BHADALE GROUP OF COMPANIES
- IT AND REAL ESTATE

Aug 01 2022

Quantum Development platform Catalogue

Bhadale IT Pvt. Ltd. (https://www.bhadaleit.com/)

Bhadale Group of Companies

Bhadale Group of Companies consists of:

1. **Bhadale IT Pvt. Ltd** is an IT and Computer Engineering subsidiary company

   This division provides consultation in areas of cutting edge technologies, research outsourcing, and software consultation related to data center and related engineering practices

2. **Bhadale Engineering Pvt. Ltd** is a multi-engineering subsidiary company

Various divisions under this group provide design, & development of various IT and Engineering programs.

Bhadale Group has aggressive programs in place to serve the niche market. Below is related to cloud ecosystems

Bhadale Group IT Division, Quantum Services department

Below is a newer generic model for quantum based systems, and software development model. This offers various layers that work cohesively to ensure the classical and quantum parts work well, are backward compatible and allow future expansions

We offer generic platforms, frameworks, mathematical models, processes that ensure better usability across various domains, enterprises, and industries

We now describe one of these in the following sections

Platform details

Image courtesy (The Web, Quantum), no intention for copyright infringement

Introduction: Quantum projects need some base level templates and guidance that ensures work starts with an inventory asset enabling rapid development work. With regards to this concern, we have prepared this document that has loads of information, advises and sample template tool kit for your benefit.

State-of-the art software development methodologies cannot be used directly, and hence a redesign of the lifecycle is required to suite various project needs

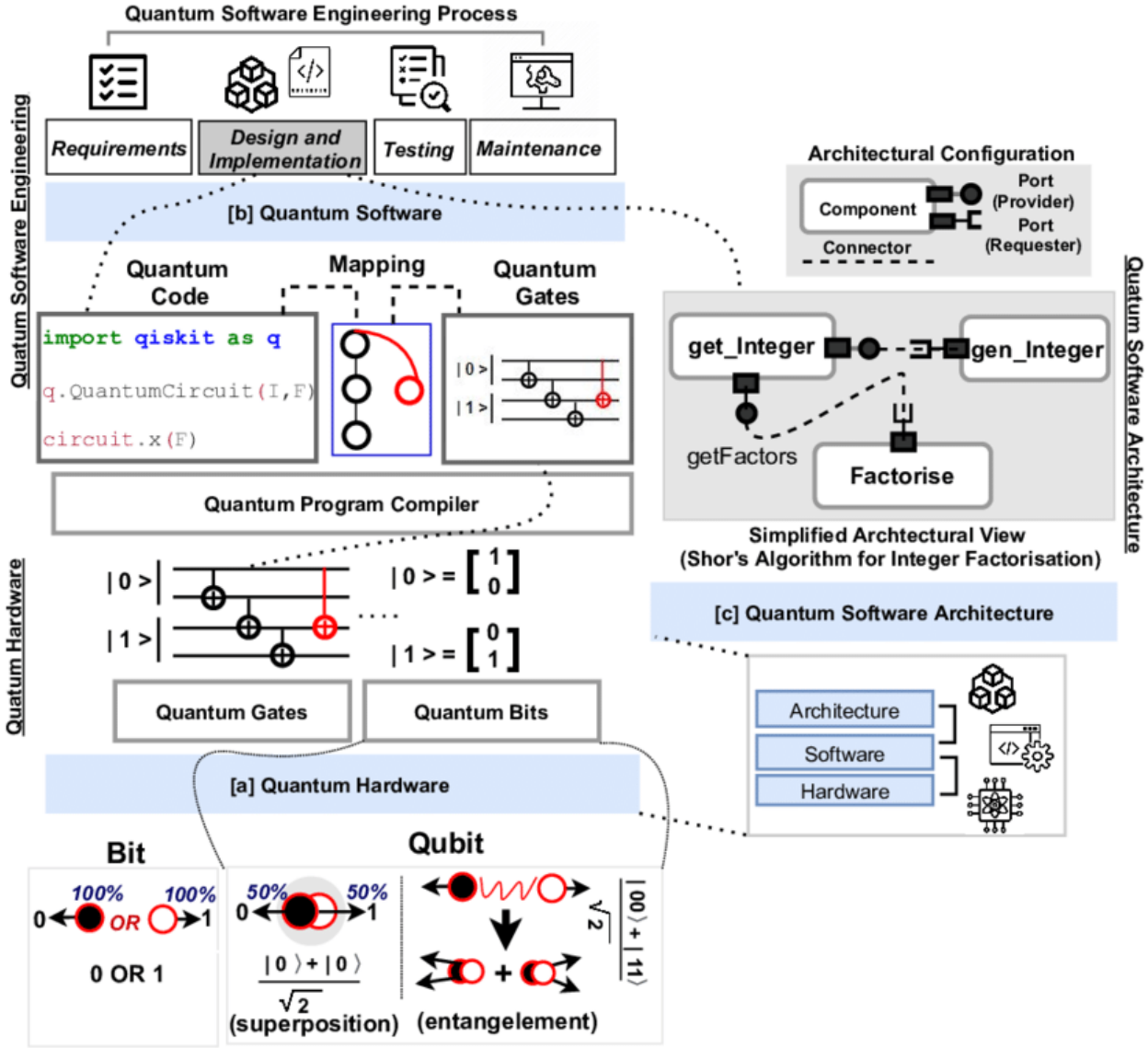

Figure 1: A Simplified View of Quantum Computing Systems ([a] Quantum Hardware, [b] Quantum Software, [c] Quantum Software Architecture) [1]

Many projects need some level of reengineering to adapt to the quantum paradigm. We have a process as shown in Figure 2

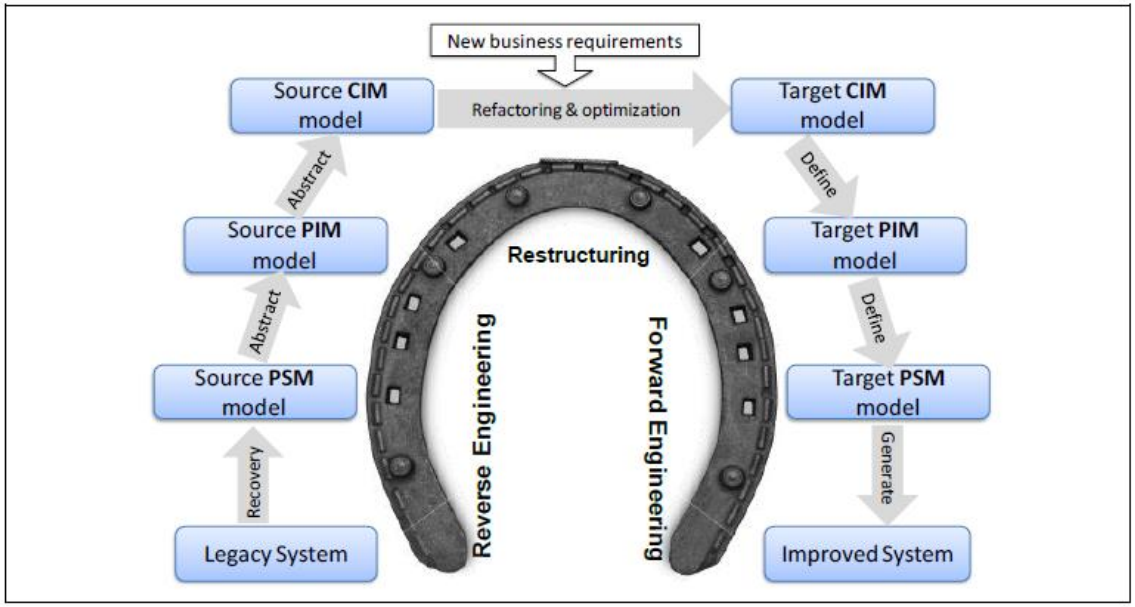

Figure 2: Horseshoe modernization model [2]

Reengineering of the existing systems are subjected to systematic rework using standard models, frameworks and processes. A logical view of the Figure 2 is shown in Figure 3 that shows how KDM and UML based models offer an easier visual path to migration from legacy systems

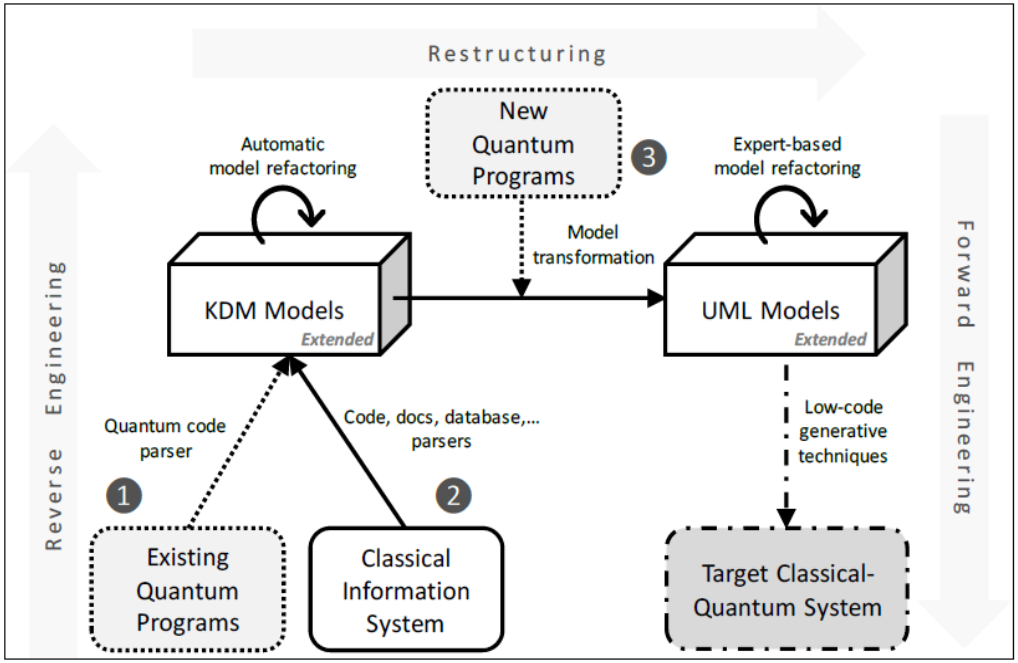

Figure 3 Quantum reengineering process [3]

The quantum development stack that we offer is made of various layers as shown in Fig 4

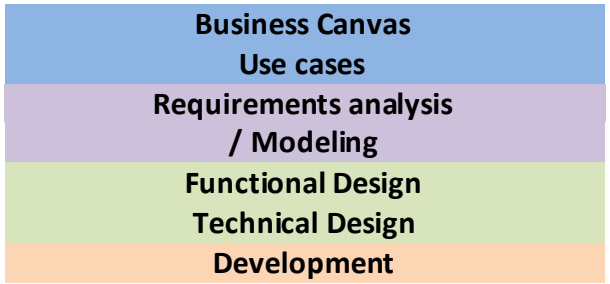

Figure 4 Quantum development stack

Each layer offers its unique feature and benefits. These are all generic and allow for better modularization of various modules, sub modules, system elements and software components. We will discuss these layers in brief

Business Canvas

Business Canvas is the starting top level enterprise assets that is developed in close coordination with business owners, stakeholders and a small segment of the market namely buyers, product reviewers, sales and marketing team etc

Key product / service features are also noted that allows the owners to invest safely that has some level of assured returns

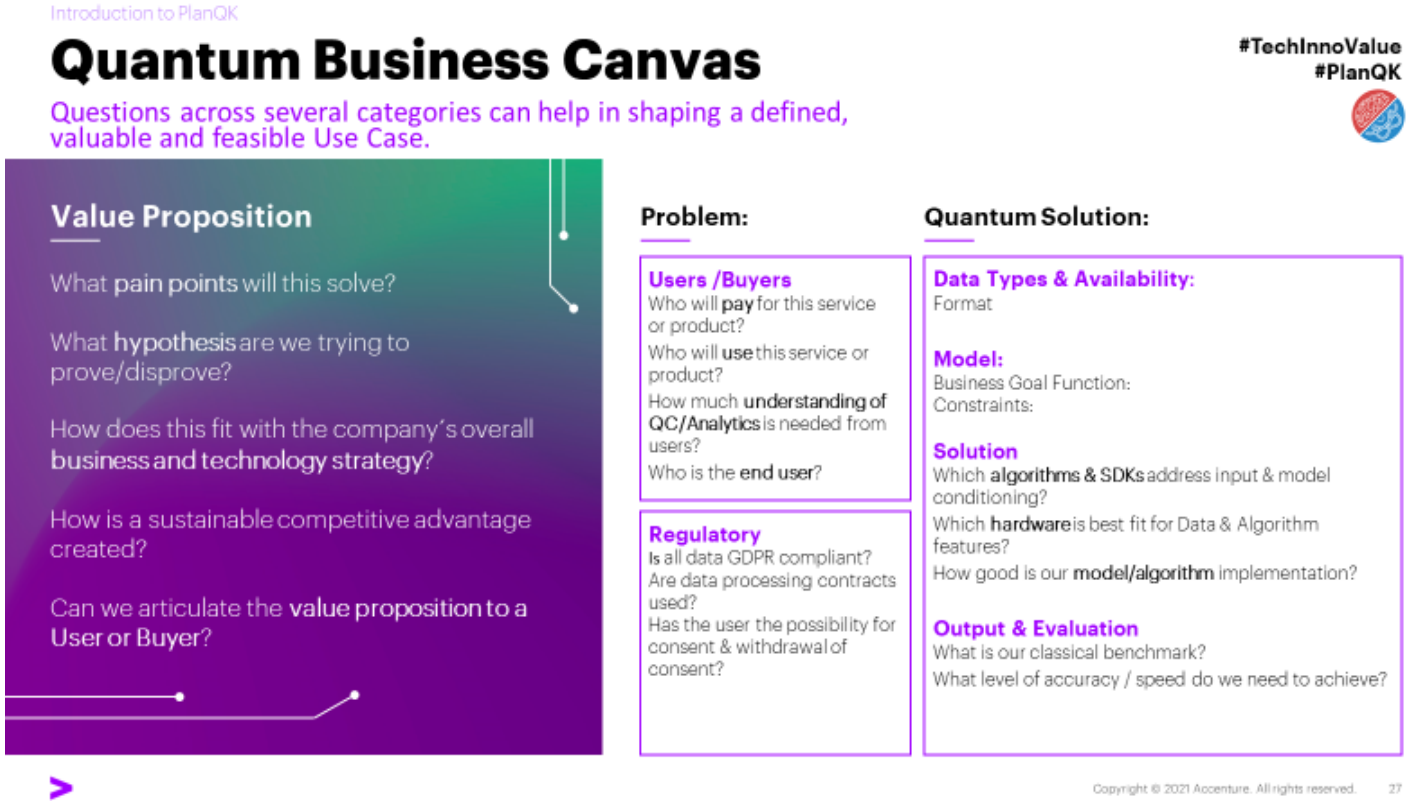

Figure 5: Business Canvas [4]

The business canvas offers a high level view of the problem that needs attention and how a solution will offer value to the problem. A high level mapping of the problem to the solution domain helps in effectively addressing various problems, concerns of all types be it stakeholder, technology or business related

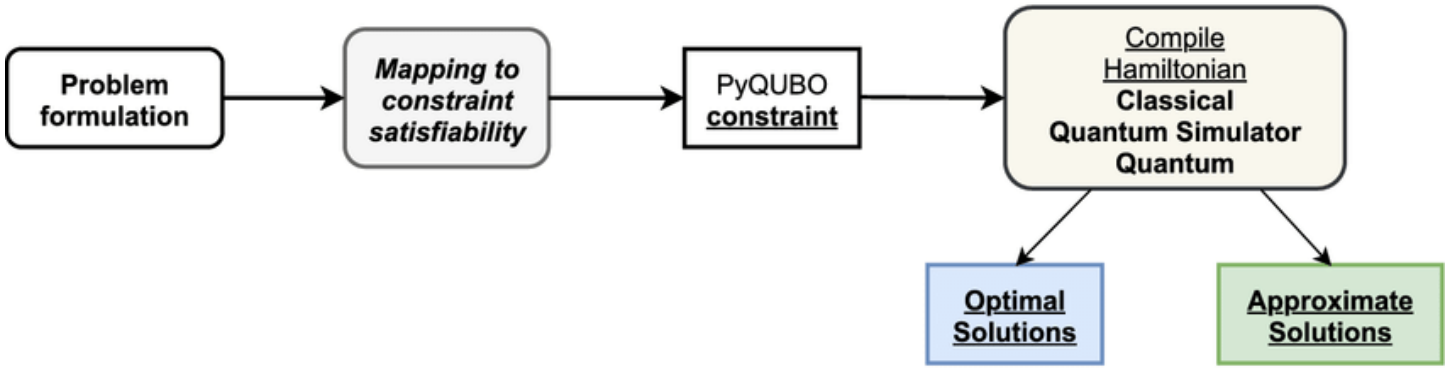

Figure 6: Steps for the problem to solution mapping [5]

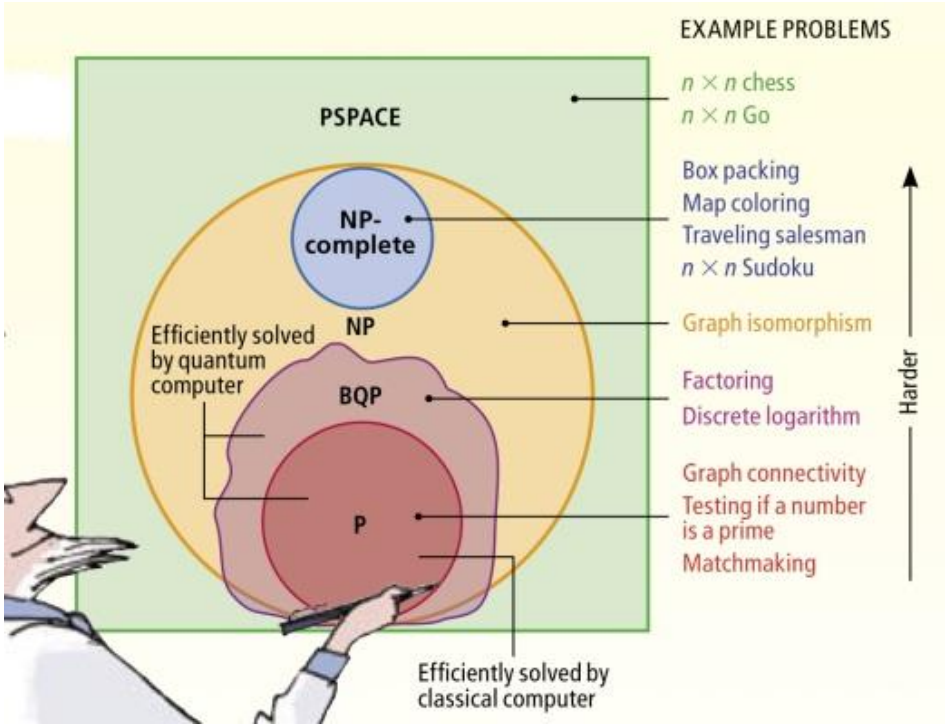

Figure 7: Problem Complexity [6]

The business canvas offers a technology-agnostic solution platform for the given quantum project and the problem being addressed. It may be a system level, software level or business level problem that is being addressed.

Use Cases

Use case being the second enterprise asset being developed after Business canvas modeling. This elaborates the business needs into usage based use cases, scenarios, actors and the data that is being fed, transformed and generated. Below is the high level matrix to hold the key features of the use case. Use cases are based on the needs of the industry, domain constraints and project application, software and system needs

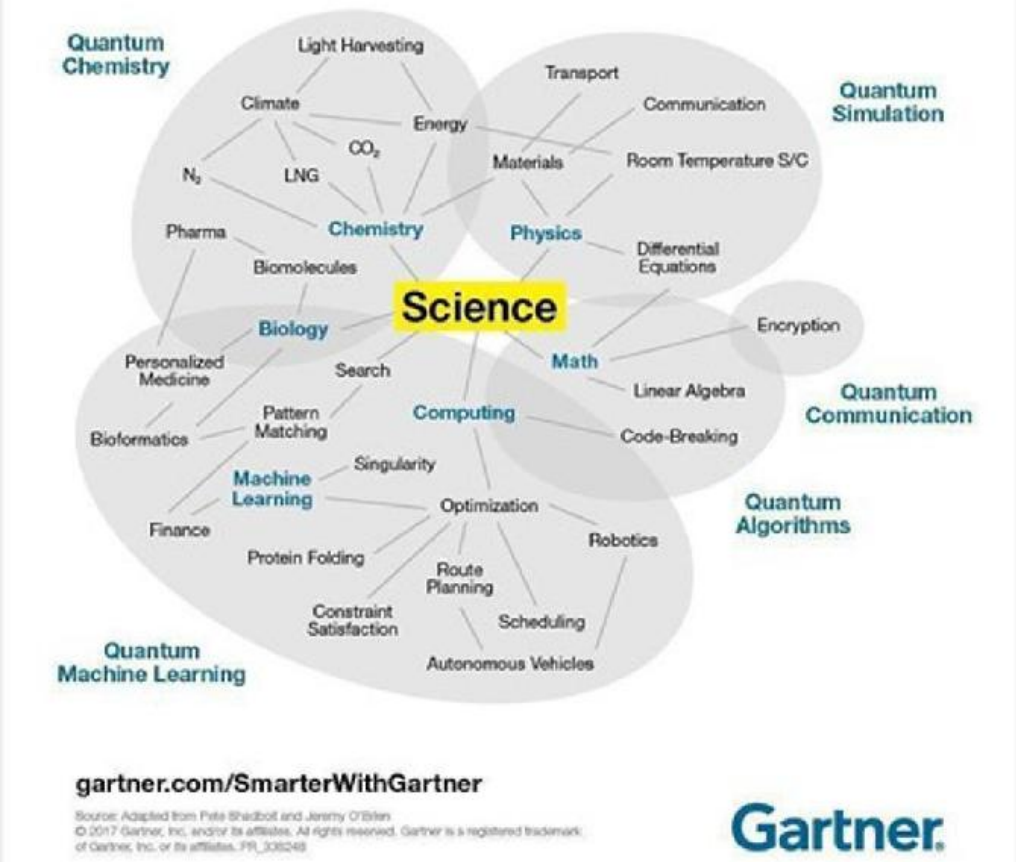

Figure 8: Quantum use cases [7]

Once the required scoping of the domain is available, then the finer details are developed. Sample templates are shown

Use Case Name	Sub Industry class	Function	Problem Domain	User	Business Challenge	Value Proposition	QC Solution Approach	Model	Algorithm	Hardware	QC Limitations	Potential Impact

Table 1: Use case template

Using this matrix, and languages and tools like UML, SysML, Papyrus, Visual Paradigm, and various diagrams are designed.

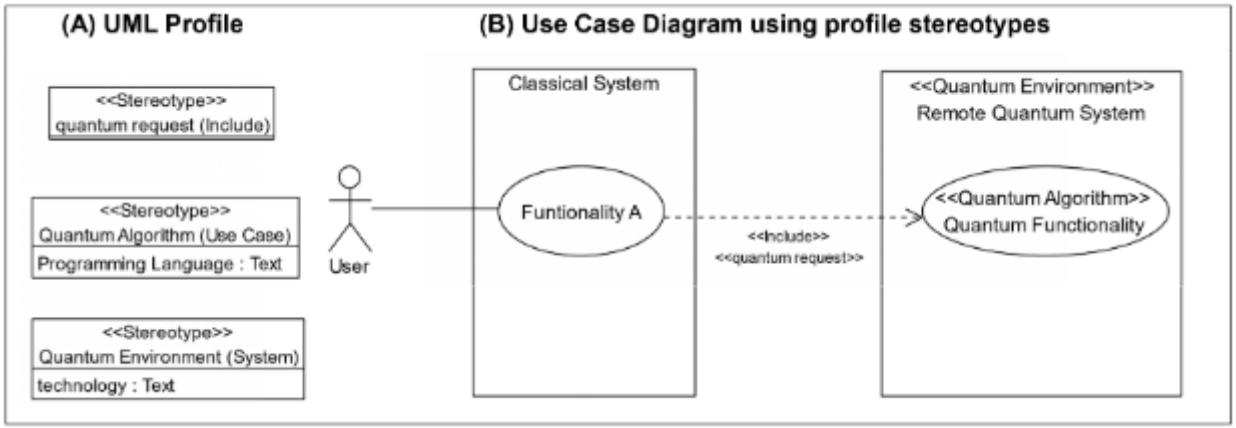

Figure 9: Quantum UML profile with use case diagrams [8]

Requirements Analysis / Modeling

Quantum requirements analysis provides the analytical support for the Use cases, the involved parties, assets, roles, functions and data that is required to have a scientific platform and base that will allow and ensure the need of scientific evidence, repeatability, non-random results, well defined returns and assured to generate same type of value and quality.

Requirements analysis generates a variety of constraints, designs, interfaces, communications, high level functional and technical roadmaps, required managerial methods, state-of-the art and research technologies that can aid in the realization.

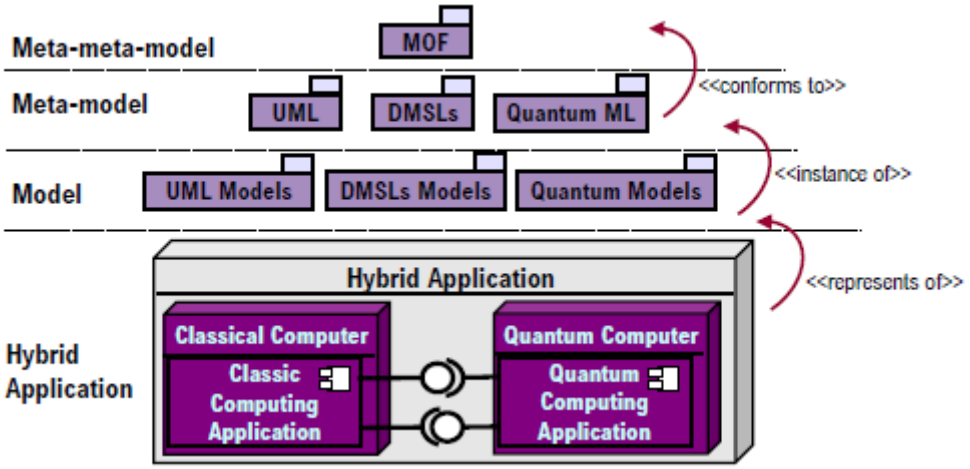

Figure 10: Meta-meta model [9]

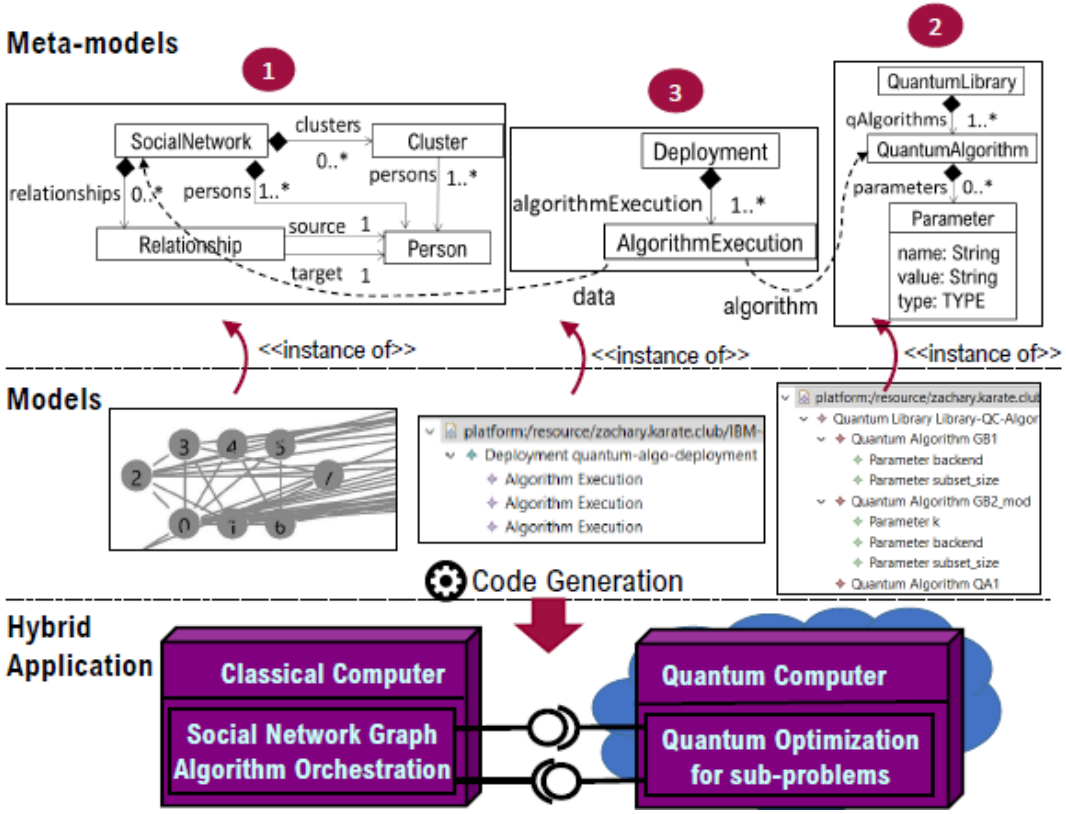

Figure 11: Metaa-models [9]

Layer of Abstraction		Pillar: Requirements	Pillar: Behavior	Pillar: Structure	Pillar: Parametrics
Problem	Black Box	Stakeholder Needs	Use Cases	System Context	Measurements of Effectiveness
Problem	White Box	System Requirements	Functional Analysis	Logical Subsystems Communication	MoEs for Subsystems
Solution		Component Requirements	Component Behavior	Component Assembly	Component Parameters

Figure 12: MBSE Grid framework [10]

Layer of Abstraction		Pillar: Requirements	Pillar: Behavior	Pillar: Structure	Pillar: Parametrics
Problem	Black box	**Stakeholder Needs:** • Requirements diagram • Requirements table	**Use Cases:** 1. Use Case diagram 2. Activity diagram	**System Context:** • Internal block diagram	**Measurements of Effectiveness:** • Block definition diagram
Problem	White box	**System Requirements:** • Requirements diagram • Requirements table	**Functional Analysis:** • Activity diagram	**Logical Subsystems Communication:** 1. Block definition diagram 2. Internal block diagram	**MoEs for Subsystems:** • Block definition diagram
Solution		**Component Requirements:** • Requirements diagram • Requirements table	**Component Behavior:** • State machine diagram • Activity diagram • Sequence diagram	**Component Structure:** 1. Block definition diagram 2. Internal block diagram	**Component Parameters:** • Parametric diagram

Figure 13: MBSE Grid mapping to SysML [10]

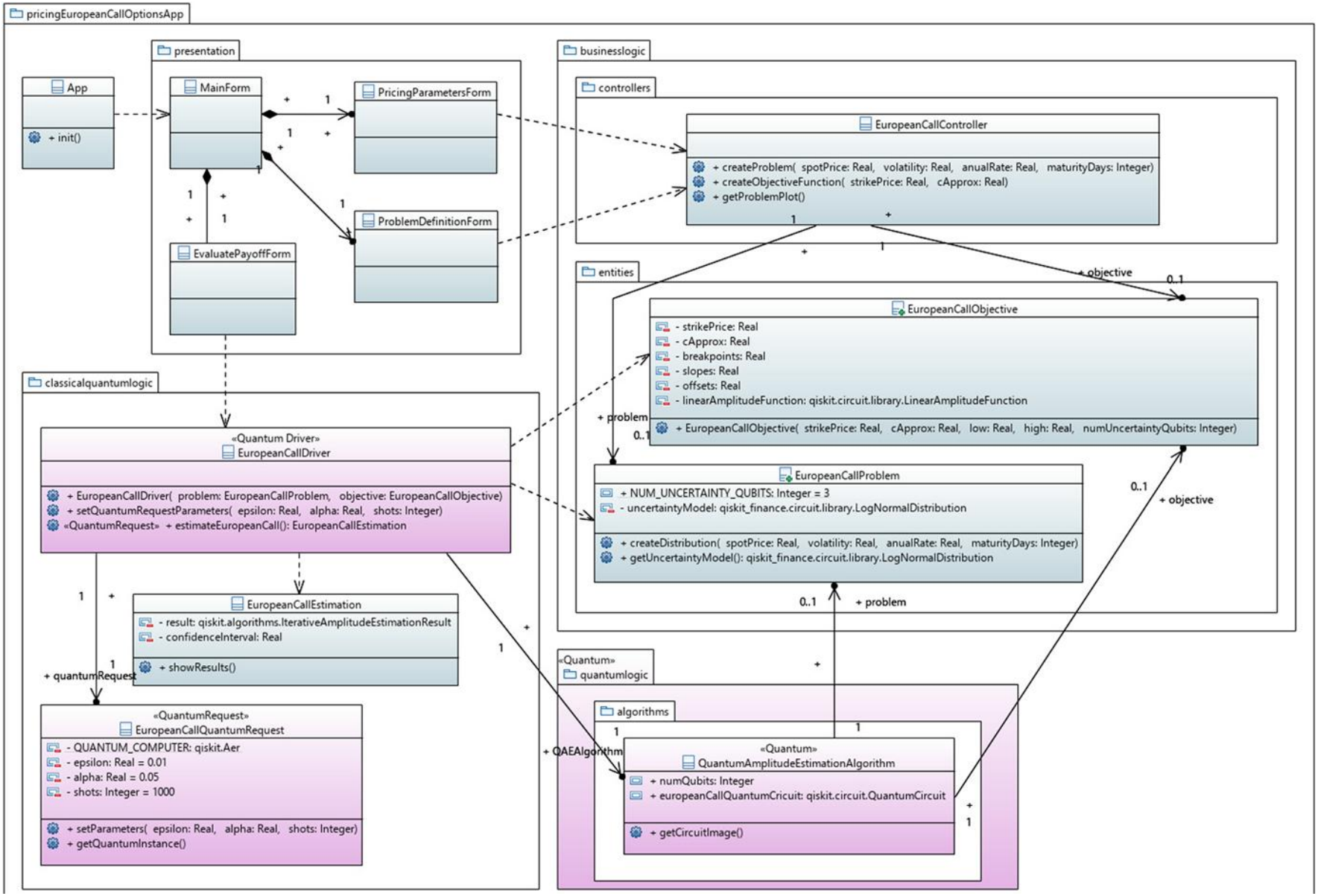

Figure 14: Class diagram with Quantum UML profile for a pricing call option [11]

Functional Design

Functional Design is based on the requirements that have been analyzed and a mathematical model has been designed that allows the foundational need to design the functional design that allows for various types of functional and non-functional attributes to be developed by a product or a service.

This document offers functional & logical views, data sets used, I/O, the model that allows the proper functioning of the system, integration, and communication channels used to accomplish the desired features

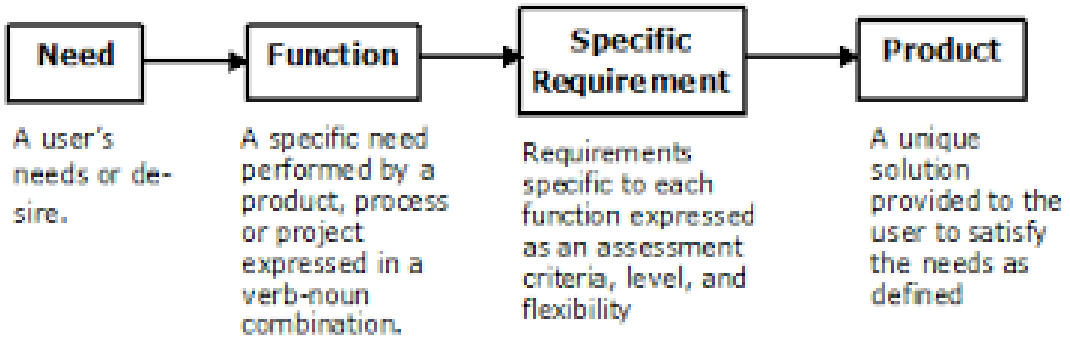

Figure 15: Function map [12]

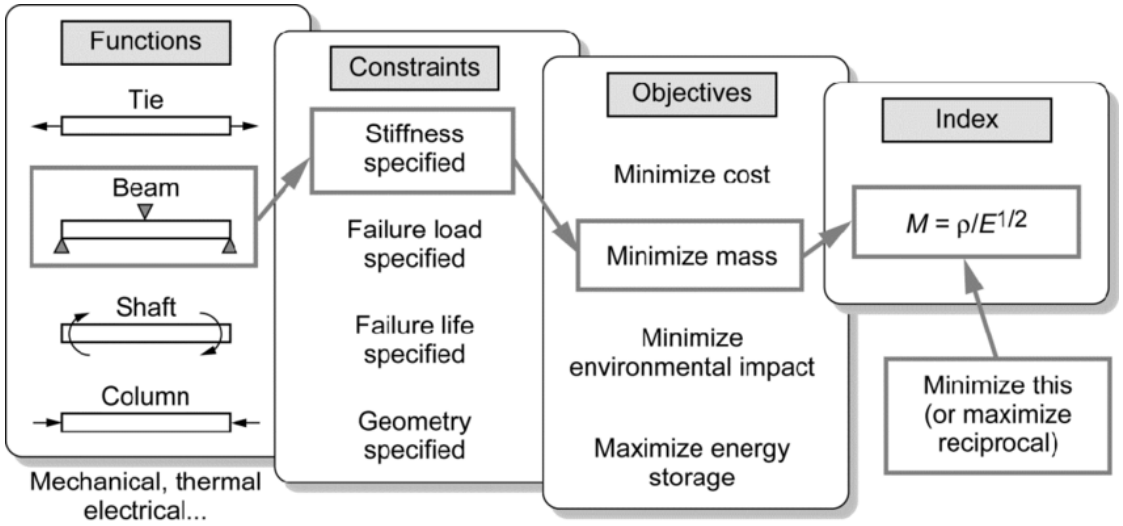

Figure 16: Function definition [13]

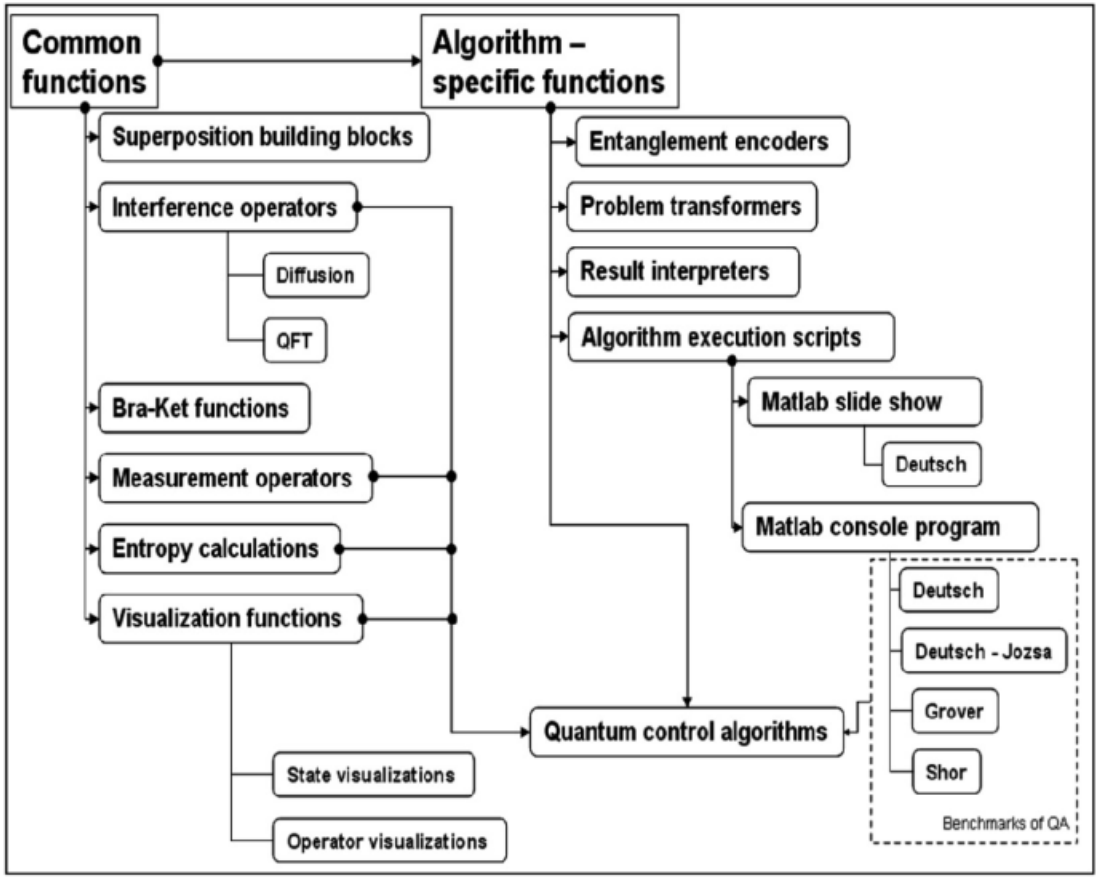

Figure 17: Quantum functions class [14]

Quantum projects need to satisfy various quantum features like superposition, entanglement. These are patterns that fulfill these few of the repeatable patterns are

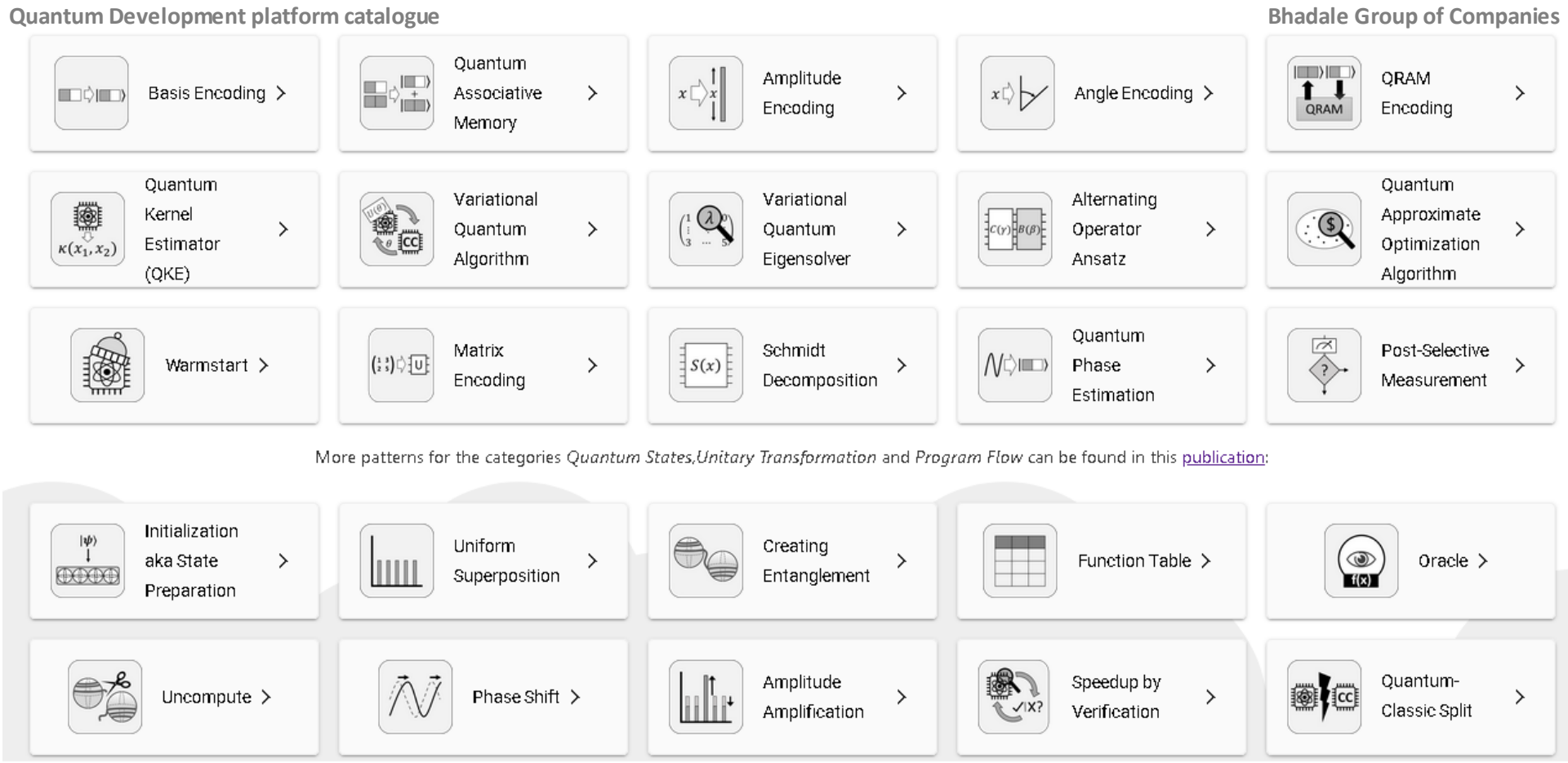

Figure 18: Quantum computing patterns [15]

Here is a good collection of quantum patterns. There is a need of further patterns for NISQ-era hybrid software and systems, enabling development of reusable enterprise and domain specific packages.

For example, Shor's algorithm needs QFT and Period finding code fragments from classical and quantum paradigms. To realize, we need a hybrid pattern that may have separate worker threads, kernel and user plane interrupt service routines, low level communication using IPC across different OS, allow debugging and plugged into IDE for DevOps, tool chains, & release trains etc

Using the standard functional building blocks, algorithms, software data structures templates, all these offer various functional capabilities. These are identified and mapped to the logical needs for further elaboration in technical design

Various stack functional layers are developed that offer the required optimized solution for the product.

Technical Design

Technical Design is based on the functional design that allows the foundational need to design the technical design for various types of technologies mapped to the functional needs of the product or a service.

This section offers technological views, data sets used, I/O, the technology stack that allows the proper functioning of the system, integration, and communication channels used to accomplish the desired features and requirements. An example of the technical stack is

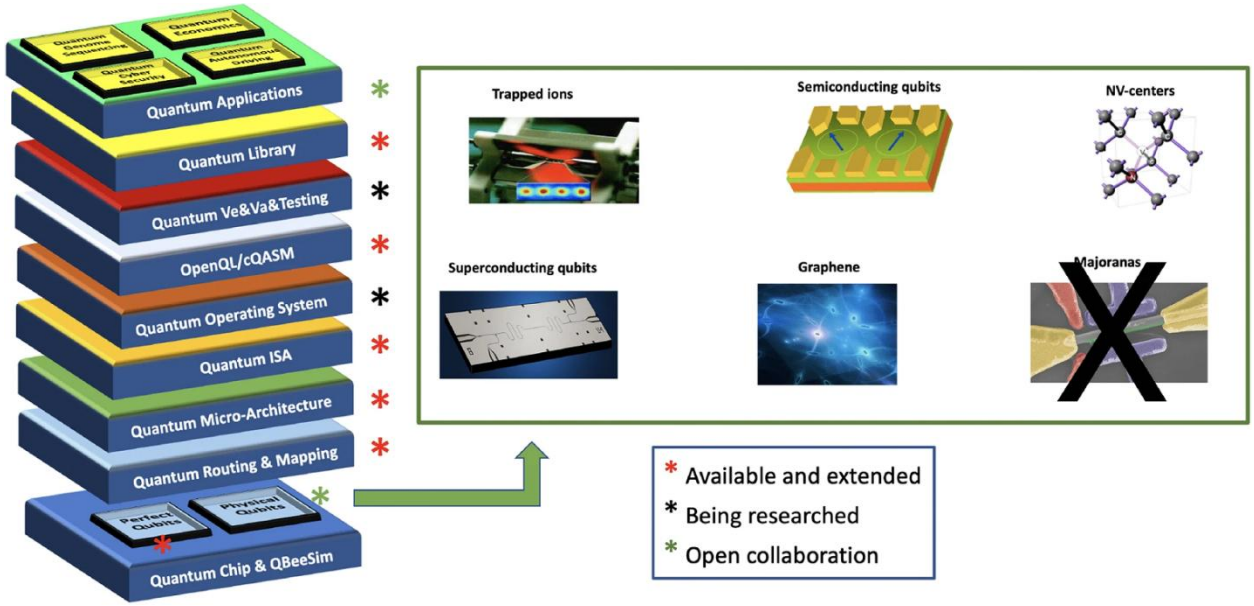

Figure 19: Sample quantum stack [16]

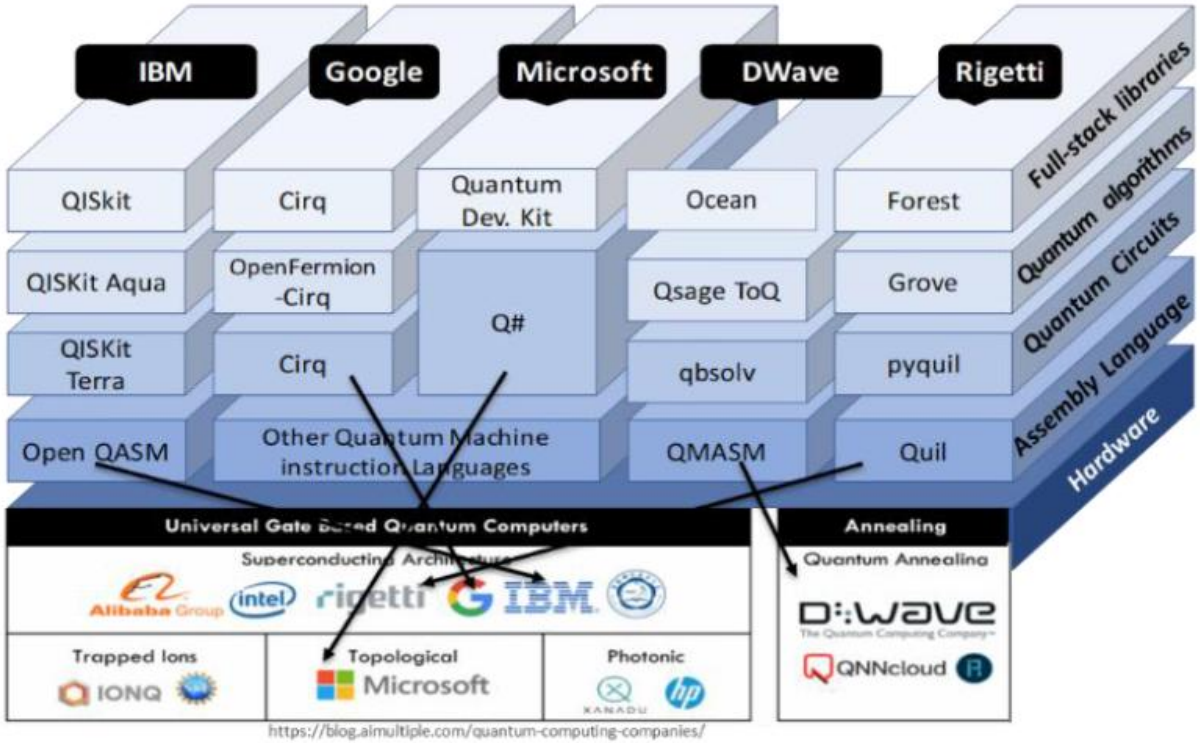

Figure 20: Mapping of technology needs to solutions [17]

Software and systems products are identified along with other products/ tools that fulfill the functional needs. These are then elaborated in the development & testing phase. Key solutions offered by the selected products at various stack layers are selected that satisfy the functional requirements of the product

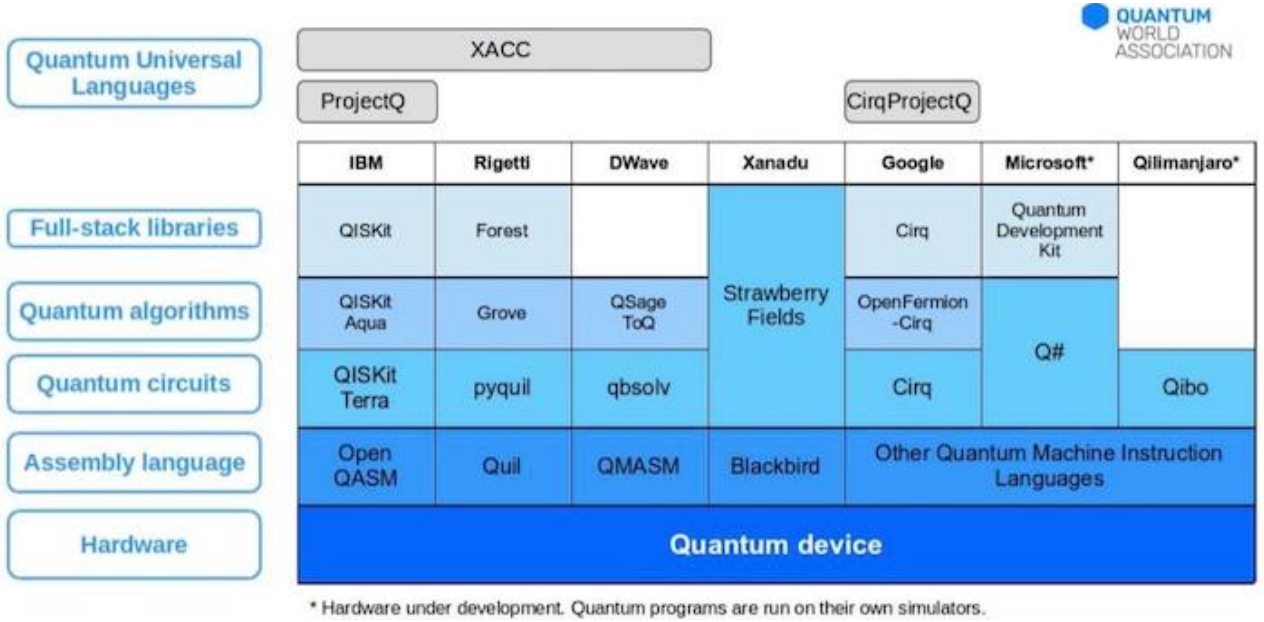

Figure 21: Quantum products [18]

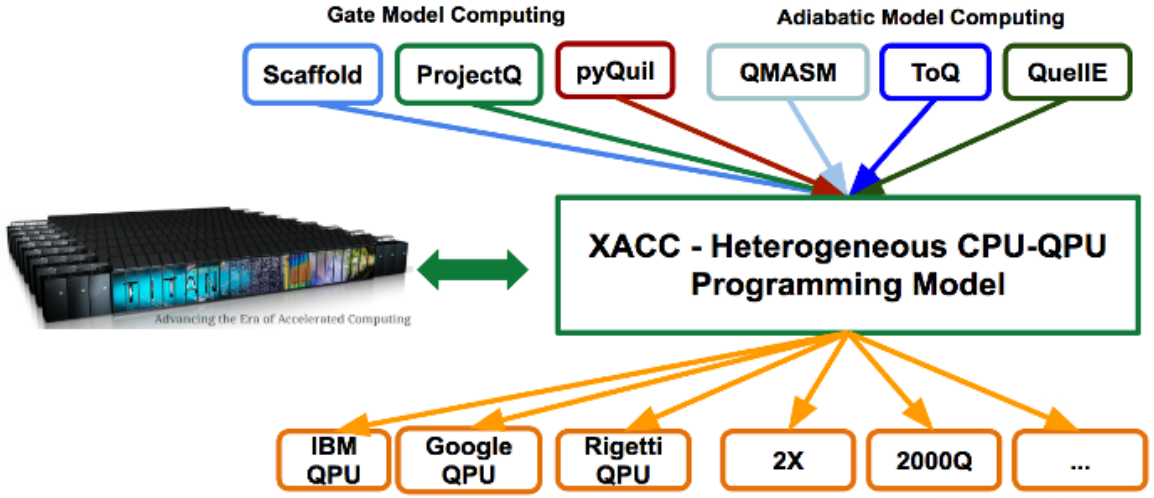

Figure 22: XACC CPU-QPU for various computing models [19]

One example of the technical stack is show in figure

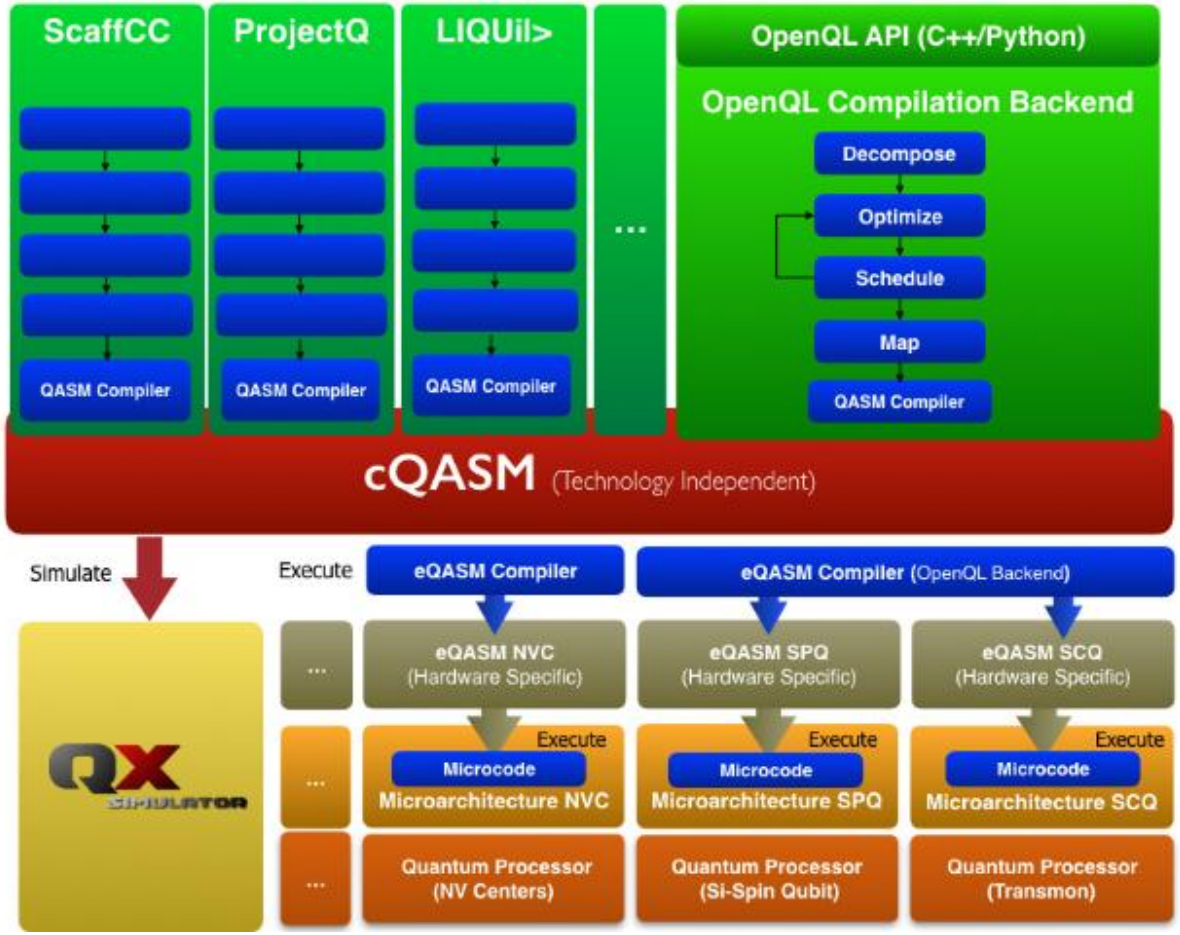

Figure 23: Technical stack example [20]

Development & Testing

Mainly concerned with DevOps, Testing, release, deployment and handover to operations, the working and final releases of assets and dependencies

Development is the last in this stack based on the upper layers. Modern development life cycles usually are agile in nature that allow flexible, scalable and adaptable to the change requests from all of the layers and even external feedbacks until a stable release is developed

This section offers the engineering features used for software, systems and multi-disciplinary tasks that are needed to implement the required product or service that works well in a given hosted environment, operating systems or under a hostile environment that is needed to fight an enemy cause in case of a military release

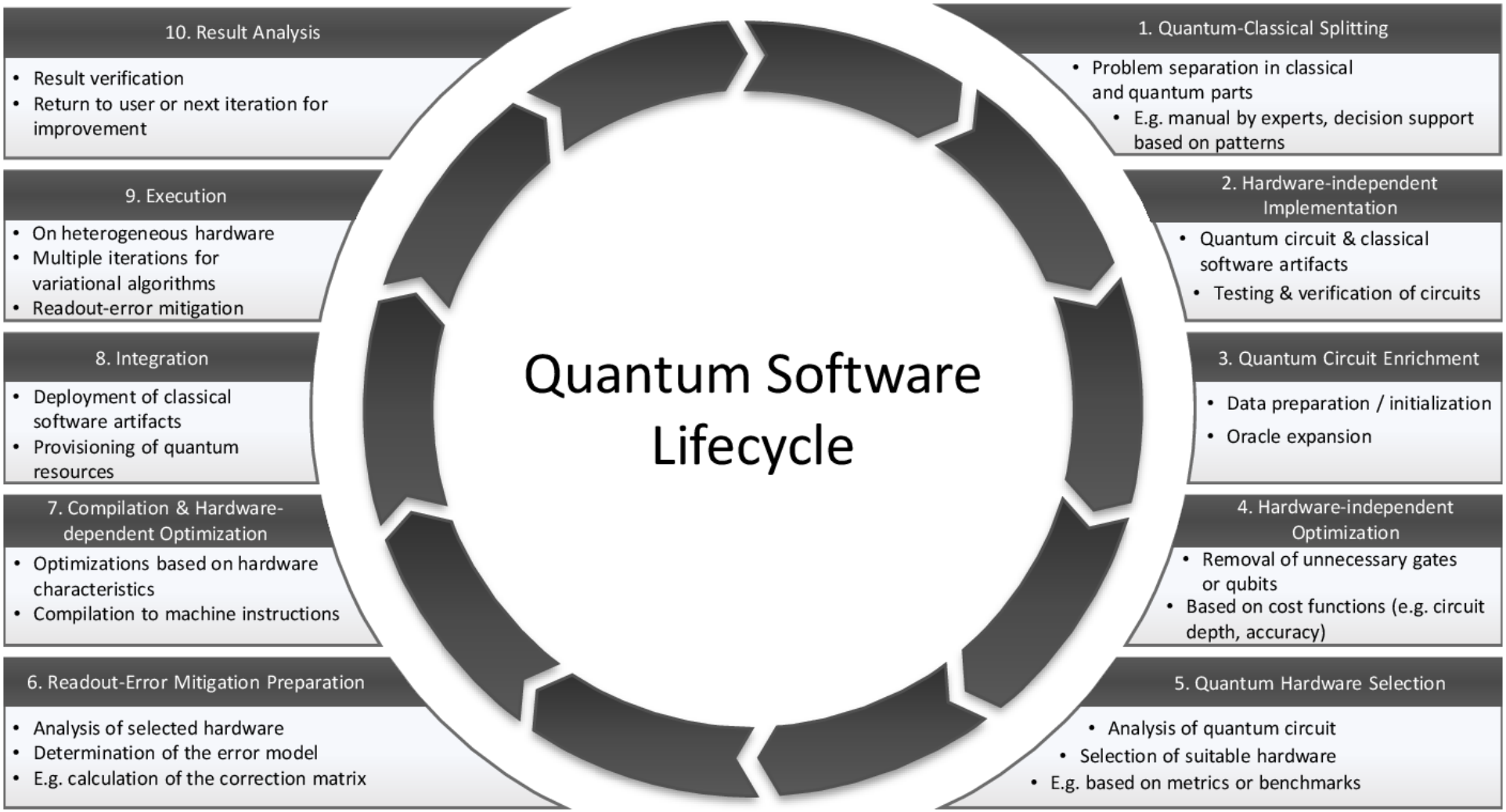

Figure 24: Quantum development life cycle [21]

As shown in the development lifecycle, there are various iterations that allow for various design updates across the full technology stack using agile development methodologies. Based on the earlier developed design documents the life cycle is followed in developing the product

Each of the 10 phases are iteratively developed along with required small releases that ensures product is developed with updates and applied across all other earlier releases to ensure backward compatibility with the requirements

Once the required entities are developed they are deployed to actual quantum and classical servers. One example

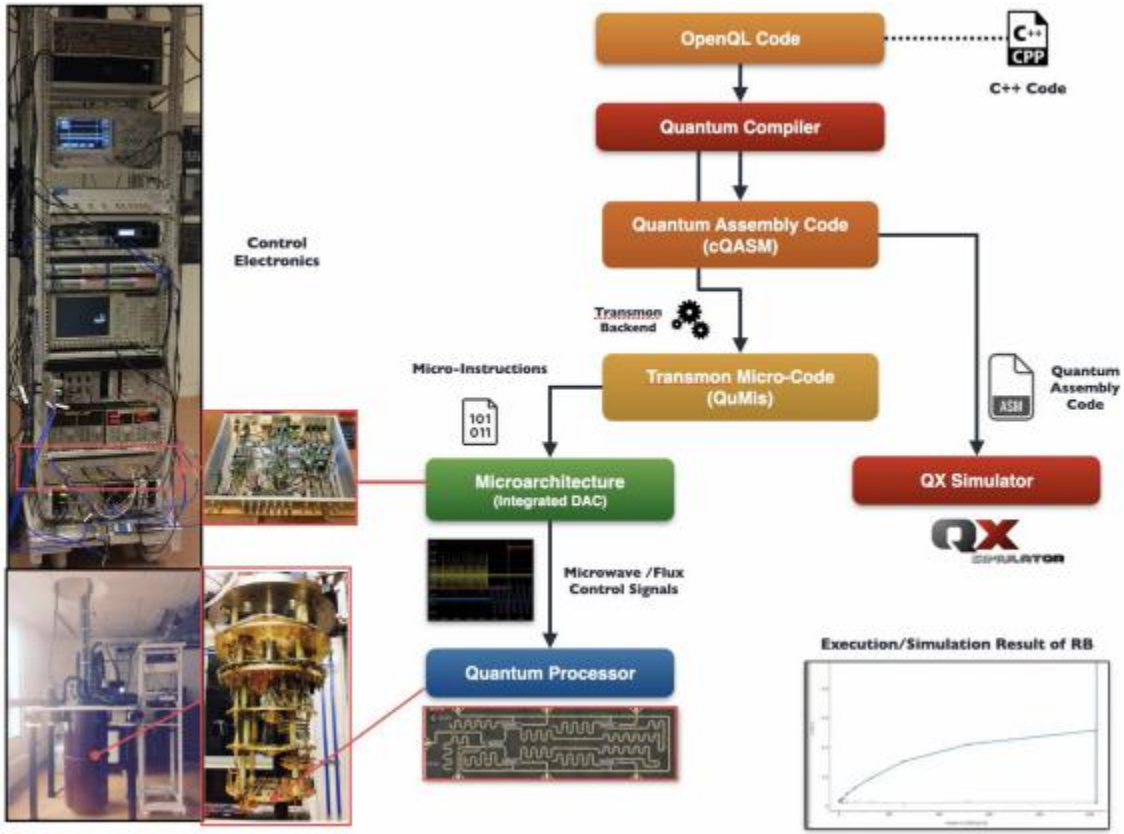

Figure 25: Quantum Deployment [22]

If there is a need for environment integration at client premises, User acceptance testing is conducted after deploying the final release. Any issues are rectified in development environment before redeploying the assets in client environment

References:

[1] Khan, Arif & Ahmad, Aakash & Waseem, Muhammad & Liang, Peng & Fahmideh, Mahdi & Mikkonen, Tommi & Abrahamsson, Pekka. (2022). Software Architecture for Quantum Computing Systems -A Systematic Review
[2] [3] Luis Jiménez-Navajas, Ricardo Pérez-Castillo, Mario Piattini. (2020). A Tool for Quantum Software Evolution
[4] Felix-Schiessl, (2021), Quantum Business Canvas, https://www.linkedin.com/pulse/quantum-business-canvas-how-we-went-from-idea-concept-felix-schiessl
[5] Silva, Carla & Aguiar, Ana & Lima, Priscila & Dutra, Inês. (2021). Mapping a logical representation of TSP to quantum annealing. Quantum Information Processing. 20. 10.1007/s11128-021-03321-8.
[6] Aurelien-Pelissier, (2019), https://github.com/Aurelien-Pelissier/IBMQ-Quantum-Programming
[7] Chuck Brooks, (2021), https://www.forbes.com/sites/chuckbrooks/2021/03/21/the-emerging-paths-of-quantum-computing/
[8] [11] Pérez-Castillo, R., Piattini, M. Design of classical-quantum systems with UML. *Computing* (2022), https://doi.org/10.1007/s00607-022-01091-4
[9] *Gemeinhardt, Felix et al. "Towards Model-Driven Quantum Software Engineering." 2021 IEEE/ACM 2nd International Workshop on Quantum Software Engineering (Q-SE) (2021): 13-15.*
[10] Aurelijus Morkevicius, Aiste Aleksandraviciene, Donatas Mazeika, Lina Bisikirskiene, Zilvinas Strolia (2017), https://doi.org/10.1002/j.2334-5837.2017.00350.x
[12] Value Analysis, Canada, (2022), https://www.valueanalysis.ca/fps.php
[13] Arnold, Steven & Cebon, David & Ashby, Mike, (2012), Materials Selection for Aerospace Systems. NASA/TM-2012-217411
[14] Upplabs, (2022), https://upplabs.com/blog/the-importance-of-functional-and-non-functional-requirements-in-software-development/
[15] QC Patterns, (2022), https://www.quantumcomputingpatterns.org/#/
[16] K. Bertels, A. Sarkar and I. Ashraf, "Quantum Computing—From NISQ to PISQ" in IEEE Micro, vol. 41, no. 05, pp. 24-32, 2021. doi:10.1109/MM.2021.3099195
[17] Pérez-Castillo, Ricardo & Serrano, Manuel & Piattini, Mario. (2021). Software modernization to embrace quantum technology, Advances in Engineering Software, 151. 102933. 10.1016/j.advengsoft.2020.102933
[18] Quantum World Association, (2022), All about Circuits, https://www.allaboutcircuits.com/news/can-new-programming-language-help-boost-innovation-quantum-computing/
[19] Alex McCaskey et.al, Eclipse Foundation, (2017), https://www.eclipse.org/community/eclipse_newsletter/2017/august/article4.php
[20][22] N. Khammassi, G.G. Guerreschi, I. Ashraf, J. W. Hogaboam, C. G. Almudever, K. Bertels, 2018, arXiv:1805.09607
[21] *Weder, Benjamin et al. "The Quantum software lifecycle." Proceedings of the 1st ACM SIGSOFT International Workshop on Architectures and Paradigms for Engineering Quantum Software (2020): n. pag.*

Disclaimer: All rights are owned by respective owners. We have no intention to infringe copyrights or brand names. All details, references are for educational illustration purpose only

For more details, contact below:

Contact

Bhadale IT Pvt. Ltd

Program Manager: Vijayananda Mohire, Email: vijaymohire@gmail.com