# What’s Special about Requirements Engineering for Quantum-Classical Systems?

*¿Qué tiene de especial la ingeniería de requisitos para los sistemas cuántico-clásicos?*

*¿O que há de especial na engenharia de requisitos para sistemas quânticos-clássicos?*

*Horacio Pendenti [1,(*)], Libardo Pantoja Yepez [2], Ricardo Perez Castillo [3],*
*Julio Ariel Hurtado Alegría [4], Leandro Antonelli [5], Alejandro Fernandez [6]*

Recibido: 10/10/2024 Aceptado: 10/10/2024

**Summary. -** Quantum computing will impact all software development life cycle phases to a greater or lesser extent. With support from current literature, we discuss its potential impact on requirements engineering. We identified four main trends. A feasibility study focusing on quantum aspects is required before or after requirements analysis. The early analysis of the availability and the choice of hardware plays a bigger role than in developing classical software. Requirements are split into the functional, non-functional, classical, and quantum dimensions. The potential speedup of a quantum solution should be estimated and considered up-front.

***Keywords:*** Requirements elicitation; Hybrid Quantum-Classical Systems; Quantum Requirements Engineering.

(*) Corresponding Author.

[1] Researcher. Universidad Nacional de Tierra, del Fuego, Ushuaia, Argentina. hpendenti@untdf.edu.ar, ORCID iD: https://orcid.org/0009-0007-4740-8109

[2] Researcher. Universidad del Cauca, Popayan, Colombia. wpantoja@unicauca.edu.co, ORCID iD: https://orcid.org/0000-0002-7235-6036

[3] Researcher. Universidad de Castilla – La Mancha, Talavera de la Reina, España. Ricardo.PdelCastillo@uclm.es, ORCID iD: https://orcid.org/0000-0002-9271-3184

[4] Researcher. Universidad del Cauca, Popayan, Colombia. ahurtado@unicauca.edu.co, ORCID iD: https://orcid.org/0000-0002-2508-0962

[5] Researcher. LIFIA, F.I., Universidad Nacional de La Plata and CAETI, F.T.I., Universidad Abierta Interamericana, La Plata, Argentina. lanto2004@gmail.com, ORCID iD: https://orcid.org/0000-0003-1388-0337

[6] Researcher. LIFIA, F.I., CICPBA and Universidad Nacional de La Plata, La Plata, Argentina. alejandro.fernandez@lifia.info.unlp.edu.ar, ORCID iD: https://orcid.org/0000-0002-7968-68717

***Resumen. -** La computación cuántica afectará en mayor o menor medida a todas las fases del ciclo de vida del desarrollo de software. Con el apoyo de la literatura actual, analizamos su posible impacto en la ingeniería de requisitos. Identificamos cuatro tendencias principales. Se requiere un estudio de viabilidad centrado en los aspectos cuánticos antes o después del análisis de requisitos. El análisis temprano de la disponibilidad y la elección del hardware desempeña un papel más importante que en el desarrollo de software clásico. Los requisitos se dividen en dimensiones funcionales, no funcionales, clásicas y cuánticas. La aceleración potencial de una solución cuántica debe estimarse y considerarse de antemano.*

***Palabras clave:** Obtención de requisitos; Sistemas híbridos cuántico-clásicos; Ingeniería de requisitos cuánticos.*

***Resumo**. **-** A computação quântica impactará todas as fases do ciclo de vida do desenvolvimento de software em maior ou menor grau. Com o suporte da literatura atual, discutimos seu impacto potencial na engenharia de requisitos. Identificamos quatro tendências principais. Um estudo de viabilidade com foco em aspectos quânticos é necessário antes ou depois da análise de requisitos. A análise inicial da disponibilidade e a escolha do hardware desempenham um papel maior do que no desenvolvimento de software clássico. Os requisitos são divididos nas dimensões funcional, não funcional, clássica e quântica. A aceleração potencial de uma solução quântica deve ser estimada e considerada antecipadamente.*

***Palavras-chave:** Elicitação de requisitos; Sistemas híbridos quânticos-clássicos; Engenharia de requisitos quânticos.*

**1. Introduction. -** Software Engineering (SE) emerged as a response to the software crisis, characterized by the challenges and difficulties of developing complex software systems. The initial concepts in this discipline began to be discussed in the late 1960s and early 1970s [1], coinciding with a high degree of maturity in hardware technology, the proliferation of programming languages, and the software ecosystem stratifying into layers. Since then, SE has evolved to consolidate processes for the different challenges in developing complex and high-quality software, always on the same underlying hardware but adapting to and adopting new technologies and environmental needs. In recent years, a new technological milestone has become a reality with the emergence of the first real quantum computers, a development that SE cannot ignore [2].

Quantum computing is still in its early stages, with hardware that is still not entirely reliable but showing steady advances. Efforts in research and development by governments, companies, and other entities have been the primary drives of these advances. While significant progress is expected in the near future, with a variety of technologies and languages available, it is already possible to develop quantum programs and run them on real quantum computers from different providers and architectures [3, 4]. So, as this evolution takes place, it is expected that more problems can be solved using quantum algorithms on quantum computers. The conditions are met for SE to begin adapting its processes, incorporating the implications of this new technological milestone [2, 5].

It is not foreseen that quantum computers will replace classical computers entirely [6, 7]. Instead, quantum and classical software will operate together [8]. At the moment, quantum software requires classical programs acting as drivers, from which they obtain inputs and other parameters that are sometimes used to build quantum software dynamically. These drivers run quantum software on specific quantum computers in the cloud. Classical software is also devoted to receiving the execution answers from quantum software and interpreting them to provide an output to the end-users. This interface between both paradigms, in which a classical program requires a quantum service, reinforces the idea of defining Quantum Software Engineering (QSE) as a classical SE encompassing the peculiarities of quantum software development.

Various studies argue that the impact of quantum development within SE occurs to a greater or lesser extent in all phases of the Software Development Life Cycle (SDLC) [9, 2, 4, 10, 11]. In this work, we focus on the impact of this new technological development on requirements engineering. Our goal is to understand how the potential inclusion of quantum components in our software could change how we conduct requirements analysis, specification, validation, and management. To contribute to this discussion, we identify and review some of the most relevant contributions currently in the literature.

**2. Requirements Engineering. -** Requirements engineering constitutes a critical phase in the software development life cycle, laying the foundation for all subsequent design and development activities. It ensures stakeholder alignment, enhances cost and time efficiency, mitigates risks, assures quality, and facilitates regulatory compliance. Proper management of this phase significantly influences the project's success by establishing clear, achievable goals guiding the development process. This is especially critical in quantum-classical software development, where hybrid systems and various technologies are involved, necessitating precise and comprehensive requirements to integrate diverse functionalities seamlessly.

There are three main philosophies of software development: classic, agile, and hybrid. Every strategy organizes the development process and deals with the requirements differently. In a classic life cycle, extensive documentation is produced and consumed. It consists, for example, of software requirement specifications with hundreds of use cases specified incrementally [12] or as a waterfall [13]. In agile development, the communication of the requirements relies on a specific role within the team, the product owner, and on a simple documentation artifact, the User Story. The hybrid approach mixes the above strategies for balancing documentation, discipline, and agility and achieving specific business goals [14]. Any attempt to study how quantum computing affects requirements engineering should consider all three of them.

Use Cases and User Stories capture functional requirements. They describe the functionality the system must provide (algorithms that transform input to output data). In addition to functional requirements, some requirements describe how the algorithms must be implemented regarding speed (performance), readability (legibility), resource usage, etc. These requirements are called non-functional requirements and condition the implementation of the functional requirements. Since quantum computing introduces radically different concepts and paradigms from classical systems, non-functional requirements have become especially relevant.

**3. Requirements Engineering for Quantum Software. -** In their Quantum Development Life Cycle (QDLC) proposal, Dey et al. [15] include a preliminary phase (called Quantum Feasibility Study) in which a detailed study of technical, operational, and economic feasibility specifically focused on quantum aspects is made. This initial phase, preceding requirement specification, addresses key questions: Is quantum hardware capable of supporting the required computation? Are there efficient algorithms to enhance traditional solution speed? Do we have the necessary scientific expertise? Do the quantum project's benefits justify its higher costs? Then, in the requirements specification phase, the availability of a quantum programming language and an appropriate compiler for the problem to be solved and the target hardware are analyzed. The availability of other tools related to quantum development, such as Quantum Integrator Plugins, Logical Quantum Circuit Synthesizer for circuit optimizations, and Classical Validators, is also studied in this preliminary phase. Furthermore, the quantum hardware to be used is specified, where the *Quantum Volume* metric, which includes the number of qubits, error rates, and qubit connectivity, is an important part. In addition, the Physical Machine Description (PMD), which describes the quantum hardware technology, completes the specification of quantum requirements. Similarly, Hernández González et al. [16], while trying to approach quantum software development from an agile perspective, suggest adding an initial Inception phase to an agile methodology such as Scrum (which then becomes a hybrid methodology). In the Inception phase, all project participants become familiar with the project characteristics and technological requirements.

The above discussion includes several aspects and activities that current research considers in different phases and/or with different denominations, groupings, or intensities. Such is the case of the work by Weder et al. [9]. They propose a phase after the requirements analysis, called quantum-classical splitting, within a general model of QDLC that orchestrates quantum and classical developments through workflows. Perez Castillo et al. [11] propose a quantum life cycle, adapted from the Incremental Commitment Spiral Model (ICSM), with an exploration phase in which the requirements play an important role. This proposal to adapt the initial phase of the classical SDLC to quantum development shows that further study of classical SE processes, such as Requirements Engineering, is necessary.

Adapting requirements engineering processes is necessary to cope with the constraints of quantum computing and to establish solid foundations for effective Quantum Software Requirements Engineering (QSRE). However, how to accomplish this remains unclear. The initial studies in this field exhibit varied content, ranging from those suggesting a starting point for discussion to those proposing solutions to specific QSRE issues. Among those pursuing a solution, one suggestion involves dividing requirements for quantum software into two well-defined sets: domain requirements and quantum requirements [17]. The former requires knowledge of the problem domain similar to traditional development practices, while the latter necessitates familiarity with Quantum Computing (QC). This latter case poses a significant challenge for Software Engineers, as there is generally a lack of domain-specific knowledge in quantum mechanics [18]. In another surveyed study, a review of the impact of quantum computing on all phases of a classic waterfall SDLC is conducted, where the brief Requirements phase suggests that in the process of specifying requirements, it is crucial to define user acceptability early on and establish critical performance requirements to determine if the benefits (still not fully defined) of quantum technology are sufficient over classical computing [19].

When studying more deeply the possibilities of an improved QSRE proposal, some recent advancements are taken into account, which, among other characteristics, suggest a Quantum Development Life Cycle (QDLC) incorporating modifications derived from the effects of quantum computing, followed by specific suggestions for each phase. While some authors propose adding the majority of quantum aspects in a phase preceding QSRE [15], others choose to do so in a subsequent one (such as [9], in a phase called Quantum Split), others maintain it in the same QSRE phase [20], and others discuss requirements without specifying their location in the life cycle. These works are complemented by other contributions that enrich this discussion in scope and depth. Among these, the studies of Saraiva et al. [21] are analyzed, framing Non-Functional Requirements (NFR) derived from quantum hardware in ISO 25010 quality model, and the NISQ (Noisy Intermediate-Scale Quantum) Analyzer [22] that analyzes and selects an appropriate algorithm implementation and a suitable quantum computer for a chosen quantum algorithm and specific input data.

Classifying requirements is a common proposal for several of the studied works. They propose identifying the functional and non-functional requirements, as in classical RE, and classifying them into classical, quantum, or hybrid categories. In the case of hybrid requirements, further disaggregation is necessary to accurately determine whether they belong to the classical or quantum domain. Regardless, whether a requirement should be fulfilled by a quantum or classical service is not a trivial decision. In addition to expert opinion, utilizing a framework such as the one proposed in [23] can provide additional assistance in decision-making.

Another crucial aspect of handling requirements is conducting a feasibility study for each requirement. With this goal in mind, a practical approach to this analysis focuses on three main aspects: Quantum Technical Feasibility, Quantum Operational Feasibility, and Economic Feasibility. The first aspect entails analyzing the existence of NISQ quantum hardware capable of supporting the algorithm. For this, the width (Qubit count) of the quantum algorithm is an essential metric, as well as the Quantum Volume (QV) [24], which encompasses factors such as depth (length of circuit layers). Depth plays a crucial role in determining the QV and consequently impacts the reliability of the software. At this stage, tools like the one described in [25] may be useful for estimating the necessary quantum hardware applicable to various quantum technologies and strategies for fault tolerant encoding. The Quantum Operational Feasibility involves assessing the extent to which the required software applies to solving the business problem and meeting user requirements. In this regard, it is necessary to address the availability of appropriate algorithms, assessing whether the less complex quantum part will result in a significant speedup and whether the more complex one will not entail an unreasonable cost increase. Furthermore, to complete this task, analyzing the availability of skills in quantum computing expertise within the development team is necessary. And finally, Economic Feasibility evaluates if the resulting software will represent a useful gain for the client.

When a requirement has been classified as “quantum” and “feasible,” it is advisable to record in its specification whether the algorithm implementing this solution can be improved with corrections or new implementations. Thus, it will be easier to leverage advancements made in associated algorithms, aiming to enhance the system’s reliability and efficiency.

Specifying all the classical and quantum software required for development throughout the project is necessary. Among the required software, various categories can be identified, such as Quantum Tools, which essentially refer to programming languages and compilers; Quantum Integrator Plugins, which are interfaces between the programming language and the simulator or quantum hardware; Logical Quantum Circuit Synthesizer, an optimizer of quantum circuits aimed at improving their efficiency and reducing error probabilities; Classical Validator, the classical software for verifying the solution provided by a quantum algorithm; and Classical Software Requirement Module, a classical software tool for systematically organizing all requirements in a Software Requirement Specification (SRS) document.

The Specifications section considers both quantum and classical hardware requirements for the project. Thus, it is necessary to specify at least the Qubit count, which is the number of qubits required by the quantum computer; the QV, a single-number metric that measures the largest random circuit of equal width and depth that the computer successfully implements; the Physical Machine Description (PMD) of the underlying quantum hardware technology, the physical arrangement of qubits, error rates of quantum operations, qubit connectivity, and other aspects related to the physical implementation of the quantum system; and the Classical Hardware Processor, which is the classical hardware specification, running a classical operating system and tools working synergistically with underlying quantum hardware.

**4. Conclusions. -** This article examined various studies in the field of QSRE, most of them focusing on requirements elicitation and specification. Little literature was found on requirements verification, validation, and management. Much remains to be done in this regard. However, by applying proven techniques from classical requirements engineering suitably adapted to quantum computing, consistent requirements and design models can be achieved. Such is the case developed by Yue et al. [20] for requirements or the UML extensions proposed by Pérez-Castillo et al. [26], where a design model can be achieved using a widely used tool in the software engineering community. A consistent classical-quantum requirement and design model is an important starting point for verifying and validating requirements.

One converging aspect in the literature is the necessity to classify requirements into functional and non-functional categories, followed by further sub-classification into quantum, classical, or hybrid. This initial task is common across various approaches and precedes the analysis of the specific challenges posed by quantum software.

Another activity that should be included early in the software development process, even before requirements analysis, is a quantum feasibility study. This study determines quantum requirements' technical, operational, and economic feasibility.

Throughout this article, a highlighted aspect is the significant impact of hardware choice on the RE process. The type of computer to be used, whether gate-based, annealer-based, or both, and various hardware-dependent metrics are highly relevant factors that must be analyzed during this life cycle phase. However, significant advancements in quantum technology are expected to gradually alleviate the strong constraints of quantum hardware. Furthermore, the recent start of a transition towards Fault-Tolerant Quantum Computing (FTQC) [27] invites us to consider both QSRE and the entire QSE with that perspective in mind. Therefore, the discussion in this field remains wide open for ongoing impact assessment and the search for more suitable alternatives for QSRE tailored to the achieved hardware.

Although the feasibility study mentioned earlier includes the economic aspect, a more comprehensive investigation is required to evaluate the relative benefits of the speedup provided by the quantum solution compared to the increased associated costs. The involvement of a collaborative team of quantum experts and the utilization of quantum facilities, among other factors, may contribute to raising the project budget.

Finally, regarding the methodological approach used for the QRE process, the analyzed research does not thoroughly explore requirements specifically applied to agile, classical, or hybrid methodologies. However, according to the various proposed QDLC models, predominantly classical methodology practices are observed, with iteration as a prominent element. Nonetheless, some less thorough studies suggest that modified agile practices transformed into hybrids can be applied. Regardless of the methodology used, the key activities or discussions concluded in this work should be considered in a quantum software requirement engineering process.

## References

[1] I. Sommerville, Software Engineering, 10th ed. Pearson Education Limited, 2016.

[2] M. Piattini, M. Serrano, R. Perez-Castillo, G. Petersen, and J. L. Hevia, “Toward a quantum software engineering,” IT Professional, vol. 23, no. 1, pp. 62–66, 2021.

[3] M. A. Serrano, J. A. Cruz-Lemus, R. Perez-Castillo, and M. Piattini, “Quantum software components and platforms: Overview and quality assessment,” ACM Comput. Surv., vol. 55, no. 8, dec 2022. [Online]. Available: https://doi.org/10.1145/3548679

[4] R. Pérez-Castillo, M. A. Serrano, and M. Piattini, “Software modernization to embrace quantum technology,” Advances in Engineering Software, vol. 151, p. 102933, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0965997820309790

[5] M. Piattini, G. Peterssen, R. Pérez-Castillo, J. L. Hevia, M. A. Serrano, G. Hernández, I. G. R. de Guzmán, C. A. Paradela, M. Polo, E. Murina, and others, “The talavera manifesto for quantum software engineering and programming,” in QANSWER, 2020, pp. 1–5. [Online]. Available: https://www.aquantum.es/wp-content/uploads/2020/03/Talavera_Manifesto.pdf

[6] M. Piattini and J. M. Murillo, “Quantum software engineering landscape and challenges,” in Quantum Software Engineering. Springer, 2022, pp. 25–38.

[7] F. Regazzoni, A. Fowler, and I. Polian, “Quantum era challenges for classical computers,” in Proceedings of the 18th International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation, ser. SAMOS ’18. New York, NY, USA: Association for Computing Machinery, 2018, p. 173–178. [Online]. Available: https://doi.org/10.1145/3229631.3264737

[8] A. D. Carleton, E. Harper, J. E. Robert, M. H. Klein, D. De Niz, E. Desautels, J. B. Goodenough, C. Holland, I. Ozkaya, and D. Schmidt, “Architecting the future of software engineering: A national agenda for software engineering research and development,” Software Engineering Institute, Carnegie Mellon University, Report, November 2021 2021. [Online]. Available: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=741193

[9] B. Weder, J. Barzen, F. Leymann, and D. Vietz, Quantum Software Development Lifecycle. Cham: Springer International Publishing, 2022, pp. 61–83.

[10] M. A. Akbar, A. A. Khan, and S. Rafi, “A systematic decision-making framework for tackling quantum software engineering challenges,” Automated Software Engineering, vol. 30, no. 2, pp. 1–44, nov 2023.

[11] R. Pérez-Castillo, M. A. Serrano, J. A. Cruz-Lemus, and M. Piattini, “Guidelines to use the incremental commitment spiral model for developing quantum-classical systems,” Quantum Information and Computation, vol. 24, no. 1&2, pp. 71–88, 2024. [Online]. Available: https://www.rintonpress.com/xxqic24/qic-24-12/0071-0088.pdf

[12] L. Pareto, “Extended abstract: requirements modeling within iterative, incremental processes,” in Proceedings. Second ACM and IEEE International Conference on Formal Methods and Models for Co-Design, 2005. MEMOCODE ’05., 2005, pp. 249–250.

[13] U. S. Shah, D. C. Jinwala, and S. J. Patel, “An excursion to software development life cycle models: An old to ever-growing models,” SIGSOFT Softw. Eng. Notes, vol. 41, no. 1, p. 1–6, feb 2016. [Online]. Available: https://doi.org/10.1145/2853073.2853080

[14] J. Marin, J. Hurtado, M. Bastarrica, and L. Silvestre, “Tailoring hybrid software processes in a medium-size

software company," in Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing, ser. SAC '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 1042–1050. [Online]. Available: https://doi.org/10.1145/3555776.3577709

[15] N. Dey, M. Ghosh, S. S. Kundu, and A. Chakrabarti, "Qdlc – the quantum development life cycle," ArXiv e-prints, oct 2020.

[16] G. J. Hernández González and C. A. Paradela, "Quantum Agile Development Framework," in Quality of Information and Communications Technology, M. Shepperd, F. Brito e Abreu, A. Rodrigues da Silva, and R. Pérez-Castillo, Eds. Cham: Springer International Publishing, 2020, pp. 284–291.

[17] P. Spoletini, "Towards quantum requirements engineering," in 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW). IEEE, 2023, pp. 04–05.

[18] M. R. El aoun, H. Li, F. Khomh, and M. Openja, "Understanding quantum software engineering challenges an empirical study on stack exchange forums and github issues," in 2021 IEEE International Conference on Software Maintenance and Evolution (ICSME), 2021, pp. 343– 354.

[19] J. Zhao, "Quantum software engineering: Landscapes and horizons," arXiv, 2020, ISBN: 2007.07047v2.

[20] T. Yue, S. Ali, and P. Arcaini, "Towards quantum software requirements engineering," in 2023 IEEE International Conference on Quantum Computing and Engineering (QCE), vol. 02, 2023, pp. 161–164. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10313750

[21] L. Saraiva, E. H. Haeusler, V. Costa, and M. Kalinowski, "Non-functional requirements for quantum programs," in 2nd Quantum Software Engineering and Technology Workshop (QSET), colocated with IEEE International Conference on Quantum Computing and Engineering (QCE21), Virtual Conference, Oct 19, 2021.

[22] M. Salm, J. Barzen, U. Breitenbücher, F. Leymann, B. Weder, and K. Wild, "The nisq analyzer: Automating the selection of quantum computers for quantum algorithms," in Service-Oriented Computing, S. Dustdar, Ed. Cham: Springer International Publishing, 2020, pp. 66–85.

[23] N. Chancellor, R. Cumming, and T. Thomas, "Toward a standardized methodology for constructing quantum computing use cases," 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2006.05846

[24] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, and J. M. Gambetta, "Validating quantum computers using randomized model circuits," Phys. Rev. A, vol. 100, p. 032328, Sep 2019. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.100.032328

[25] M. Suchara, J. Kubiatowicz, A. Faruque, F. T. Chong, C.-Y. Lai, and G. Paz, "Qure: The quantum resource estimator toolbox," in 2013 IEEE 31st International Conference on Computer Design (ICCD), 2013, pp. 419–426.

[26] R. Pérez-Castillo and M. Piattini, "Design of classical-quantum systems with uml," Computing, vol. 104, no. 11, pp. 2375–2403, 2022.

[27] J. W. Z. Lau, K. H. Lim, H. Shrotriya, and L. C. Kwek, "Nisq computing: where are we and where do we go?" AAPPS bulletin, vol. 32, no. 1, p. 27, 2022.

**Author contribution:**

1. Conception and design of the study
2. Data acquisition
3. Data analysis
4. Discussion of the results
5. Writing of the manuscript
6. Approval of the last version of the manuscript

HP has contributed to: 1, 2, 3, 4, 5 and 6.

LPY has contributed to: 1, 2, 3, 4, 5 and 6.

RPC has contributed to: 1, 2, 3, 4, 5 and 6.

JAHA has contributed to: 1, 2, 3, 4, 5 and 6.

LA has contributed to: 1, 2, 3, 4, 5 and 6.

AF has contributed to: 1, 2, 3, 4, 5 and 6.

**Acceptance Note:** This article was approved by the journal editors Dr. Rafael Sotelo and Mag. Ing. Fernando A. Hernández Gobertti.