# Requirements for Quantum Software Platforms

Jose Luis Hevia Oliver[a]

*aQuantum by Alhambra, Albasanz 16, Madrid, 28037, Spain*

**Abstract**

In present days, we are attending the raise and fast evolution of the quantum machines. Without even being as powerful as they can be, we can experiment their first steps and enjoying with our "knowledge from scratch" training creating, designing, testing and executing quantum algorithms from a software engineer perspective.

But, how this new context can affect our vision and experience from a classical perspective? As IT industry experts, how those new quantum technologies can alter the life cycle of our projects, designs, tests, validations, deployments…? And of course, what is the impact of those new quantum technologies when creating new software architectures…?

In this paper, our goal will be to identify the challenges that must be faced when designing new software architectures in this new context: the raise of quantum software technology

**Keywords**

Quantum computing, Quantum software engineering, Quantum software platforms, Multidisciplinary teams, Quantum SDK, Quantum API.

## 1. Introduction

In present days, we are attending the raise and fast evolution of the quantum machines. Without even being as powerful as they can be, we can experiment their first steps and enjoying with our "knowledge from scratch" training actions that includes: creating, designing, testing and launching quantum algorithms from a software engineer perspective. And we will be trying to understand how these new pieces of computation fit in the engineering of software solutions.

But, to start in this new quantum world era, we must keep in mind a certain number of challenges that may be faced in order to walk following the right path:

- **Challenge 1**: There exists a limited set of quantum hardware providers that are investing a lot of resources trying to build a 100% reliable quantum computer. But today, this quantum machine is still a dream. Those providers are making a lot of discovers faster and faster in terms of upgrade the number of qubits and their quality, better scalability solutions, great community efforts and so on…and all of this tremendous work has advantages and disadvantages that can totally change the curse of a software solution.
- **Challenge 2**: In the mentioned set of quantum hardware providers, we must know that not all of them has create the same "thing". There exists two great paths and philosophy on how must be used the quantum mechanics, and some of the providers has bet for one of the paths, and the rest to the other. And of course, each of the providers put its unique vision to the development of its own technology… so not all them share the same techniques even though the principles seem the same.
- **Challenge 3**: Associate with each one quantum hardware provider, there exists a lot of SDK – APIS, libraries, simulators, documentation and all the stuff around that- that is precisely proprietary of each one of them. These elements compose the Quantum Provider Software Stack.

Q-SET'20: 1st Quantum Software Engineering and Technology Workshop, October 13, 2020, Denver — Broomfield, Colorado, USA

EMAIL: jluis.hevia@a-e.es (J. L. Hevia)

- **Challenge 4**: Determine the kind of solution that solves real use cases. Quantum computers are here not to replace to the classical ones [3]… instead, with the knowledge we have today and the technology we have, quantum computers are really good to exploit the parallelism of the nature…. So, NP problems that even a supercomputer cannot solve in million years, can be approached by not so powerful quantum computer. So, we must look up for a specific kind of problems –all of them impossible to solve today or are not viable due to the resources that would be needed to solve them- that will fit with the capabilities of a quantum machine
- **Challenge 5**: How to refactor all the software engineering knowledge [2] to implement a new software architecture with the best practices and quality assurance, keeping in mind the integration of quantum systems with defined roles and responsibilities, and the same requirements in terms of abstraction layer in the most classical of senses.

With all of this challenges in our hands, we will conclude with proposals to deal with all of them.

## 2. Challenges designing quantum software architectures

As software engineers we must be aware of the elements that we have at our disposal to make the best design of a software solution [4]. In our two years of analysis of new quantum platforms we have identified five challenges.

### 2.1. Challenge 1: Quantum hardware providers

As mentioned in the introduction, there exists a limited set of quantum hardware providers than offer -and compete to- provide an impressive variety of quantum services. Even though these services are not powerful enough to bring all the capabilities of the quantum mechanics, they put in our hands the possibility to explore those services and take long-term actions.

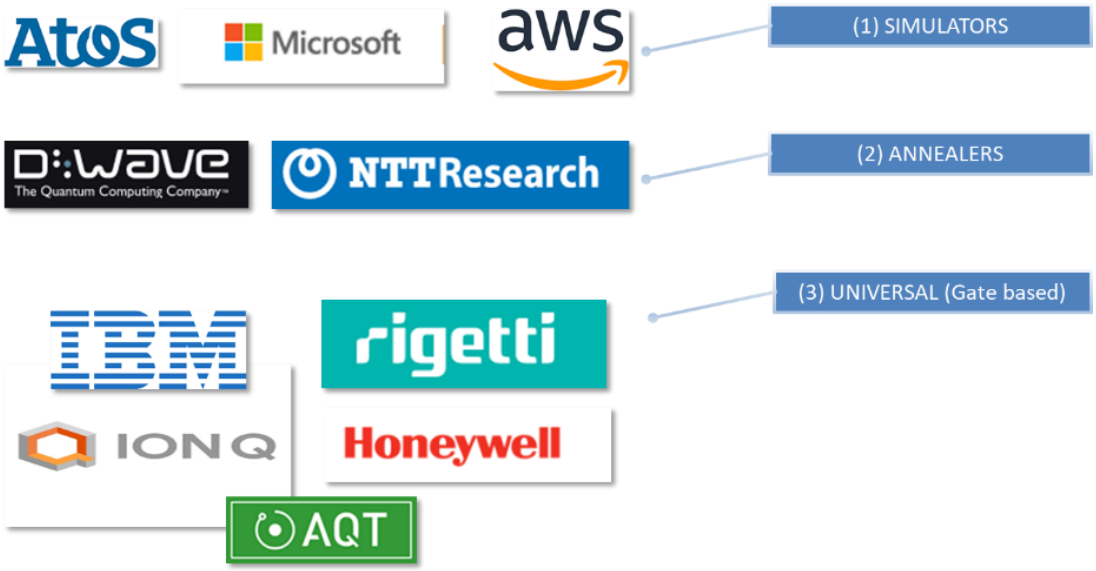


**Figure 1** Ecosystem of quantum providers

But each one of quantum hardware providers, has its own vision and tools to provide services. This means that we must do a big research in order to acquire the enough knowledge to:

- Know how to consume the quantum services
- Know how the quantum mechanics characteristics are engineered in order to make it programmable in a reasonable way

- Know which are the limitations of the computer, because we are at the beginning of the quantum era
- Foresee the evolution of the provider in order to foresee potential changes in the investment we are doing trying to adapt the technology to our new solutions.
- Evaluate if it's available a simulator or we can access to real hardware
- Know how much it cost

## 2.2. Challenge 2: Different approaches, different ways to solve a problem

It is relevant in this context to keep in mind that currently exists two approaches on how to use the quantum mechanics: the topological approach (quantum gates multi-purpose computers) and the annealers approach (quantum annealing effect). Each of them is different and requires different requirements as each of them applies to different kind of problem resolutions.

The topological computers based on quantum gates, try to put in our hands a reinvention of the multi-purpose programmable machine with totally new rules to do that, based on amazing quantum characteristics such as superposition: high speed for really complex calculations, information stores in qubits, infinite number of possible states, unprecedented parallelism, probabilistic calculations, linear algebra's defined operations…

Quantum annealing, on the other hand, uses a very specific behavior of Nature that let the engineers the opportunity to play with the optimization Nature does when we have several possible paths and all them interconnected in some way.

Each of these alternatives has its own benefits and disadvantages, and of course, its own specific uses cases in which they best fit… and must be evaluated in order to choose the best approach when we design a new software architecture for a specific set of problems.

On the other hand, with critical impact as part of this challenge: when we start a new software solution definition… Do we have the proper resources in our team?... that is, can we design, analyze or understand the needs of this new type of quantum requirements and be able to design the right solution for them only with software engineering technics?

## 2.3. Challenge 3: Each quantum provider has its own software stack

Again, I need to remark the context in which every quantum provider develops the technology defining its own path and software stack [5]. As it has been occurred in the classical approach… each provider offers its own technology. That is the salt of the IT world. And of course, it is a reality in the quantum context.

So, first of all, we must know the catalog of development platforms available for each quantum provider, its potential and select it carefully when a new project is being prepared.

At the end, our concern is that all of the emerging quantum technologies are growing and evolving day by day… and it involves a lot of experiments, decisions, changes, evolutions… in a really short period of time. From the perspective of the provider all their movements are coherent with the research activity they are involved… but the impact from a software's solution architecture perspective it may be a truly headache and a big risk in terms of resources, time and costs.

For example, IBM Qiskit or Microsoft Q# are development platforms that are changing every 3-6 months… and these changes modify API, libraries, ways to compile or even deprecated/redefined service's calls. The changes are reactive and response in every situation, a state-of-the-art research. Every minute something may be really good or a big mistake –in terms of research processes- that must be rethought to open new research branches.

During the last year, I was forced to refactor my own quantum development courses of IBM Qiskit or Microsoft Q# due to changes in the core of the platforms: changes of the libraries and languages with a great risk of conflicts, different ways to do the same in order to be more accurate or complete respecting the quantum theories or experience acquired… or even the community reaction go to some kind of language or environment (python, Jupiter notebooks…etc.).

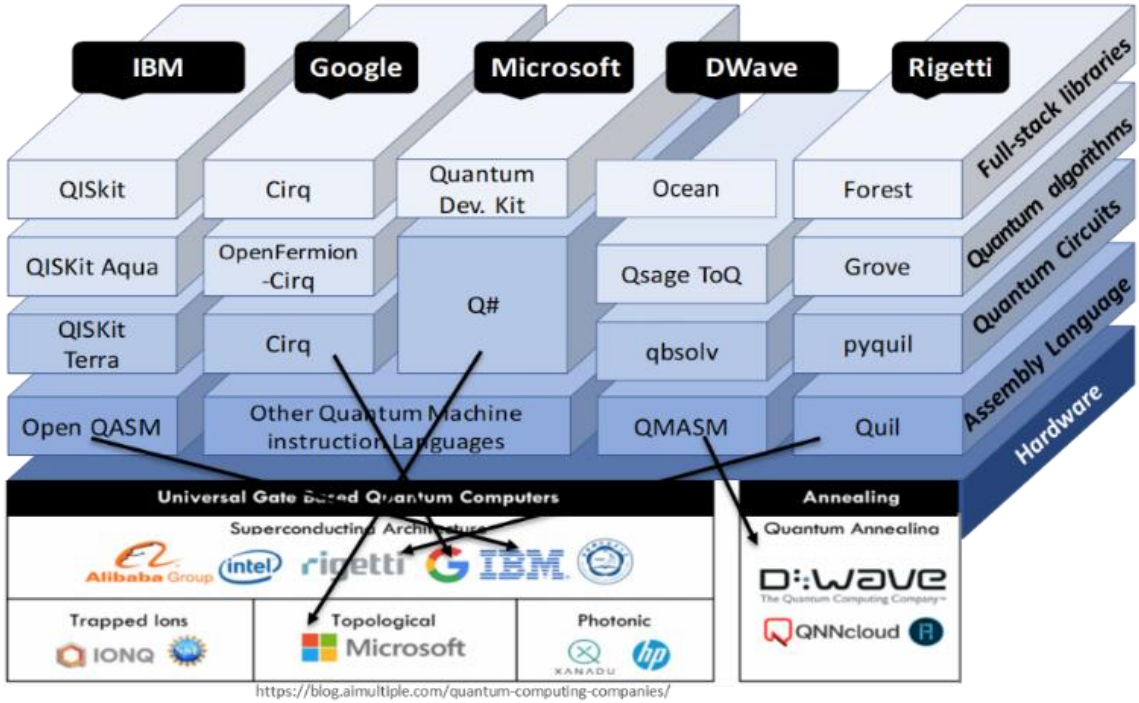


**Figure 2** Software stack platforms

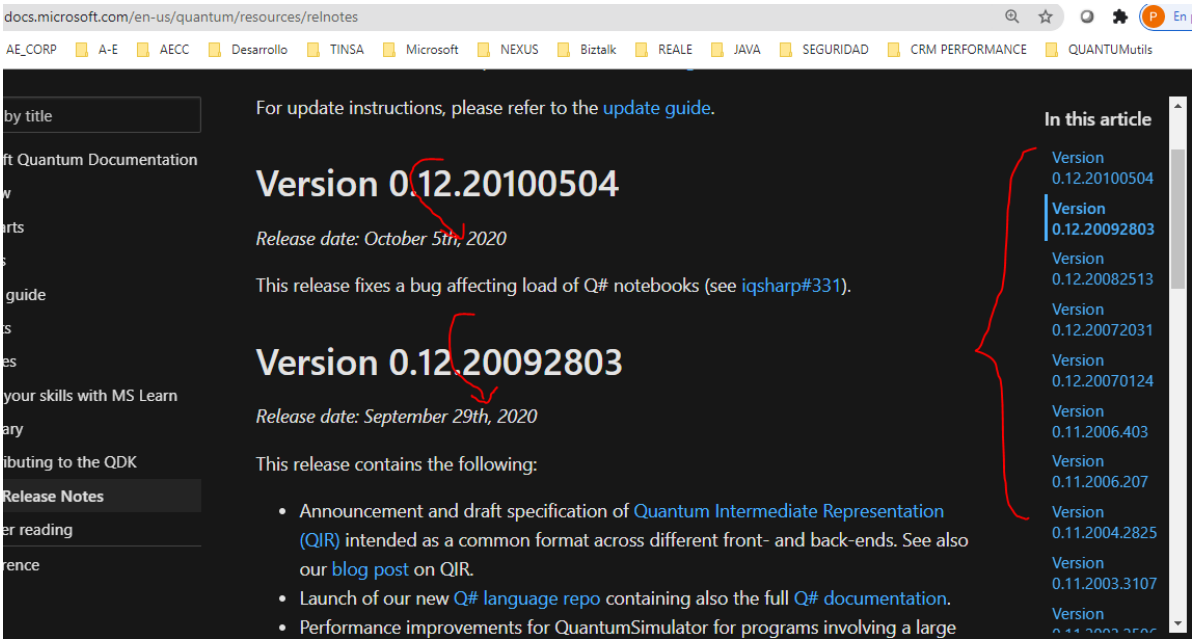


**Figure 3** real example, the fast evolve of MS Q# SDK (https://docs.microsoft.com/en-us/quantum/resources/relnotes)

Most of this software stack has simulators and real hardware, so that means we have two contexts for testing environments… the ideal one and the real one… but they do not always have the same limitations, powerful features, error control types or return the same results.

So, in order to design a new quantum solution architecture, we have to be careful in terms of be able to withstand very short-term evolution of technology platforms. And be ready to FAST refactor the models with a truly flexible and modular requirements from design.

## 2.4. Challenge 4: Determine the kind of solution for real quantum use cases

This is truly a hot pot. All the main actors of this amazing quantum adventure are behind the discovery of the set of use cases that fits like a crystal shoe on the special foot of a quantum Cinderella.

As I said in the introduction, not every problem fit with the powerful quantum mechanics of the quantum computers. So each of one of the available Quantum providers (not only the providers of quantum machines) are searching the most complete catalog of use cases that take advantage of the power of the quantum technology.

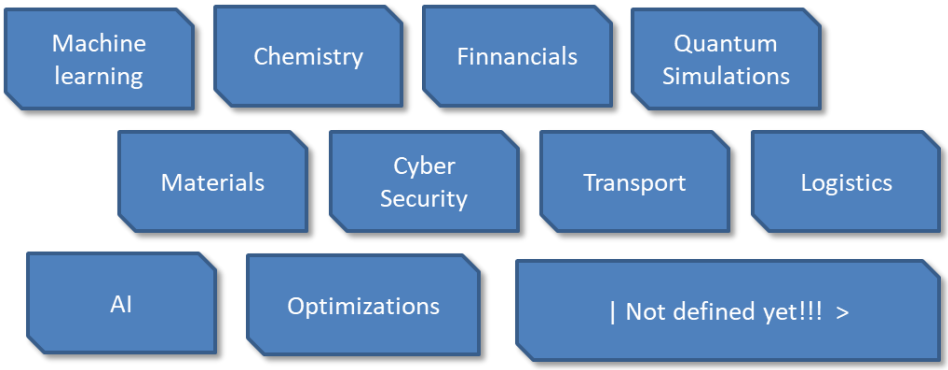


**Figure 4** the search of the Use Case...

As software engineers, we must face this in the same way as we face the domain of the problem when we start a new project. But with a new value added: the kind of the problem could be fit better in certain types of quantum technologies than the others…So, be careful determining the kind of solution. And as said before, ensure you have the resources you need in your team in order to understand the requirements that solve the problem.

## 2.5. Challenge 5: Refactor the principles of software engineering

All the challenges introduced respond to the fundamentals of the software engineering life cycle when designing a software architecture. But once we have all ready and the pieces of the puzzle ready on the table… what will be the best architecture solution in order to be able to minimize the risks of the proposed challenges?

If I have the correct -human- resources in my team and we can design the solution in terms of functional requirements… what is the best approach to put the new quantum capabilities under the classical system? At this point, the *hybrid architectures* fit perfect in this context. We must design the new software architecture in classical layers, and one of these layers must encapsulate the quantum features under some kind of services.

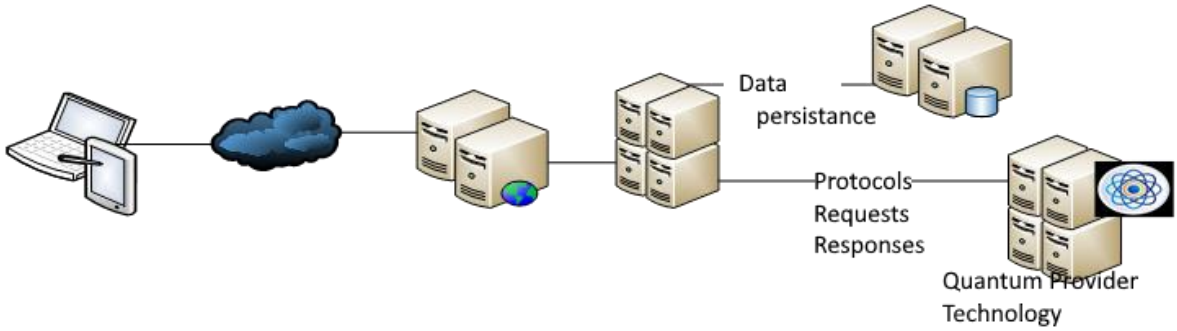


**Figure 5** How to include a Quantum software Layer in a classical system architecture

In order to provide and open and interoperable layer, the quantum services must be encapsulated under a web services layer to be more precisely. This let the architecture enough capabilities to provide an abstract layer isolating the quantum complexities…

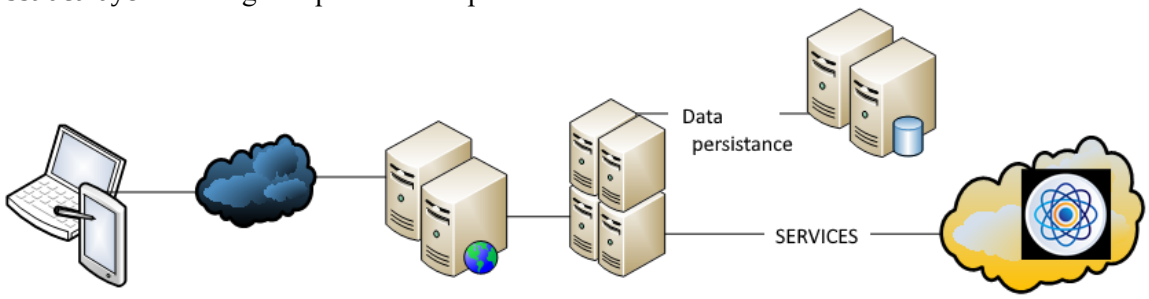


**Figure** 6 Isolating Quantum as a Service

But we must be aware with current democratization limitations… Depend on agreement level the requests could be queued in the quantum machine execution layer and asynchronous process and the response time must be a problem to be managed. To do that, in this challenge we need to keep in mind:

- What abstractions must be defined to make a request and normalize the response?
- What protocol must be designed to interact with the quantum services?
- How can we measure the results with quality enough?

## 3. Conclusions: Quantum software accelerators

After two years researching how to engage the emerging quantum computers and their services in the software engineer architecture and it's fundamentals, it is our concern to raise the alert that must be taken in care around the global preparation that is needed to accomplish the quantum revolution.

Current, we are not prepared enough in terms of software engineer fundamentals, to accomplish the development of a completely new project based on quantum computation. Knowledge, techniques, tools, methodologies, design patterns, telemetry, quality, technology adoption…. Are issues that today are not stable¿?

- We must prepare as soon as possible and efficiently, a new generation of quantum IT professionals [1], creating…Great academia courses to put the knowledge to our future engineers,
- Great professional courses to re-arm the current engineers brain,
- Multidisciplinary teams to design a complete solution with complementary points of view

We must also provide companies with services and/or great valuable software platforms that makes possible the quantum software accelerators: offering complete set of tools that simplifies the comprehension and design of the quantum requirements, a complete set of methodologies, techniques and best practices (that let us not reinvent the wheel), a complete set of experience badges, global algorithms and common repositories to access all the knowledge needed to FAST Access to quantum technologies with minimal effort; and offering platforms and ecosystems to abstract the quantum details to the lower levels, making possible the focus on the problem and not in the details.

## Acknowledgements

The author wants to acknowledge the aQuantum group for the strong bet that it is making on quantum software technologies, specially Guido Peterssen and Mario Piattini for their support and enthusiasm. And specially, the IEEE Quantum Week for the opportunity that has bring to all the professionals around quantum technologies.

## References

[1] Peterssen, Guido," Quantum technology impact: the necessary workforce for developing quantum software", http://ceur-ws.org/Vol-2561/paper1.pdf,Vol-2561, 2019.

[2] M. Piattini, G. Peterssen, R. Perez-Castillo, J.L. Hevia, M.Serrano, “QANSWER 2020 - International Workshop on the QuANtum SoftWare Engineering & pRogramming”, University of Castilla-La Mancha (UCLM) Spain, ceur-ws.org,Vol-2561, 2019.
[3] Perez-Castillo, Ricardo, “Reengineering of Information Systems toward Classical Quantum Systems”, http://ceur-ws.org/Vol-2561/paper7.pdf. ,Vol-2561, paper 7, 2019.
[4] Hevia, Jose Luis, “Introduction to Quantum Development”,ceur-ws.org,Vol-2561,2019
[5] K.M. Svore ; A.V. Aho ; A.W. Cross ; I. Chuang ; I.L. Markov, “A layered software architecture for quantum computing design tools”, Computer 39 (1), 74-83, 2006.