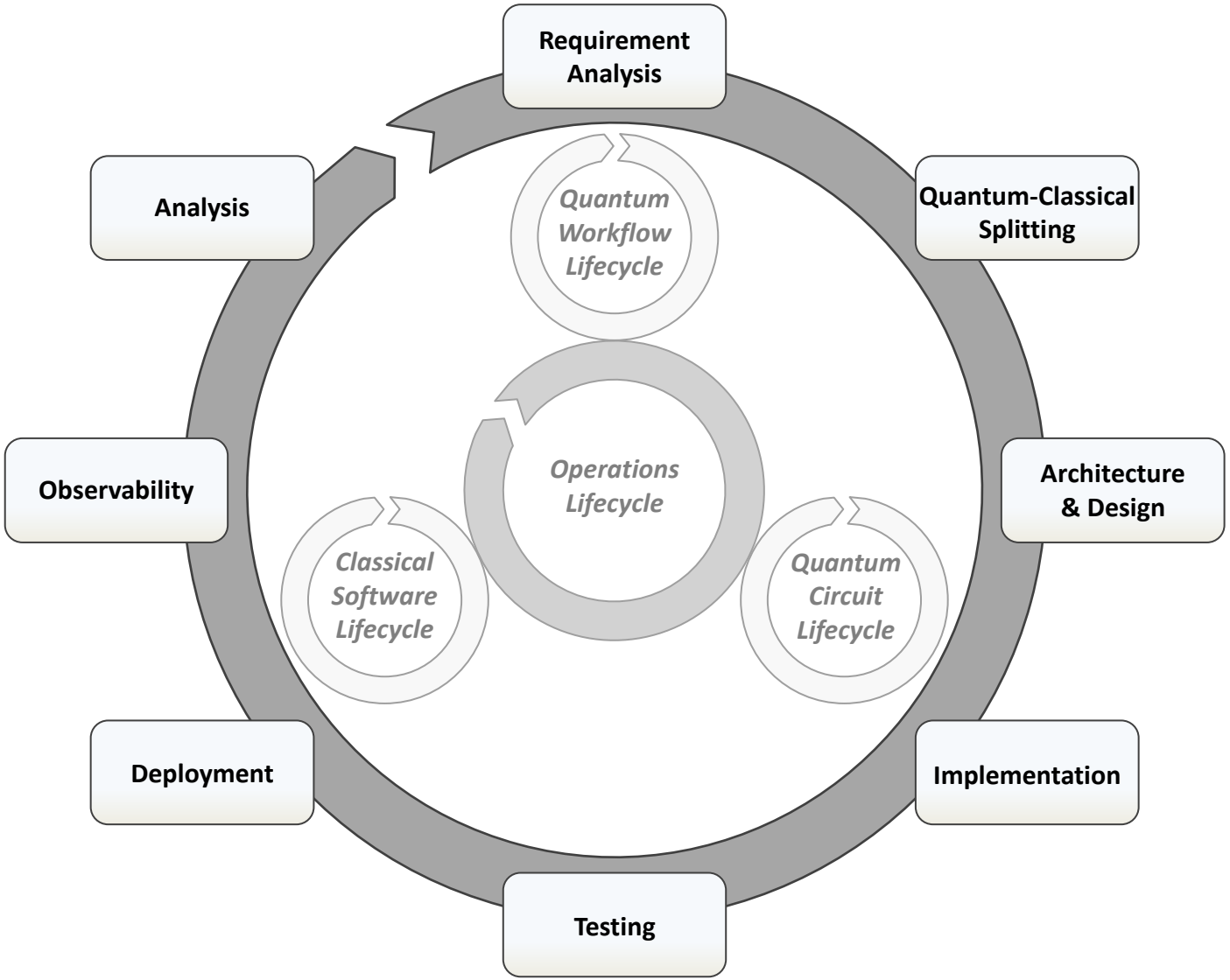
Quantum-Classical Splitting

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

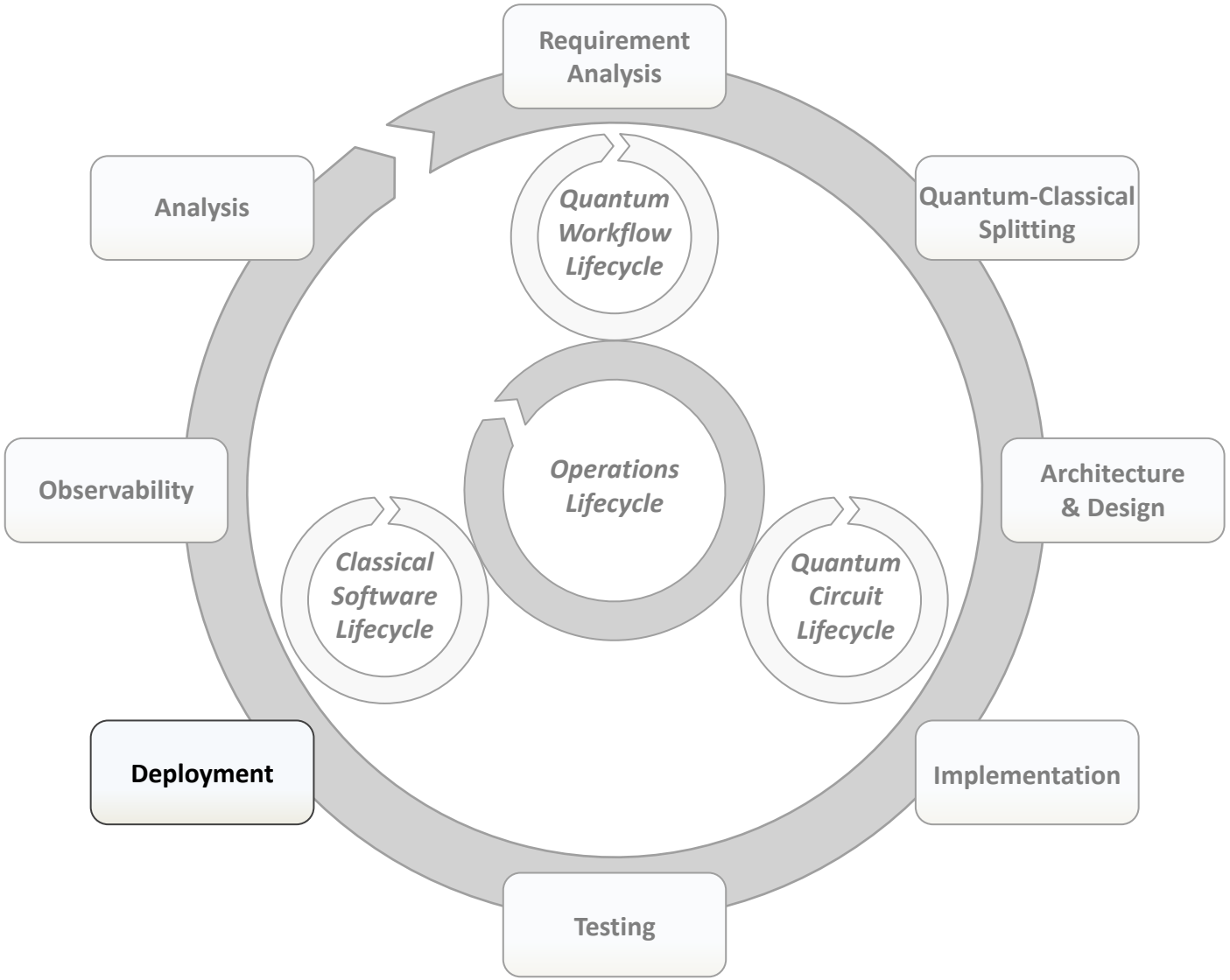


- Different techniques:
  - Manually by experts
  - Comparison of quantum algorithms with classical algorithms ➔ QHAna
  - Automated recommender (based on patterns, provenance, …)

Enclosing Lifecycle

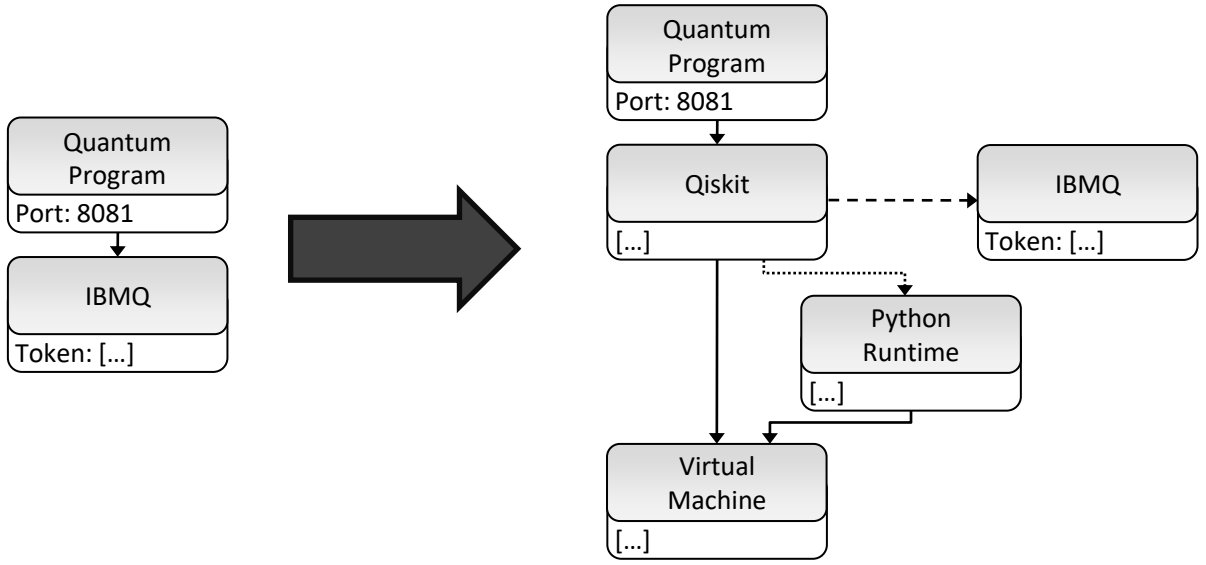


Enclosing Lifecycle

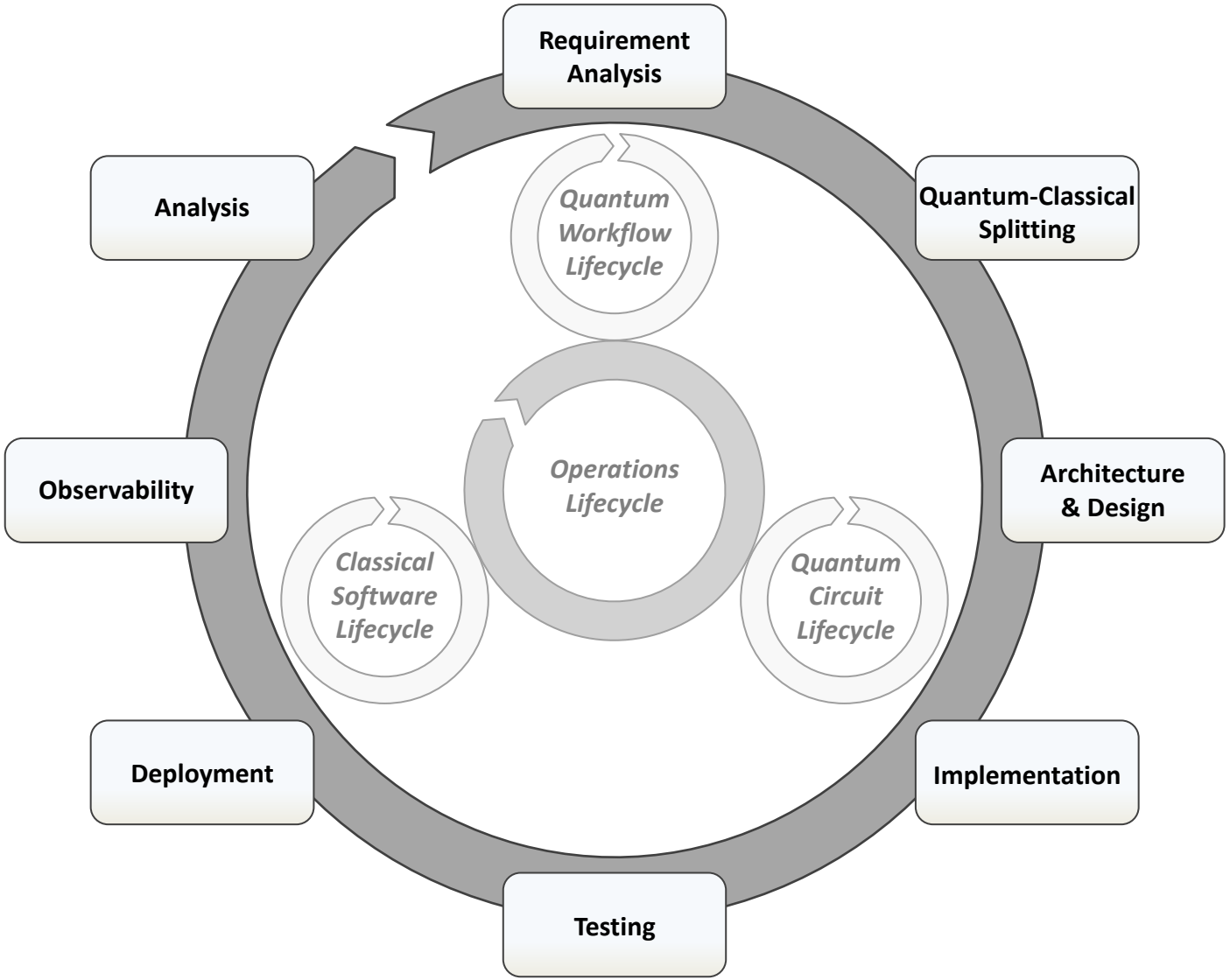


Topology Modeling – TOSCA4QC

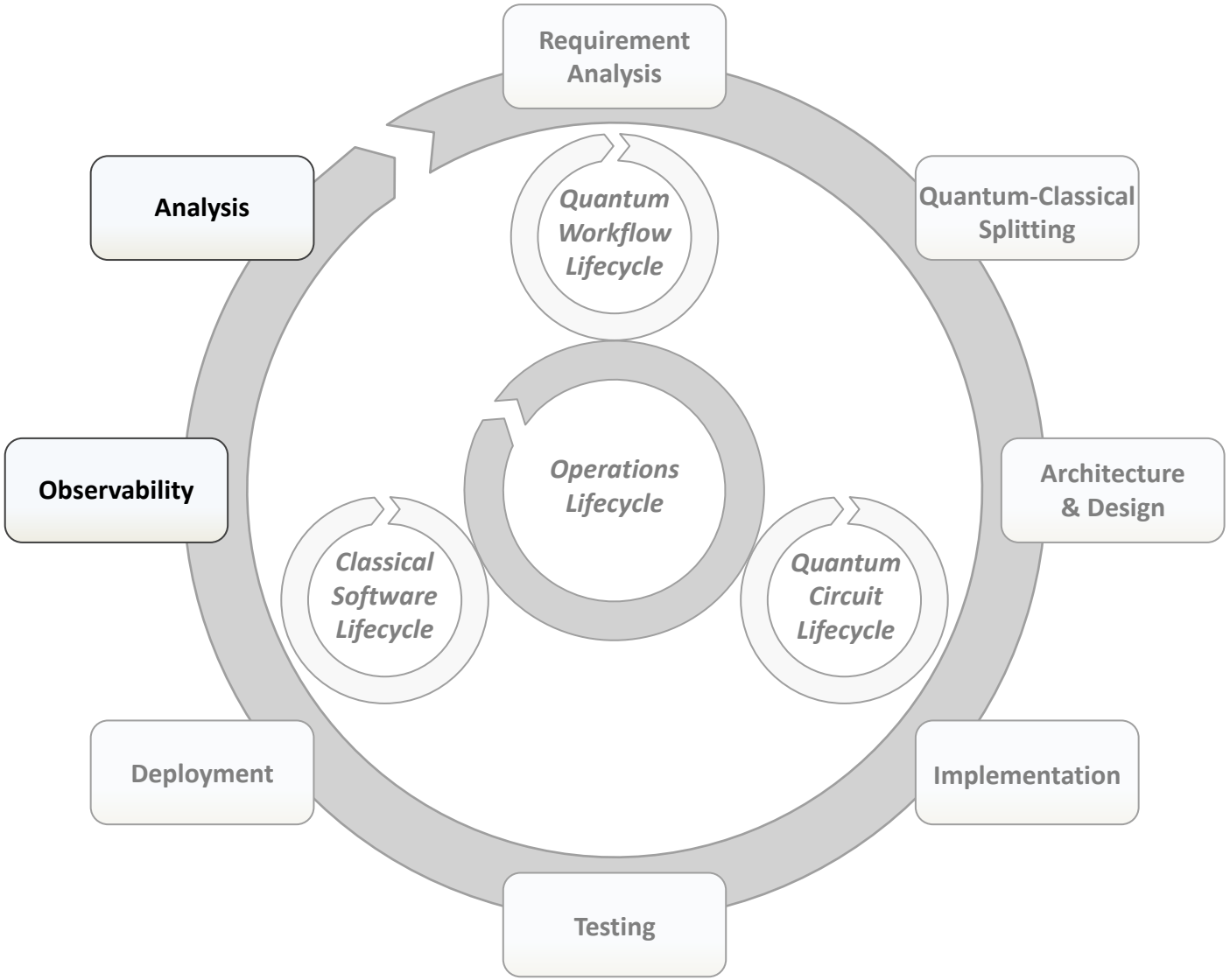
- Quantum programs are often deployed when they are invoked
- Common modeling principles do not apply
- *TOSCA4QC*:
  - Introcude two modeling styles for quantum applications
  - Automatic transformation between them



Enclosing Lifecycle



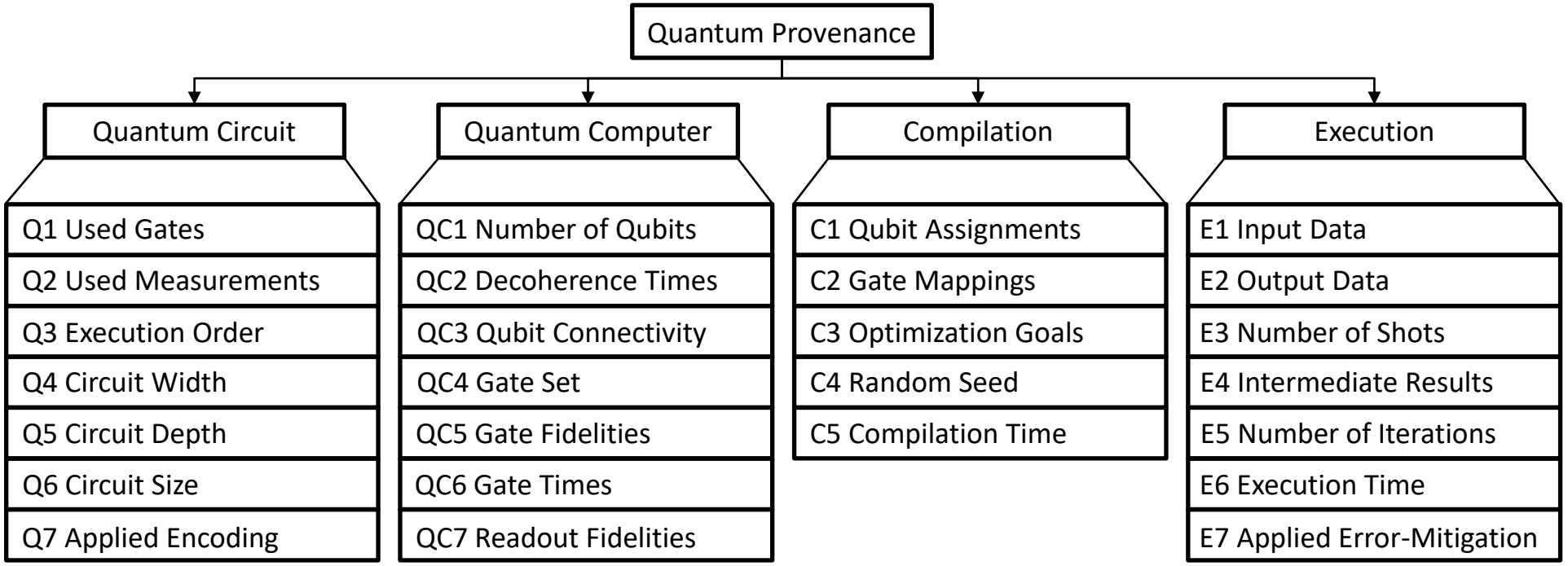
Enclosing Lifecycle



Quantum Provenance

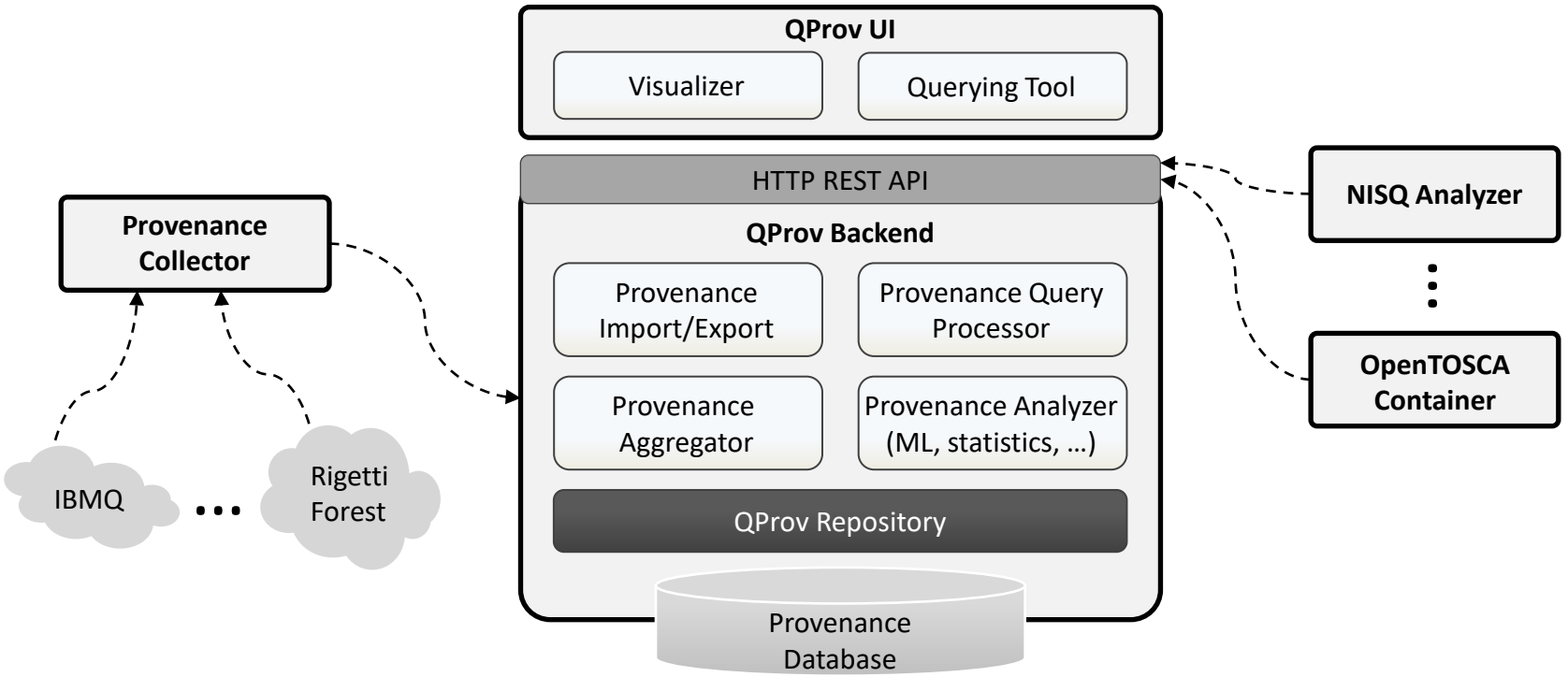
- Provenance
  - Information describing a process or computation
  - Goals: reproducibility, understandability, quality
- Especially important for quantum computing
  - Noisy devices (decoherence, gate errors, …)
  - Different realizations (trapped ions, superconducting, …)
- Example use cases:
  - Quantum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

Quantum Provenance Attributes



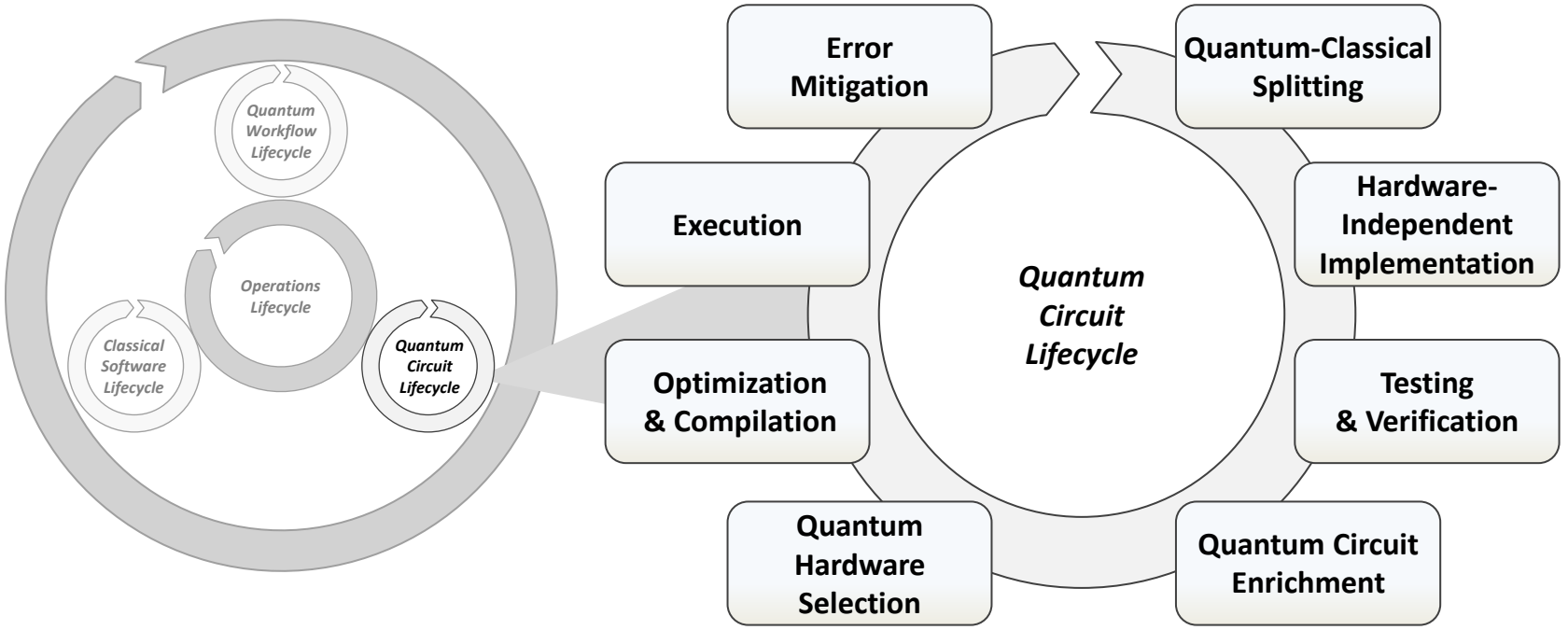
The QProv System

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

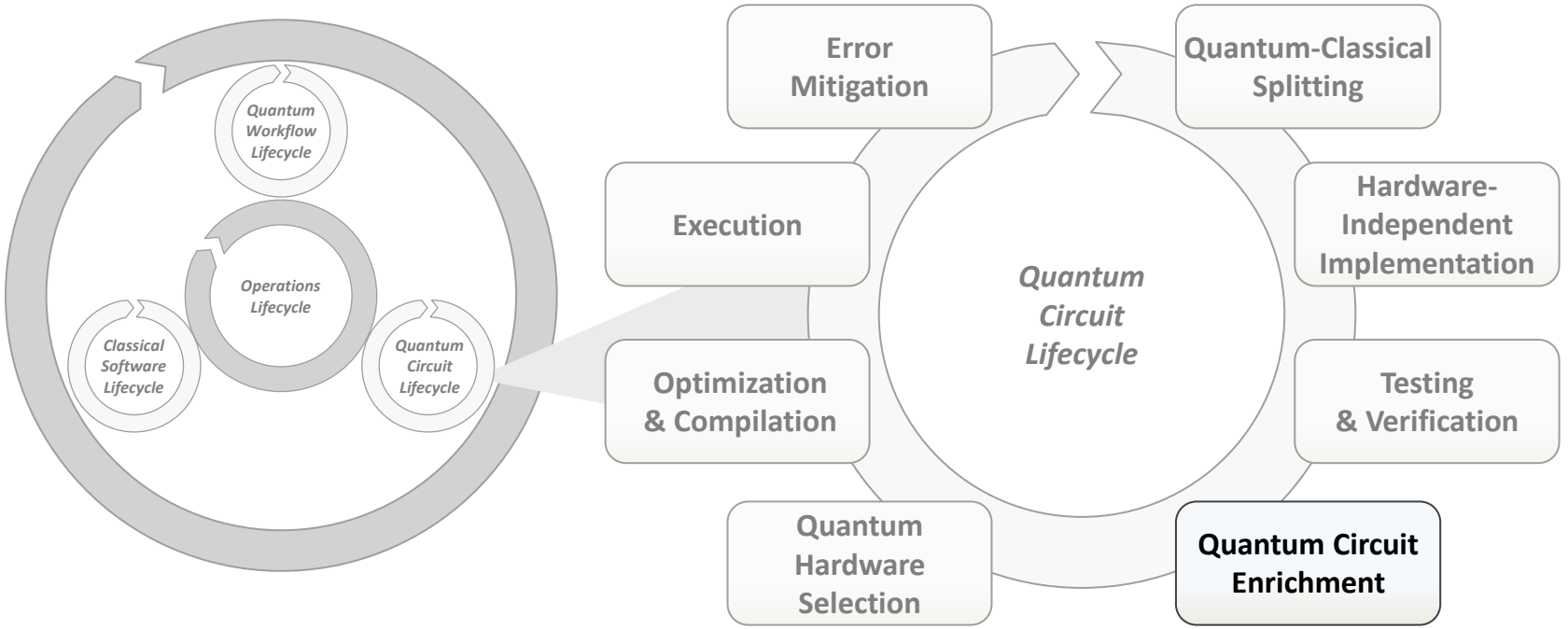


Quantum Circuit Lifecycle

Detailed View of the Quantum Circuit Lifecycle

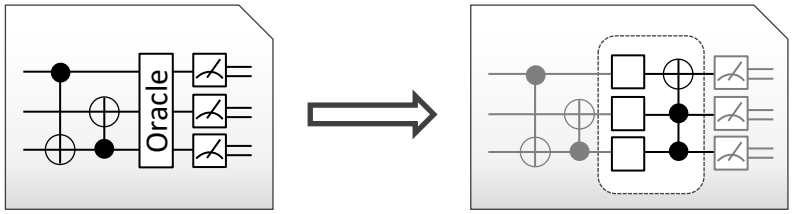


Detailed View of the Quantum Circuit Lifecycle

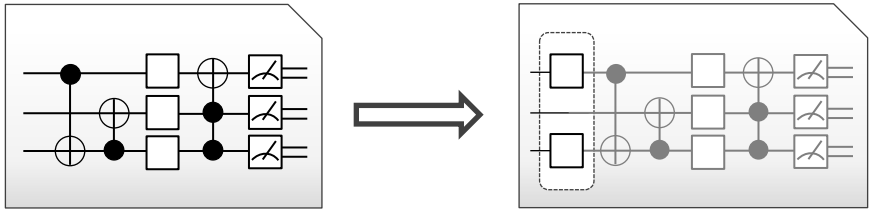


Quantum Circuit Enrichment

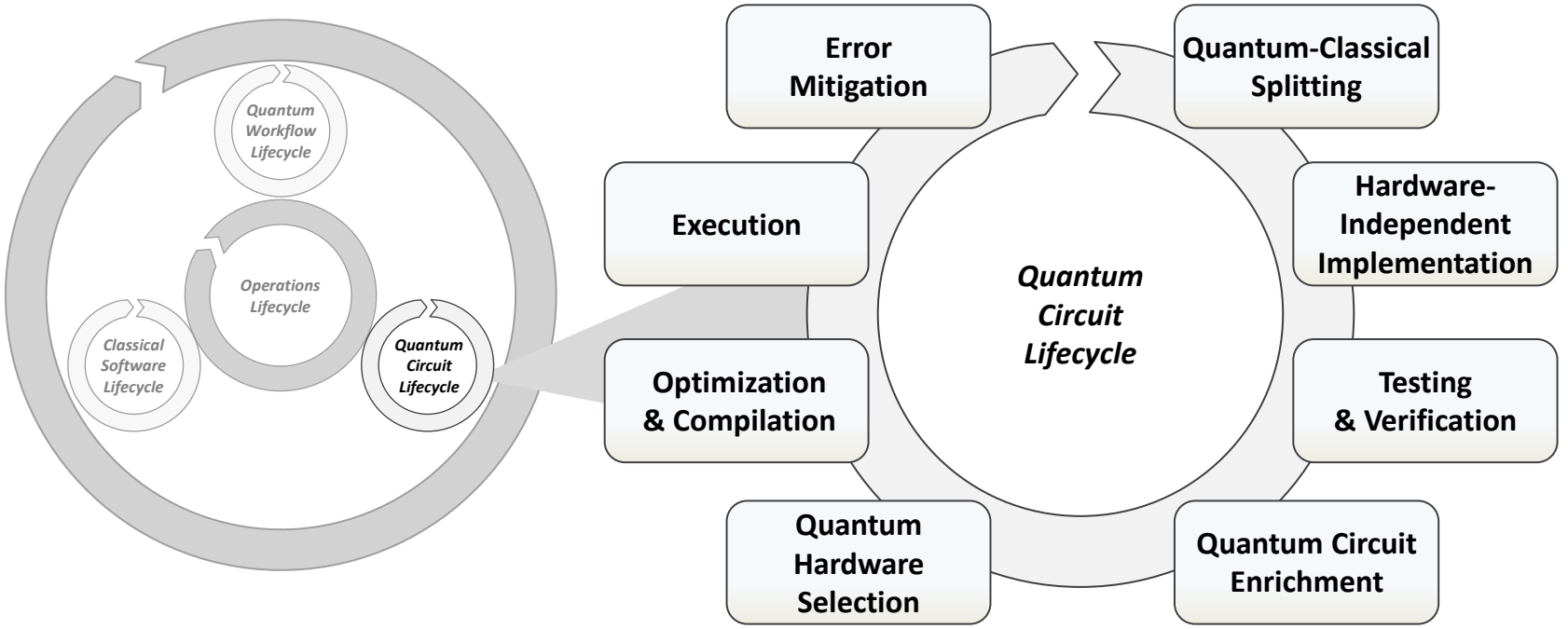
- Enrichment with details for a certain problem instance
- Oracle expansion:



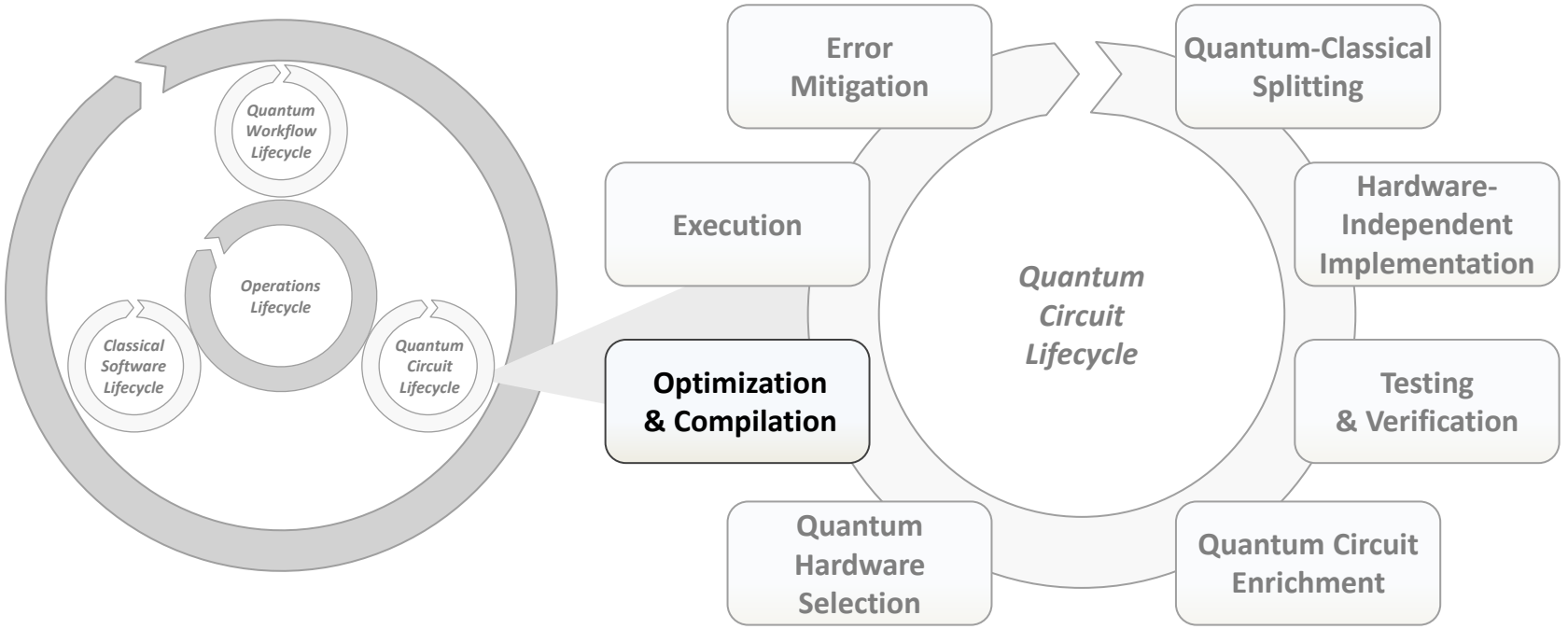
- Data preparation:
  - Adding an initialization circuit to the beginning of the original circuit
  - Different encodings: basis encoding, angle encoding, …



Detailed View of the Quantum Circuit Lifecycle

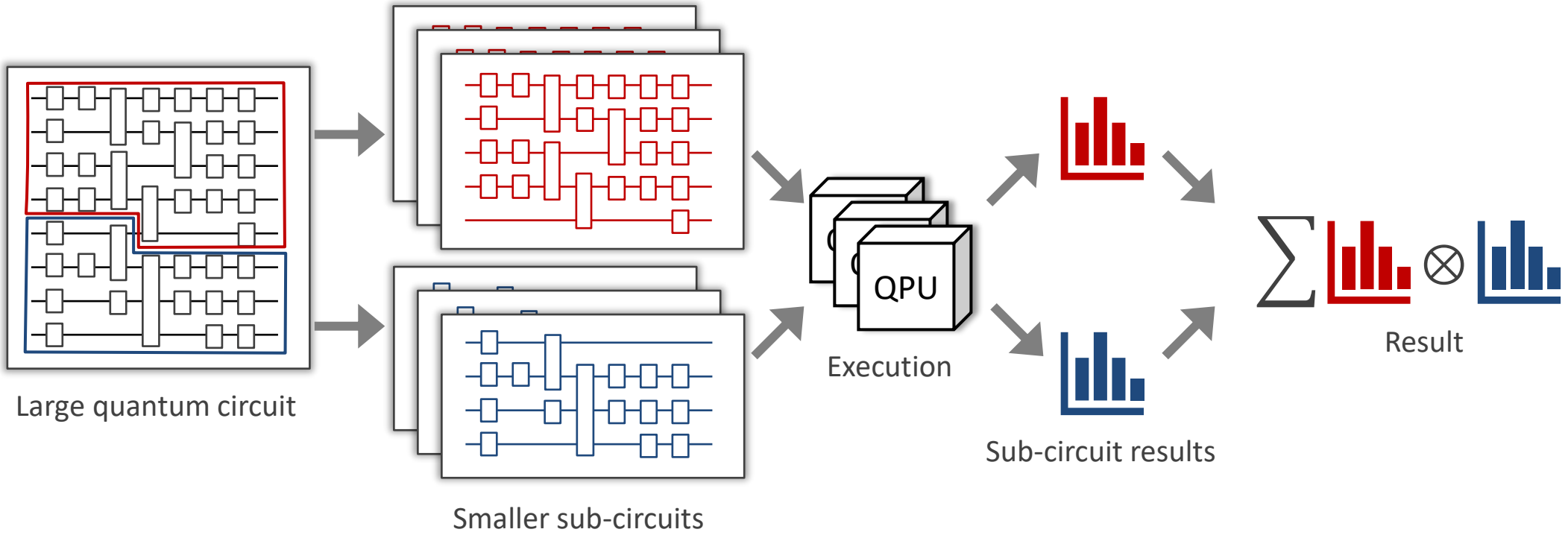


Detailed View of the Quantum Circuit Lifecycle



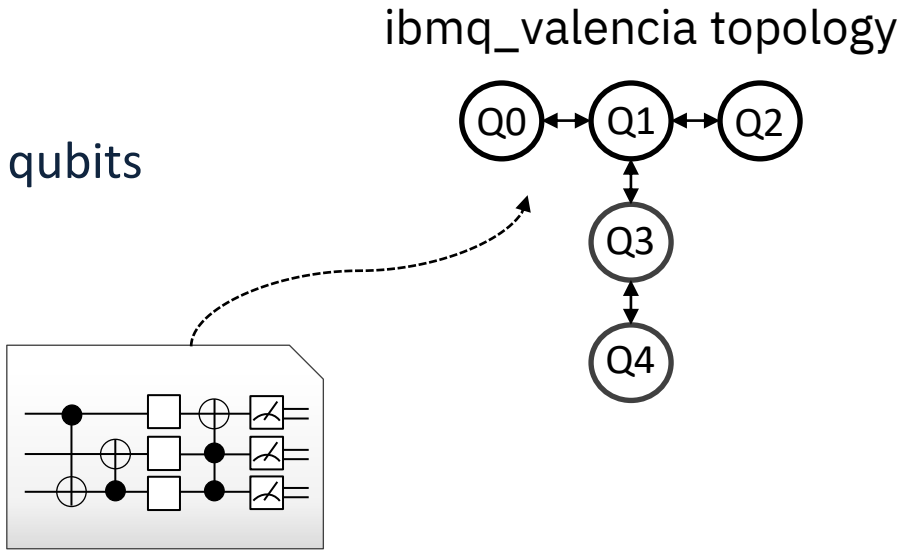
Optimization: Cutting Quantum Circuits

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

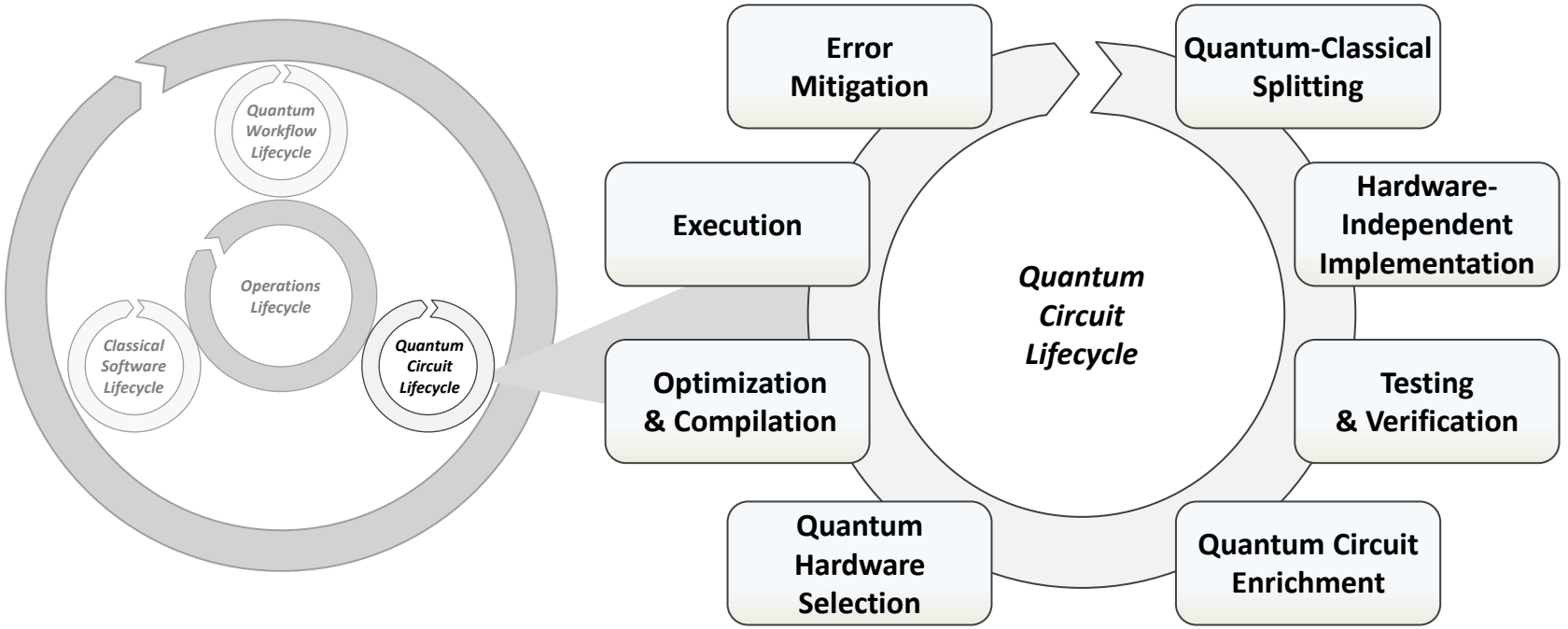


Compilation of Quantum Circuits

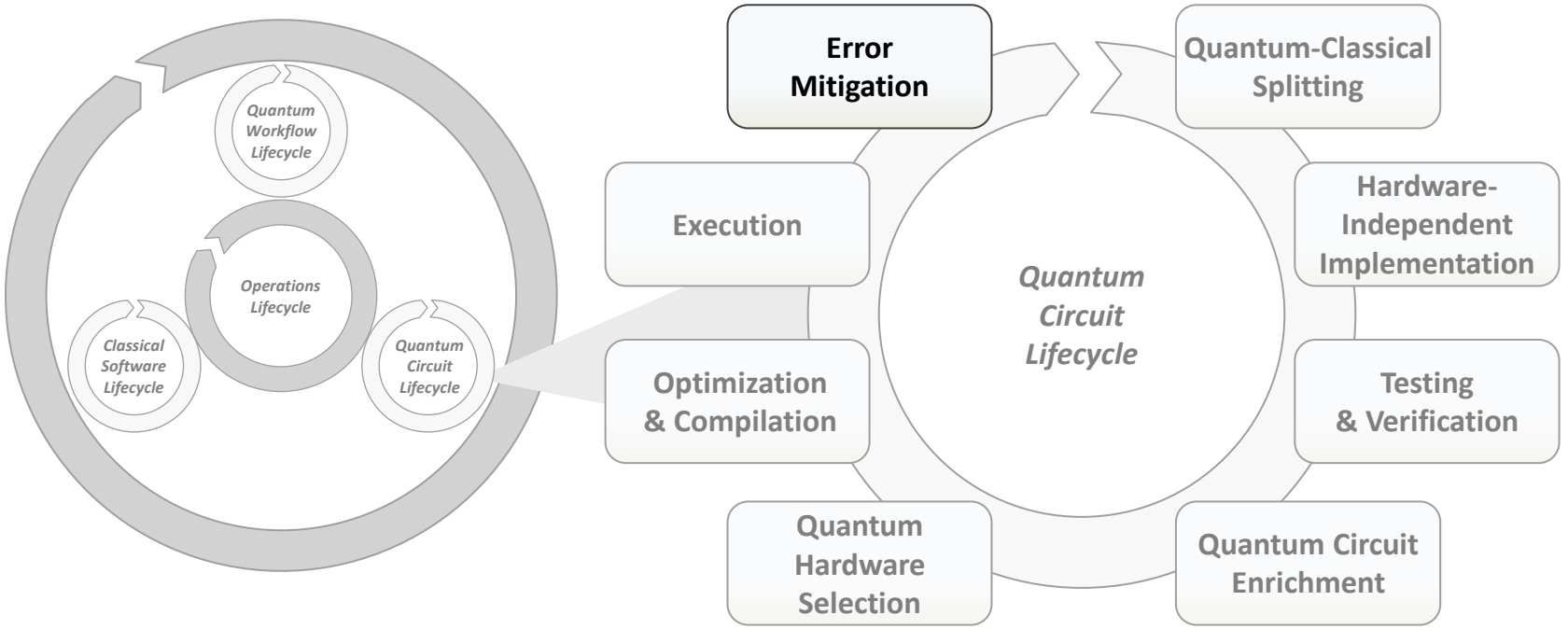
- Compilation to machine instructions required
- Replacement of not physically implemented gates
- Qubit allocation on the quantum computer
- Optimization based on:
  - Decoherence times of different qubits
  - Gate fidelities
  - Qubit connectivity



Detailed View of the Quantum Circuit Lifecycle

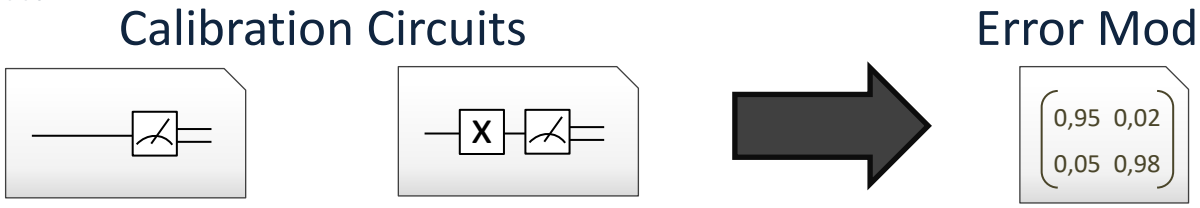


Detailed View of the Quantum Circuit Lifecycle

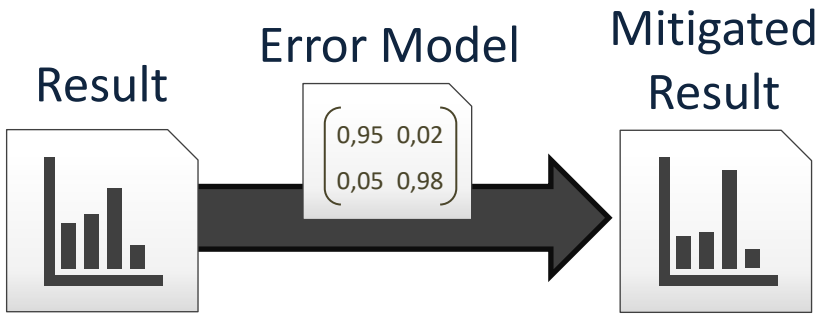


Error-Mitigation

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



  - Mitigating the result:



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 ☺