Towards a hybrid classical–quantum software architecture derived from literature harmonization

Hacia una arquitectura de software híbrida clásico–cuántica derivada de la armonización de la literatura

Rumo a uma arquitetura de software híbrida clássico-quântica derivada da harmonização da literatura

Carlos Eduardo Orozco Garcés[1]
César Jesús Pardo Calvache[2]
Ricardo Pérez del Castillo[3]
Samuel Eduardo Sepúlveda Cuevas[4]

Received: March 1st, 2026
Approved: June 20th, 2026
Available: September 1st, 2026

How to cite this article:
C. E. Orozco Garcés, C. J. Pardo Calvache, R. Pérez del Castillo, and S. E. Sepúlveda Cuevas, "Towards a Hybrid Classical–Quantum Software Architecture Derived from Literature Harmonization," *Revista Ingeniería Solidaria,* vol. 22, no. 3, 2026. doi: https://doi.org/10.16925/2357-6014.2026.03.07

Review article. https://doi.org/10.16925/2357-6014.2026.03.07

[1] Faculty of Electronic Engineering and Telecommunications, Universidad del Cauca, Popayán, Colombia.
Email: carlosorozco@unicauca.edu.co
ORCID: https://orcid.org/0000-0003-2615-3294
CvLAC: https://bit.ly/2OlXNlx

[2] Faculty of Electronic Engineering and Telecommunications, Universidad del Cauca, Popayán, Colombia.
Email: cpardo@unicauca.edu.co
ORCID: https://orcid.org/0000-0002-6907-2905
CvLAC: https://bit.ly/422UAtr

[3] Department of Systems Engineering, Universidad de Castilla–La Mancha, Ciudad Real, Spain.
Email: ricardo.pdelcastillo@uclm.es
ORCID: https://orcid.org/0000-0002-9271-3184

[4] Faculty of Informatics, Universidad de La Frontera, Temuco, Chile.
Email: samuel.sepulveda@ufrontera.cl
ORCID: https://orcid.org/0000-0002-0369-7750

CC BY

Abstract

Introduction: This article is the result of doctoral research conducted in 2025 on quantum software architecture, addressing the transformative potential of quantum computing and the challenges associated with integrating quantum capabilities into software systems.

Problem: The lack of standardized architectural models for hybrid classical–quantum systems limits interoperability, reuse, and systematic software design.

Objective: This study proposes a reference architecture to support the design and evolution of hybrid classical–quantum software systems.

Methodology: The architecture was derived through a harmonization process.

Results: The proposed framework defines ten architectural elements organized across three dimensions: Conceptual, Organizational, and Quality.

Conclusion: The architecture provides a structured foundation for designing hybrid classical–quantum solutions.

Originality: The work consolidates dispersed knowledge into a unified framework for hybrid classical–quantum systems.

Limitations: The model is based on the currently limited body of literature and industrial experience, requiring future validation.

Keywords: Quantum Software Engineering; Quantum Software Architecture.

Resumen

Introducción: El presente artículo es resultado de una investigación doctoral desarrollada en 2025 sobre arquitectura de software cuántica, que aborda el potencial transformador de la computación cuántica y los desafíos de integrar sus capacidades en sistemas de software.

Problema: La ausencia de modelos arquitectónicos estandarizados para sistemas híbridos clásico–cuánticos limita la interoperabilidad, la reutilización y el diseño sistemático de soluciones de software.

Objetivo: Este estudio propone una arquitectura de referencia para apoyar el diseño y la evolución de sistemas de software híbridos clásico–cuánticos.

Metodología: La arquitectura fue derivada mediante un proceso de armonización que sintetiza enfoques académicos e industriales existentes.

Resultados: El marco propuesto define diez elementos arquitectónicos organizados en tres dimensiones: conceptual, organizacional y de calidad.

Conclusión: La arquitectura proporciona una base estructurada para el diseño y la evolución de soluciones híbridas clásico–cuánticas.

Originalidad: Este trabajo consolida conocimiento académico e industrial en un marco unificado para sistemas híbridos clásico–cuánticos.

Limitaciones: El modelo se fundamenta en un cuerpo de literatura y experiencia que requiere validación empírica futura.

Palabras clave: Ingeniería de software cuántica; Arquitectura de software cuántica.

Resumo

Introdução: Este artigo é resultado de uma pesquisa de doutorado desenvolvida em 2025 sobre arquitetura de software quântica, abordando o potencial transformador da computação quântica e os desafios associados à integração de capacidades quânticas em sistemas de software.

Problema: A ausência de modelos arquitetônicos padronizados para sistemas híbridos clássico-quânticos limita a interoperabilidade, a reutilização e o projeto sistemático de software.

Objetivo: Propor uma arquitetura de referência para apoiar o projeto e a evolução de sistemas de software híbridos clássico-quânticos.

Metodologia: A arquitetura foi derivada por meio de um processo de harmonização.

Resultados: O framework proposto define dez elementos arquitetônicos organizados em três dimensões: Conceitual, Organizacional e de Qualidade.

Conclusão: A arquitetura fornece uma base estruturada para o projeto de soluções híbridas clássico-quânticas.

Originalidade: O trabalho consolida conhecimentos dispersos em um framework unificado para sistemas híbridos clássico-quânticos.

Limitações: O modelo baseia-se no conjunto atualmente limitado de literatura e experiência industrial, sendo necessária validação futura.

Palavras-chave: Engenharia de software quântica; arquitetura de software quântica.

1. INTRODUCTION

Currently, the study of quantum computing, in both its theoretical and applied dimensions, has evolved substantially, enabling a deeper understanding of its implications for academia and the software industry. In particular, the implementation of quantum devices capable of solving problems directly related to the emerging needs of specific sectors within the software domain began during the last decade. This progress has been driven by initiatives from companies such as Google, IBM, and Microsoft [1], which have invested in the development of solutions based on the paradigm introduced by quantum computing. Consequently, it has been possible to advance in areas such as artificial intelligence, data science, and cybersecurity, among others [2].

Building upon these developments, the design of new technologies that integrate quantum computing with contemporary approaches has become a critical focus for the software industry. At the same time, the academic community, motivated by the need to establish a clear and structured knowledge foundation for this emerging field, has promoted research in the area known as Quantum Software Engineering [3]. This discipline aims to adapt the theoretical basis of Classical Software Engineering to the challenges and requirements imposed by the quantum paradigm. To this end, several studies have focused on defining a software development life cycle aligned with the characteristics of quantum computing [2], [4], including the documentation

of activities, tasks, roles, tools, and processes that arise in the quantum domain [5], [6], as well as proposing hybrid approaches that combine the advantages of classical and quantum paradigms.

From this perspective, academic and industrial efforts have begun to establish a roadmap [5] that bridges theory and practice, fostering the progressive incorporation of quantum technologies into current software development processes. This structured incorporation enables quantum advancements to be effectively embedded in software systems while addressing the conceptual and engineering challenges associated with this new computational paradigm. However, such integration requires architectural mechanisms that support the design and management of new systems, whether purely quantum or hybrid classical–quantum. Consequently, it becomes necessary to conduct a comprehensive analysis of the requirements and challenges that these paradigms pose from a software architecture perspective, which constitutes the core focus of this research.

In that sense, this study seeks to answer the following research question: How can a reference architecture be defined that integrates the principles of quantum computing into the design of hybrid software systems?

To address this question, the present work introduces the advances achieved in constructing such an architecture through a knowledge harmonization process. In contrast to previous proposals that focus on specific implementations, this study presents a harmonization-based approach aimed at unifying the fragmented knowledge in the field, resulting in the first reference architecture grounded in a systematic synthesis of the literature.

The remainder of this paper is structured as follows: Section 2 summarizes the findings of a Systematic Literature Review (SLR) on software architectures in quantum computing, conducted as a foundation for this research. Section 3 analyzes the identified gaps and introduces the guiding proposal of the study. Section 4 defines the research objectives and scope. Section 5 presents the harmonization process used to identify the key architectural elements that compose the proposed framework. Finally, Sections 6, 7, and 8 discuss the challenges, limitations, conclusions, and future research directions.

2. MATERIALS AND METHODS

To establish a solid and up-to-date knowledge base on software architecture from the perspective of quantum computing, a Systematic Literature Review (SLR) was conducted and published in [4]. The review revealed several critical areas of study,

including: (i) conducting exploratory studies on concrete solutions that apply software architecture concepts and techniques to the challenges of quantum computing; (ii) developing methodological solutions, such as models, reference frameworks, and processes, that promote the development of hybrid and quantum software systems; and (iii) analyzing and documenting key quality criteria for the design of hybrid software architectures. These aspects converge around two major areas of interest identified in the literature: hybrid architectures, which aim to establish the transition between classical computing and the emerging needs of quantum computing, and quantum architectures, which represent a completely new and evolving research domain.

As a result of the SLR, a total of 17 primary studies were identified, which are summarized in Table 1.

Table 1. Primary articles

Identifier	Study	Reference
S1	Software architecture for quantum computing systems	[7]
S2	Towards process centered architecting for quantum software systems	[8]
S3	Engineering Reliable Hybrid Quantum Software	[9]
S4	Selection of quantum computing architecture using a decision tree	[10]
S5	Quality attributes on quantum computing platforms	[11]
S6	Challenges and Opportunities in Quantum Software Architecture	[12]
S7	A serverless cloud integration for quantum computing	[13]
S8	Research on Quantum Computing Standard System Architecture	[14]
S9	Design of classical-quantum systems with UML	[15]
S10	Hybrid Quantum Algorithms and Quantum Software Frameworks	[16]
S11	Quantum computing platforms	[17]
S12	The Quantum software lifecycle	[18]
S13	Towards a distributed quantum computing ecosystem	[19]
S14	Classical to Quantum Software Migration Journey Begins: A Conceptual Readiness Model	[20]
S15	Software Architecture Challenges in Classical-Quantum Systems	[21]
S16	Architecture Decisions in Quantum Software Systems	[22]
S17	Patterns for Quantum Software Development	[23]

Source: own work

The detailed description of the search string, inclusion and exclusion criteria, and the methodological process applied in the Systematic Literature Review (SLR) was published in [4]. Table 2 presents a summary of the four main axes of analysis

identified during the review, and the complete methodological details can be consulted in that reference:

Table 2. General results from the SLR

No	Main axis	Description	Article
1	**Heterogeneity in Solutions**	The proposals in the field are fragmented and lack a unified conceptual framework, which hinders their adoption outside large corporations.	S6, S10, S14, S17
2	**Capital and Infrastructure Challenges**	Industrial interest in adopting quantum computing–based solutions has grown; however, the absence of standards forces companies to invest significant resources in developing their solutions from scratch.	S2, S6, S11, S14
3	**Definition of Architectural Approaches**	Several architectures (layered, cloud, serverless, among others) have been identified to facilitate interaction between classical and quantum systems; however, the lack of a standardized foundation leads to compatibility, interoperability, and quality issues.	S1, S3, S4, S7, S8, S9, S13, S15, S16
4	**Quality Criteria**	The quality criteria is divided into two categories: (i) classical non-functional requirements, and (ii) non-functional requirements in quantum computing.	S5, S11, S12

Source: own work
Abbreviation: No (Number).

To contrast the academic findings with industrial practice, the SLR was complemented by an analysis of case studies and real-world applications in companies. Table 3 presents a summary of the studies identified because of this exploration.

Table 3. Pilot studies and industry efforts

Study	Category	Contribution
Qiskit Runtime (IBM) – cloud hybrid service [24]	SDK	Enables the execution of hybrid workloads, sessions, iteration management, and execution modes for both classical and quantum tasks.
TensorFlow Quantum [25]	Study case	Binary image classification (Fashion-MNIST) using hybrid QNN/CNN models with TFQ and Cirq.
TensorFlow Quantum – hybrid ML [26]	SDK	Allows the construction of hybrid models, quantum data structures, integration with **Cirq**, automatic differentiation, among other capabilities.

Source: own work

On the other hand, Table 4 summarizes the reports released by companies involved in the development of quantum and hybrid software architectures.

Table 4. Technical reports published by software companies

Study	Contribution
Microsoft Azure Quantum [27]	Describes the **Azure Quantum hybrid architecture**, explaining how classical and quantum components are integrated, and includes practical examples in environments such as **VQE** and **QAOA**.
Microsoft / Quantum [28], [29]	Announces the **"Integrated Hybrid"** feature, which enables the combination of classical and quantum code running on the same hardware [11] [12].
Google Quantum AI [30]	Hybrid approach that combines **digital and analog simulation** to model physical magnetism systems.

Source: own work

In general terms, the SLR highlights the need to establish standards for the development of hybrid and quantum software architectures through common frameworks that facilitate their design and maintenance. This would enable the creation of efficient hybrid and quantum solutions with high levels of cohesion, addressing aspects such as (i) compatibility with classical systems by integrating both functional and non-functional elements from the two paradigms, and (ii) the progressive transition from classical to quantum systems through hybrid approaches, among others.

Likewise, the analysis of the literature shows that the development of software architectures in the quantum context follows two main directions: (i) the creation of purely quantum approaches focused on optimizing well-known methodologies through quantum algorithms and resources, and (ii) the definition of hybrid solutions that integrate quantum capabilities within existing classical frameworks. However, the diversity of approaches in this field has resulted in a fragmented landscape, where the transition between the classical and quantum paradigms faces multiple methodological challenges. To provide a concrete focus for the research presented in this study, the scope of the proposal is limited to the exploration of hybrid approaches.

This decision is justified because hybrid architectures represent a major current challenge aligned with the present technological reality of the industry. Unlike purely quantum solutions, which require specialized infrastructures, highly technical expertise, and environments still under active development, hybrid architectures enable a controlled transition from classical systems, significantly lowering the entry barriers for companies seeking to experiment with quantum technologies without completely restructuring their processes. Moreover, hybrid architectures promote compatibility and component reuse by allowing classical and quantum systems to coexist, thereby optimizing resource utilization, reducing operational costs, and accelerating development cycles, as indicated in technical reports published by companies such as IBM, Google, and Microsoft, which highlight latency reductions and resource optimization in hybrid environments [26]–[30]. Finally, these architectures provide a practical

short- and medium-term solution without compromising the possibility of an evolution toward fully quantum models in the future.

On the other hand, recent studies conducted by leading companies in the industry (see Tables 3 and 4) show that practical approaches already exist for solving problems through hybrid schemes in real-world contexts, such as combinatorial optimization and machine learning. These industrial developments demonstrate a growing interest in defining reusable hybrid patterns and best practices to guide the integration of classical and quantum components. However, these initiatives are often oriented toward proprietary environments or specific use cases, which limits their generalization. This highlights the need for a broader and more standardized solution that facilitates interoperability, reduces adoption barriers, and serves as a convergence point between academia and industry.

Furthermore, the findings from the SLR and the exploratory studies indicate that the need for differentiated architectures is not exclusive to the quantum domain, as clear parallels can be found in other areas such as High-Performance Computing (HPC). Specific communities related to HPC have faced similar challenges when integrating heterogeneous components into hybrid systems. For instance, research centers such as Oak Ridge National Laboratory [31], Leibniz Supercomputing Centre [32], Forschungszentrum Jülich [33], Barcelona Supercomputing Center [34], and RIKEN [35] have reported experiences in which the incorporation of new paradigms, such as GPUs, FPGAs, and specialized accelerators, required the definition of specific architectures to manage interoperability, data flows, and quality attributes. In this way, the challenges reported by active research groups in the field reinforce the relevance of establishing a general framework that facilitates the instantiation of architectures adapted to specific contexts.

Considering the above, this work proposes a solution focused on the definition of a hybrid software reference architecture, understood as a high-level abstraction that encapsulates reusable components across different contexts [36]. This architecture aims to establish a homogeneous framework that facilitates the integration of quantum technologies into software development while ensuring key attributes such as interoperability, consistency, and reliability, and simultaneously mitigating heterogeneity and fostering collaboration between the academic and industrial sectors.

In general, the development of the proposed solution is structured into four stages: (i) conducting a Systematic Literature Review (SLR) to identify the current state of knowledge regarding software architectures in the context of quantum computing; (ii) designing a reference architecture for hybrid enterprise systems, integrating principles from Quantum Software Engineering and the harmonized elements identified

in the review; (iii) evaluating the architecture through expert judgment based on heuristic assessment to detect critical aspects and improvement opportunities; and (iv) validating its applicability through hypothetical scenarios using a mixed evaluation approach. However, it is important to clarify that stages (iii) and (iv), which correspond to the evaluation of the proposal, are not part of this work. Therefore, the proposed architecture should be considered a preliminary result still under development. The evaluation and validation activities will be addressed in subsequent studies and are outlined as future work.

Furthermore, it is essential to specify the connection between this study and the previously conducted Systematic Literature Review (SLR). In particular, Objective OE1 was achieved in the SLR published in [4], where an overview of emerging trends in hybrid software architectures for quantum computing was consolidated and later updated to include new emerging proposals and reports from companies actively engaged in this evolving field. Consequently, the present article focuses on Objective OE2 through the integration and organization of the identified elements into coherent dimensions and components.

3. RESULTS

Based on the results obtained from the SLR, a reference software architecture is proposed. To advance toward this objective, a harmonization process was developed, understood as a methodological strategy aimed at addressing the terminological and methodological fragmentation identified in the literature, which hinders the comparison, integration, and reuse of existing proposals and constitutes a barrier to the long-term construction of adaptable solutions.

The harmonization process was applied with the purpose of unifying, grouping, and standardizing the concepts and approaches identified in the reviewed primary studies, integrating them into a homogeneous knowledge framework that facilitates the design of a reference architecture. To guide this process, a protocol based on qualitative analysis and thematic categorization techniques [37] was applied, through which the architectural elements considered relevant for the formal structuring of the reference architecture were identified. As a result of this process, a structured representation of the available knowledge was obtained, oriented toward the design of an architecture aligned with the current challenges of hybrid development. Harmonization was essential to establish a common language and to build a solid methodological foundation that enables the evolution of existing proposals into solutions applicable to industrial environments.

Harmonization method: The harmonization followed the approach proposed in [37], consisting of the following activities: (i) identification and analysis of models; (ii) terminology normalization; (iii) alignment of structures and processes; and (iv) validation of the harmonized structure. As a result of these activities, a total of four studies were identified that define the elements necessary to characterize a reference architecture, which are described in Table 5.

Table 5. Relevant studies identified for the homogenization

Number	Reference	Description
1	[38]	Proposes guidelines based on the international standards for the evolution of architectures.
2	[39]	Conducts a review of the conceptual elements for the representation of reference architectures.
3	[40]	Addresses software architecture from a practical perspective, establishing the relevance of quality attributes as a central axis of design.
4	[41]	Proposes a framework for the design of reference architectures, identifying elements such as the relationship between context, objectives, and formalization.

Source: own work

To address the heterogeneity identified in the SLR, 10 elements were defined from the studies described in Table 5. The glossary of these elements is presented in Table 6:

Table 6. Homogenization results

Identifier	Element	Description	Reference
E1	Architecture objective	The design of a software architecture can be oriented toward standardization.	[38], [41]
E2	Architecture context	Specifying the application environment of an architecture requires considering its characteristics from a business perspective, using standards such as TOGAF [42].	[41]
E3	Level of abstraction	Classifies the architecture as abstract, semi-concrete, or concrete	[40], [41]
E4	Glossary of terms and definitions	Establishes a standardized vocabulary through Architecture Description Languages (ADLs)	[38], [39], [40], [41]
E5	Architectural structures	These structures define how the system's functionalities are decomposed into modules or components and how they communicate.	[39]

(continúa)

(viene)

Identifier	Element	Description	Reference
E6	Quality attributes	Identifying and classifying non-functional requirements or quality criteria ensures that a solution effectively meets user needs.	[38], [39], [40], [41]
E7	Architectural styles and patterns	The use of architectural patterns and styles allows systems to be structured following best practices, providing proven solutions to common design problems.	[38], [40]
E8	Design patterns	Design patterns are reusable templates that address common design problems.	[39]
E9	Structural components	In software architecture, the definition of modules, subsystems, and interfaces helps organize a system coherently.	[39]
E10	Reusable elements	In software architecture, reusable components encapsulate common functionalities that can be integrated into systems within the same domain.	[39]

Source: own work

Comparison method: After applying the homogenization process, a comparison was carried out between a base model, composed of the elements described in Table 6, and the 17 studies identified in the SLR, following the protocol described in [43]. This protocol consists of two activities: (i) selecting a comparison model, and (ii) applying a matrix to contrast each study with the selected model. Table 7 presents an excerpt of the comparison between the base model and study [11] for element E1.

Table 7. Comparing stage results extract - Element E1

Comparison question: Which elements of the model are defined in [11]?

Goal: To determine which elements defined in the base model have a close relationship with the elements defined in [11].

Elements considered for comparison: Elements defined in **Table 6.**

Element	Does the study define the element?			Description
	Yes	No	Partially	
E1			X	The article does not explicitly define model with clearly goals.

Source: own work

Table 7 is divided into two parts: the first presents the comparison question, objective, and elements; the second evaluates whether each study addresses these elements fully, partially, or not at all. Based on this comparison between the base model and the primary studies, the general knowledge was integrated. Figure 1 presents a heat map summarizing these results, where the rows (S1–S17) correspond to the

studies and the columns (E1–E10) to the analyzed elements. The color coding reflects the degree of coverage: green (yes), yellow (partial), and red (no). In this work, *"partial"* indicates that the element is mentioned or addressed indirectly or incompletely.

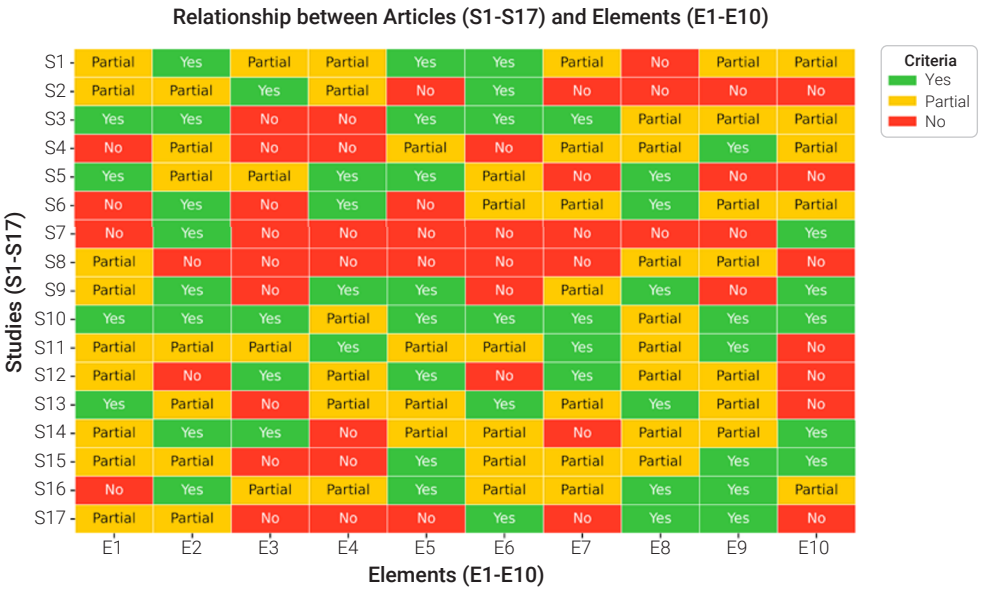

Fig 1. Base model vs RSL identified studies
Source: own work

According to the results shown in Figure 1, none of the primary studies reviewed address all the elements defined in the baseline model in an integrated manner, indicating a high degree of dispersion in the architectural treatment within the field of Quantum Software Engineering. When analyzing the data by element, it is observed that E2 and E6 exhibit the highest levels of coverage, being fully addressed by 47% and 41% of the studies, respectively. Conversely, elements E1, E3, E4, and E10 show significantly lower coverage levels, with percentages equal to or below 23%, suggesting a lack of systematic attention to aspects that could be relevant for the design of hybrid architectures. Finally, elements E5 and E7 present intermediate coverage, with 29% and 35% of the studies addressing them completely, respectively, while elements E8 and E9 show quite limited coverage, reaching only 17% and 29% of the studies. This reflects a weak integration of these components within the reviewed architectures, despite their potential importance in ensuring interoperability, traceability, and the quality of hybrid solutions.

As a result of this analysis, the comparison process allowed us to conclude that there is considerable fragmentation in the literature identified in the SLR, both in terms of the coverage of the defined architectural elements and the methodological focus

adopted by the primary studies. This lack of convergence hinders the development of reusable, interoperable, and coherent solutions.

Integration method: After completing the comparison, the next step was the integration phase, where the method proposed in [44] was applied to integrate the elements identified during the comparison. This method is divided into five activities: (i) designing the integration; (ii) defining and establishing the integration criteria; (iii) executing the integration; (iv) analyzing the results; and (v) presenting the integrated model. As a result of the integration process, a total of ten elements were identified as fundamental for the design of a reference architecture. The integration, based on the protocol proposed in [44], aimed to ensure four main aspects, which are described in Table 8.

Table 8. Integration criteria

Criteria	Description
Relevance	Ensures that the selected elements are aligned with the objectives and scope of the study.
Coherence	Verifies that the concepts and definitions are compatible with each other, avoiding contradictions in the integrated model.
Generalization	Promotes the abstraction of the identified elements to guarantee their applicability across different contexts.
Consistency	Evaluates whether the information extracted from the different studies maintains uniformity in terms of terminology, categories, and relationships.
Completeness	Ensures that the elements reflect the available knowledge, incorporating fundamental aspects for the construction of the reference architecture.

Source: own work

Additionally, a complementary activity was carried out in which the integrated elements were grouped and classified into specific dimensions (see Table 9). In the context of a reference architecture, defining dimensions allows for organizing knowledge into structural axes that cluster conceptually related elements, facilitating their interpretation, communication, and application. This classification serves to establish a semantic and conceptual order among the elements identified during the homogenization. Likewise, defining a clear set of dimensions aims to align with the general structure of the ISO international standards, which states that an architecture must be described through multiple views aligned with the stakeholders' concerns [45]. Based on this perspective, the study seeks to ensure traceability among the integrated elements, structure the architecture based on internationally recognized principles [46], [47], [48], [49].

Table 9. Integration method results for dimensions

Dimension	Description	Elements
Conceptual (DC)	Define the purpose and scope of the reference architecture.	E1, E2, E3, E4
Organizational (DO)	Establish the components and relationships among the elements that compose the architecture.	E5, E9, E10
Design and Quality(DCD)	Define the principles and guidelines that ensure the quality of the architecture.	E6, E7, E8

Source: own work

For this study, the dimensions emerged from a qualitative analysis process guided by the protocol proposed in [50], which consists of two stages: (i) open coding, aimed at identifying conceptual similarities among the elements, and (ii) axial coding, focused on establishing relationships and constructing common categories.

Open coding: Aims to thoroughly examine the data without prior assumptions, identifying recurring concepts, ideas, or terms that reveal relevant thematic relationships. This stage allows the information to be fragmented into meaningful units and labeled with initial codes, thereby facilitating the detection of similarities, redundancies, or divergences among them.

Axial coding: Seeks to reorganize and regroup the codes obtained in the previous phase by establishing connections between them through more abstract categories or analytical dimensions. This stage involves identifying causal relationships among the concepts, which enables the construction of a logical structure that articulates the different elements within a common framework.

The application of these techniques is particularly useful because it allows transforming a dispersed set of concepts into an organized and coherent structure. Open coding ensures that the findings from the literature are analyzed without initial bias, guaranteeing that no relevant patterns are overlooked, while axial coding facilitates the integration of those patterns into higher-level categories, establishing clear relationships among the elements. Together, this approach helps reduce terminological and methodological fragmentation, generating a conceptual framework that serves as the foundation for defining a reference architecture applicable to classical–quantum hybrid scenarios.

Illustrative use case (logistic optimization with QAOA): The scenario described below was constructed following the scenario-based evaluation technique [40], which aims to facilitate the analysis of architectures during the early stages of development. This method enables a preliminary validation of the applicability and coherence of an architecture by defining representative usage contexts and mapping them to the proposed elements and dimensions. In this study, the route optimization

case using QAOA is employed as an illustrative scenario that demonstrates how the components of the hybrid architecture address integration needs between classical and quantum systems. Figure 2 summarizes this workflow, coordinated by a hybrid orchestrator that integrates classical and quantum modules.

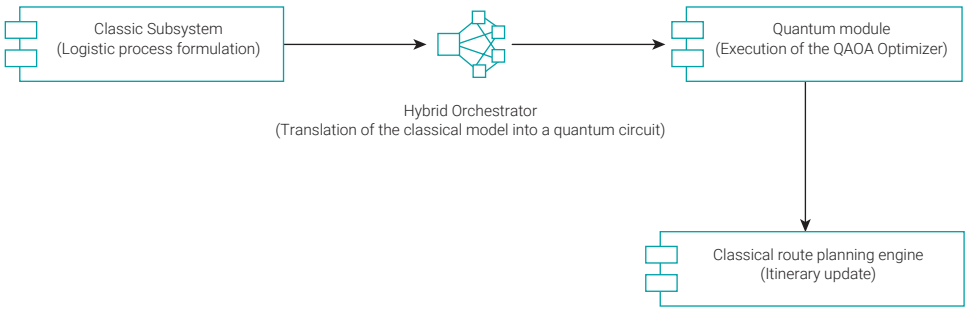

Fig 2. Case scenario - route planning
Source: own work.

According to Figure 2, the process integrates classical and quantum subsystems through a hybrid orchestrator that coordinates the interaction between both paradigms.

Classical Subsystem (Problem Formulation): Responsible for modeling the route optimization problem as a mathematical instance, for example, a Max-Cut or Vehicle Routing Problem. This module uses classical algorithms to generate the initial representation of the problem in a format that can be interpreted by the orchestrator.

Hybrid Orchestrator: Acts as a bridge between the classical and quantum worlds. Its main function is to translate the problem formulation into a quantum circuit compatible with the selected backend. It also manages bidirectional communication, including the preparation of initial parameters and the collection of results.

Quantum Module (QAOA Execution): Receives the translated instance and executes the Quantum Approximate Optimization Algorithm (QAOA). This module can run on real quantum hardware or on general-purpose simulators, depending on availability. During execution, critical aspects related to quality criteria are considered to ensure that the results are reliable.

Classical Route Planning Engine: Integrates the results returned by the quantum module and updates the proposed routes in the transportation system. In this way, quantum information is incorporated into the classical decision-making cycle, providing optimized solutions ready to be applied in the business context.

Based on the structure defined for the problem, the next step consists of mapping the processes of the scenario to the three dimensions (DC, DO, and DCD) and

the ten elements (E1–E10) defined during the harmonization phase, aiming to identify inconsistencies or gaps that help refine the model. To achieve this, a qualitative correspondence exercise was conducted, in which each module of the logistics optimization flow (classical subsystem, hybrid orchestrator, quantum module, and classical route planning engine) was compared against the identified architectural elements. This analysis allowed the verification of component coverage in relation to objectives, structures, and quality attributes, as well as the identification of potential gaps where the theoretical definition has not yet been fully translated into an applied case. The correspondence was established following a scenario-based evaluation approach [40], which facilitates linking architectural proposals with representative usage contexts. In this way, the logistics optimization case becomes a preliminary validation scenario that illustrates how the harmonized elements can guide the design of hybrid classical–quantum solutions. Table 10 presents a summary of the results after cross-referencing the information described in the problem domain with elements E1–E10.

Table 10. Correspondence between elements and the domain

Dimension	Element	Correspondence
Conceptual	E1	Objective: interoperability in route optimization.
	E2	Context: transportation/logistics environment with existing transportation management systems.
	E3	Semi-concrete level: definition of modules without specifying hardware.
Organizational	E4	Glossary: problem instance, QAOA, hybrid pipeline, orchestrator.
	E5	Structures: classical subsystem, orchestrator, quantum module, classical engine.
	E9	Components: APIs, SDK drivers, results repository.
	E10	Reusable elements: optimization library applicable to other domains.
Quality and Design	E6	Attributes: scalability, quantum reliability, security.
	E7	Styles: hybrid orchestrator, data pipeline.
	E8	Patterns: Factory, Adapter, Proxy, Facade.

Source: own work

A mapping is marked as "covered" when the module explicitly implements the element, for example, the orchestrator materializes elements E5 and E9 through well-defined interfaces; "partial" when the element does not yet have a complete operational definition, for instance, element E8 is defined at a conceptual level but still requires the specification of concrete design patterns at the implementation level; or "pending" when the theoretical definition has not yet been translated into measurable artifacts, for example, the elements identified for E6 (fidelity/coherence) in the quantum module

require further analysis to ensure the existence of calibration functions to measure compliance.

This highlights the need for a future research line focused on defining a quantifiable quality model for hybrid systems, one that translates metrics such as fidelity and coherence into architectural-level performance indicators.

Finally, the correspondence analysis facilitates the definition of a traceability table, which allows determining the degree of adoption of each element defined in the proposal with respect to the modules described in Figure 2, as shown in Table 11.

Table 11. Traceability Matrix

Element	Description
E1	**Classical subsystem:** Interoperates with the optimizer; minimizes route cost (DC). **Hybrid orchestrator:** Ensures classical quantum coupling and controlled latency (DC). **Quantum module (QAOA):** Achieves the optimization objective using the agreed ansatz/optimizer (DC). **Classical planning engine:** Integrates the solution into the TMS while meeting the business objective (DC).
E2	**Classical subsystem:** Logistics/TMS domain; operational constraints/SLAs (DC). **Hybrid orchestrator:** Orchestration rules according to policies/environment (DC). **Quantum module:** Provider backend/quota/noise basins (DC).
E3	**Classical subsystem:** Semi-concrete model (no fixed backend) (DC). **Hybrid orchestrator:** Generic contracts with adapters (DC). **Quantum module:** Parameterizable circuits—hardware or simulator (DC).
E4	**Classical subsystem:** Defines "instance," "pipeline," "QAOA," "route" (DO). **Hybrid orchestrator:** Defines "job," "backend," "session," "callback" (DO). **Quantum module:** Defines "ansatz," "fidelity," "shots" (DO).
E5	**Classical subsystem:** Data modeling/normalization module (DO). **Hybrid orchestrator:** ETL-H (classical→quantum→classical) flow and queues (DO). **Quantum module:** QAOA execution service and parameter manager (DO). **Planning engine:** Route planning and update service (DO).
E6	**Classical subsystem:** Efficiency, data security, auditability (DCD). **Hybrid orchestrator:** Availability, latency, retry, observability (DCD). **Quantum module:** Fidelity, error mitigation, coherence T1/T2 (DCD).
E7	**Classical subsystem:** Data pipeline; modeling microservice (DCD). **Hybrid orchestrator:** Hybrid orchestrator, event-driven (DCD). **Quantum module:** Specialized service; batch/session/hybrid execution (DCD).
E8	**Classical subsystem:** Strategy/Builder for cost functions (DCD). **Hybrid orchestrator:** Adapter/Facade for SDKs; Circuit Breaker (DCD). **Quantum module:** Factory for ansätze/optimizers; Template Method (DCD).
E9	**Classical subsystem:** Instance creation API; DTOs/validator (DO). **Hybrid orchestrator:** SDK adapters (Qiskit/Cirq/PennyLane); queues (DO). **Quantum module:** Executors; fidelity meter; results repository (DO).
E10	**Classical subsystem:** Modeling templates (VRP/Max-Cut) (DO). **Hybrid orchestrator:** Reusable multi-backend drivers (DO). **Quantum module:** Libraries of ansätze/optimizers (DO).

Source: own work

Results analysis: The correspondence exercise made it possible to verify the extent to which the ten elements (E1–E10) and the three dimensions (DC, DO, DCD) cover the processes defined in the logistics optimization case. In general terms, it

was observed that the modules in the workflow (classical subsystem, hybrid orchestrator, quantum module, and classical planning engine) are consistently aligned with the proposed conceptual and organizational structure. This finding suggests that the reference architecture can guide the design of hybrid solutions in practical scenarios while maintaining the methodological coherence established during the harmonization process.

In addition, the analysis revealed critical gaps that define the future research agenda. Quantum quality attributes (E6) require the creation of formal metrics and evaluation thresholds to allow comparison with classical criteria in real scenarios. Likewise, the integration of design patterns (E8) must evolve from a declarative level to a concrete implementation catalog, and the definition of reusable elements (E10) requires a higher degree of formalization to ensure portability. However, the mapping functions as an initial approximation consistent with the preliminary definition of the reference architecture, demonstrating how the elements and dimensions project onto a representative use case.

Nevertheless, this represents only the first step in the maturation process of the proposal. The next challenge lies in breaking down each element into concrete instances so that they can be represented through specific architectural components, software tools, and technologies applied to the problem context. This transition from the conceptual to the implementable will allow the robustness of the model to be validated and advance toward hybrid solutions with a higher degree of applicability in real-world environments.

4. DISCUSSION AND CONCLUSIONS

The correspondence exercise made it possible to verify the extent to which the ten elements (E1–E10) and the three dimensions (DC, DO, DCD) cover the processes defined in the logistics optimization case. In general terms, it was observed that the modules in the workflow (classical subsystem, hybrid orchestrator, quantum module, and classical planning engine) are consistently aligned with the proposed conceptual and organizational structure. This finding suggests that the reference architecture can guide the design of hybrid solutions in practical scenarios while maintaining the methodological coherence established during the harmonization process.

In addition, the analysis revealed critical gaps that define the future research agenda. Quantum quality attributes (E6) require the creation of formal metrics and evaluation thresholds to allow comparison with classical criteria in real scenarios. Likewise, the integration of design patterns (E8) must evolve from a declarative level

to a concrete implementation catalog, and the definition of reusable elements (E10) requires a higher degree of formalization to ensure portability. However, the mapping functions as an initial approximation consistent with the preliminary definition of the reference architecture, demonstrating how the elements and dimensions project onto a representative use case.

Nevertheless, this represents only the first step in the maturation process of the proposal. The next challenge lies in breaking down each element into concrete instances so that they can be represented through specific architectural components, software tools, and technologies applied to the problem context. This transition from the conceptual to the implementable will allow the robustness of the model to be validated.

Considering the above, this study made it possible to identify and consolidate a set of ten relevant elements for the design of a hybrid reference architecture, integrating classical and quantum approaches through a Systematic Literature Review (SLR) [4]. The methodological process included the harmonization, comparison, and integration of proposals found in the literature, characterized by their dispersion and varying levels of formalization.

Based on the obtained results, the elements were organized into three structured dimensions, enabling a clear and functional representation of the harmonized knowledge. This classification enhances the understanding and applicability of the model across different contexts by facilitating the identification and delineation of classical and quantum components according to the problem requirements. The proposed organization thus provides a guide for both the analysis and design of hybrid architectures.

One of the main contributions of this work lies in the distillation of a heterogeneous knowledge field, whose fragmentation was evidenced in the initial SLR, into a unified conceptual framework. This allows overcoming the conceptual dispersion observed in the literature and moving toward an integrative perspective. As a result, the harmonization process clarifies points of convergence among different approaches and supports architectural decision-making from a more unified standpoint.

Finally, it is expected that the elements and dimensions identified in this work will serve as inputs for the formal definition of the reference model, facilitating the clear identification of classical and quantum components within each architectural dimension. To achieve this, the model will provide a common foundation for describing and analyzing the integration of technologies, tools, and patterns at different levels of abstraction, ensuring that each element has well-defined implementation criteria.

As future work, the next steps of the proposal will focus on: (i) defining the reference architecture in detail, with special emphasis on the formalization of quality metrics (E6) and patterns (E8 and E10); (ii) conducting an initial expert evaluation aimed at gathering feedback on the conceptual soundness and applicability of the model; and (iii) carrying out a scenario-based evaluation by designing representative cases to analyze the architecture's adaptability to different contexts and requirements.

Additionally, this study opens the door to complementary research lines that extend the scope of the proposal, including the development of a methodological framework for implementation by exploring strategies that facilitate adoption in industrial and academic environments, and the examination of applicable standards and regulatory frameworks, with the goal of aligning the architecture with international standards and strengthening interoperability.

REFERENCES

[1] E. Gibney, "Hello quantum world! Google publishes landmark quantum supremacy claim," *Nature*, vol. 574, no. 7779, pp. 461–462, Oct. 2019, doi: https://doi.org/10.1038/d41586-019-03213-z.

[2] M. Piattini, G. Peterssen, and R. Pérez-Castillo, "Quantum computing: A new software engineering golden age," *ACM SIGSOFT Software Engineering Notes*, vol. 45, no. 3, pp. 12–14, Jul. 2020, doi: https://doi.org/10.1145/3402127.3402131.

[3] M. A. Serrano, R. Pérez-Castillo, and M. Piattini, Eds., *Quantum Software Engineering*. Cham, Switzerland: Springer Nature Switzerland, 2022, doi: https://doi.org/10.1007/978-3-031-05324-5.

[4] C. E. Orozco and C. J. Pardo Calvache, "Tendencias emergentes y desafíos de las arquitecturas de software en la computación cuántica: Una revisión sistemática de la literatura," *Revista Científica*, vol. 51, no. 3, pp. 87–111, 2024, doi: https://doi.org/10.14483/23448350.22793.

[5] J. M. Murillo *et al*., "Quantum software engineering: Roadmap and challenges ahead," *ACM Transactions on Software Engineering and Methodology*, vol. 34, no. 5, Art. no. 154, 2025, doi: https://doi.org/10.1145/3712002.

[6] M. Piattini and J. M. Murillo, "Quantum software engineering landscape and challenges," in *Quantum Software Engineering*, M. A. Serrano, R. Pérez-Castillo, and M. Piattini, Eds. Cham, Switzerland: Springer Nature Switzerland, 2022, pp. 25–38, doi: https://doi.org/10.1007/978-3-031-05324-5_2.

[7] A. A. Khan *et al*., "Software architecture for quantum computing systems: A systematic review," *arXiv*, Art. no. 2202.05505, Feb. 2022, doi: https://doi.org/10.48550/arXiv.2202.05505.

[8] A. Ahmad, A. A. Khan, M. Waseem, M. Fahmideh, and T. Mikkonen, "Towards process centered architecting for quantum software systems," in *Proc. IEEE International Conference on Quantum Software (QSW)*, Barcelona, Spain, 2022, pp. 26–31, doi: https://doi.org/10.1109/QSW55613.2022.00019.

[9] M. Scheerer, J. Klamroth, and O. Denninger, "Engineering reliable hybrid quantum software: An architectural-driven approach," in *Proc. 2nd International Workshop on Quantum Software Engineering and Technology (Q-SET), co-located with IEEE Quantum Week*, Broomfield, CO, USA, Oct. 2021, pp. 29–37. [Online]. Available: https://ceur-ws.org/Vol-3008/short3.pdf

[10] L. Nallamothula, "Selection of quantum computing architecture using a decision tree approach," in *Proc. 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, Tirunelveli, India, Dec. 2020, pp. 644–649, doi: https://doi.org/10.1109/ICISS49785.2020.9315893.

[11] B. Sodhi, "Quality attributes on quantum computing platforms," *arXiv*, Art. no. 1803.07407, Mar. 2018, doi: https://doi.org/10.48550/arXiv.1803.07407.

[12] T. Yue, W. Mauerer, S. Ali, and D. Taibi, "Challenges and opportunities in quantum software architecture," in *Software Architecture: Research Roadmaps from the Community*, P. Pelliccione, R. Kazman, I. Weber, and A. Liu, Eds. Cham, Switzerland: Springer Nature Switzerland, 2023, pp. 1–23, doi: https://doi.org/10.1007/978-3-031-36847-9_1.

[13] M. Grossi *et al*., "A serverless cloud integration for quantum computing," *arXiv*, Art. no. 2107.02007, Jul. 2021, doi: https://doi.org/10.48550/arXiv.2107.02007.

[14] H. Yang, J. Wang, and X. Sun, "Research on quantum computing standard system architecture and roadmap," *Journal of Physics: Conference Series*, vol. 2433, no. 1, Art. no. 012035, 2023, doi: https://doi.org/10.1088/1742-6596/2433/1/012035.

[15] R. Pérez-Castillo and M. Piattini, "Design of classical-quantum systems with UML," *Computing*, vol. 104, no. 11, pp. 2375–2403, 2022, doi: https://doi.org/10.1007/s00607-022-01091-4.

[16] R. Ramouthar and H. Seker, "Hybrid quantum algorithms and quantum software development frameworks," *ScienceOpen Preprints*, Aug. 2023, doi: https://doi.org/10.14293/PR2199.000298.v1.

[17] B. Sodhi and R. Kapur, "Quantum computing platforms: Assessing the impact on quality attributes and SDLC activities," in *Proc. IEEE 18th International Conference on Software Architecture (ICSA)*, Stuttgart, Germany, 2021, pp. 80–91, doi: https://doi.org/10.1109/ICSA51549.2021.00016.

[18] B. Weder, J. Barzen, F. Leymann, M. Salm, and D. Vietz, "The quantum software lifecycle," in *Proc. 1st ACM SIGSOFT International Workshop on Architectures and Paradigms for Engineering Quantum Software (APEQS 2020)*, 2020, pp. 2–9, doi: https://doi.org/10.1145/3412451.3428497.

[19] D. Cuomo, M. Caleffi, and A. S. Cacciapuoti, "Towards a distributed quantum computing ecosystem," *IET Quantum Communication*, vol. 1, no. 1, pp. 3–8, 2020, doi: https://doi.org/10.1049/iet-qtc.2020.0002.

[20] M. A. Akbar, S. Rafi, and A. A. Khan, "Classical to quantum software migration journey begins: A conceptual readiness model," in *Proc. International Conference on Product-Focused Software Process Improvement*, ser. *Lecture Notes in Computer Science*, vol. 13709, Cham, Switzerland: Springer Nature Switzerland, 2022, pp. 563–573, doi: https://doi.org/10.1007/978-3-031-21388-5_42.

[21] V. Stirbu and T. Mikkonen, "Software architecture challenges in integrating hybrid classical-quantum systems," in *Proc. IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 2, Bellevue, WA, USA, Sep. 2023, pp. 203–204, doi: https://doi.org/10.1109/QCE57702.2023.10313643.

[22] M. S. Aktar *et al*., "Architecture decisions in quantum software systems: An empirical study on Stack Exchange and GitHub," *Information and Software Technology*, vol. 177, Art. no. 107587, Jan. 2025, doi: https://doi.org/10.1016/j.infsof.2024.107587.

[23] F. Bühler, J. Barzen, M. Beisel, D. Georg, F. Leymann, and K. Wild, "Patterns for quantum software development," in *Proc. 15th International Conference on Pervasive Patterns and Applications (PATTERNS 2023)*, Barcelona, Spain, Jun. 2023, pp. 30–39, ISBN 978-1-68558-049-0. [Online]. Available: https://www.iaas.uni-stuttgart.de/publications/Buehler2023_PatternsQuantumSE.pdf

[24] IBM Quantum, "Introduction to Qiskit," *IBM Quantum Platform*. [Online]. Available: https://quantum.cloud.ibm.com/docs/en/guides. [Accessed: Apr. 13, 2026].

[25] S. S. Kadam, A. C. Adamuthe, and A. B. Patil, "CNN model for image classification on MNIST and Fashion-MNIST datasets," *Journal of Scientific Research*, vol. 64, no. 2, pp. 374–384, 2020, doi: https://doi.org/10.37398/jsr.2020.640251.

[26] TensorFlow, "TensorFlow Quantum: A library for hybrid quantum-classical machine learning," Google, 2020. [Online]. Available: https://www.tensorflow.org/quantum. [Accessed: Apr. 13, 2026].

[27] Microsoft, "What is hybrid quantum computing?," *Microsoft Learn*. [Online]. Available: https://learn.microsoft.com/en-us/azure/quantum/hybrid-computing-overview. [Accessed: Apr. 13, 2026].

[28] Microsoft, "How Microsoft and Quantinuum achieved reliable quantum computing," *Microsoft Azure Blog*, Apr. 3, 2024. [Online]. Available: https://azure.microsoft.com/en-us/blog/quantum/2024/04/03/how-microsoft-and-quantinuum-achieved-reliable-quantum-computing. [Accessed: Apr. 13, 2026].

[29] Microsoft, "Azure Quantum unlocks the next generation of hybrid quantum computing," *Microsoft Quantum Blog*. [Online]. Available: https://quantum.microsoft.com/en-us/insights/blogs/qir/azure-quantum-unlocks-the-next-generation-of-hybrid-quantum-computing. [Accessed: Apr. 13, 2026].

[30] Google, "A new hybrid platform for quantum simulation of magnetism," *Google Research Blog*, Apr. 21, 2025. [Online]. Available: https://research.google/blog/a-new-hybrid-platform-for-quantum-simulation-of-magnetism. [Accessed: Apr. 13, 2026].

[31] A. D. Barker, D. E. Bernholdt, and A. S. Bland, "High-performance computing facility operational assessment 2015: Oak Ridge Leadership Computing Facility," Oak Ridge National Laboratory, Oak Ridge, TN, USA, Tech. Rep. ORNL/SPR-2016/110, 2016. [Online]. Available: https://www.osti.gov/biblio/1324094

[32] D. Krause, "JUWELS: Modular tier-0/1 supercomputer at the Jülich Supercomputing Centre," *Journal of Large-Scale Research Facilities*, vol. 5, Art. no. A135, 2019, doi: https://doi.org/10.17815/jlsrf-5-171.

[33] J. Dongarra *et al*., "The international exascale software project roadmap," *International Journal of High Performance Computing Applications*, vol. 25, no. 1, pp. 3–60, Feb. 2011, doi: https://doi.org/10.1177/1094342010391989.

[34] I. Iglesias Álvarez, "MareNostrum 5 conquista los 314.000 billones de cálculos por segundo," *Computerworld España*, Dec. 21, 2023. [Online]. Available: https://www.computerworld.es/article/2100534/marenostrum-5-conquista-los-314-000-billones-de-calculos-por-segundo.html. [Accessed: Apr. 13, 2026].

[35] M. Sato, Y. Kodama, M. Tsuji, and T. Odajima, "Co-design and system for the supercomputer 'Fugaku'," *IEEE Micro*, vol. 42, no. 2, pp. 26–34, Mar.–Apr. 2022, doi: https://doi.org/10.1109/MM.2021.3136882.

[36] H. Cervantes, P. Velasco-Elizondo, and L. Castro, *Arquitectura de Software: Conceptos y Ciclo de Desarrollo*. Mexico City, Mexico: Cengage Learning, 2016. [Online]. Available: https://books.google.com/books/about/Arquitectura_de_software.html?id=2Za7vQEACAAJ

[37] C. Pardo, F. J. Pino, F. García, M. Piattini, and M. T. Baldassarre, "A process for driving the harmonization of models," in *Proc. 11th International Conference on Product-Focused Software Process Improvement (PROFES 2010)—Short Papers, Doctoral Symposium, and Workshops*, Limerick, Ireland, Jun. 2010, pp. 51–54, doi: https://doi.org/10.1145/1961258.1961271.

[38] E. S. Palma, E. Y. Nakagawa, D. M. B. Paiva, and M. I. Cagnin, "Evolving reference architecture description: Guidelines based on ISO/IEC/IEEE 42010," *arXiv*, Art. no. 2209.14714, Sep. 2022, doi: https://doi.org/10.48550/arXiv.2209.14714.

[39] M. A. Sánchez Palmero, N. Silega Martínez, and O. Y. Rojas Grass, "Revisión de elementos conceptuales para la representación de las arquitecturas de referencia de software," *Revista Cubana de Ciencias Informáticas*, vol. 13, no. 1, pp. 143–157, 2019. [Online]. Available: https://www.redalyc.org/journal/3783/378360617011/html/

[40] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*, 3rd ed. Upper Saddle River, NJ, USA: Addison-Wesley, 2012. [Online]. Available: https://www.informit.com/store/software-architecture-in-practice-9780321815736

[41] S. Angelov, P. Grefen, and D. Greefhorst, "A framework for analysis and design of software reference architectures," *Information and Software Technology*, vol. 54, no. 4, pp. 417–431, Apr. 2012, doi: https://doi.org/10.1016/j.infsof.2011.11.009.

[42] A. Josey, *TOGAF® Version 9.1: A Pocket Guide*. Zaltbommel, The Netherlands: Van Haren Publishing and The Open Group, 2016. [Online]. Available: https://publications.opengroup.org/g117

[43] F. J. Pino, M. T. Baldassarre, M. Piattini, and G. Visaggio, "Harmonizing maturity levels from CMMI-DEV and ISO/IEC 15504," *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 22, no. 4, pp. 279–296, Jun. 2010, doi: https://doi.org/10.1002/spip.437.

[44] C. Pardo, F. García, M. Piattini, F. J. Pino, S. Lemus, and M. T. Baldassarre, "Integrating multiple models for definition of IT governance model for banking ITGSM," *International Business Management*, vol. 10, no. 19, pp. 4644–4652, 2016, doi: https://doi.org/10.36478/ibm.2016.4644.4652.

[45] ISO/IEC/IEEE, *ISO/IEC/IEEE 42010:2022, Systems and Software Engineering—Architecture Description*. Geneva, Switzerland: International Organization for Standardization, Nov. 2022. [Online]. Available: https://www.iso.org/standard/74393.html

[46] F. Amaro and L. Domingues, "PMBOK 6th meets 7th: How to link both guides in order to support project tailoring?," *Procedia Computer Science*, vol. 219, pp. 1877–1884, 2023, doi: https://doi.org/10.1016/j.procs.2023.01.486.

[47] M. Geiger, S. Harrer, J. Lenhard, and G. Wirtz, "BPMN 2.0: The state of support and implementation," *Future Generation Computer Systems*, vol. 80, pp. 250–262, Mar. 2018, doi: https://doi.org/10.1016/j.future.2017.01.006.

[48] J. A. Zachman, "A framework for information systems architecture," *IBM Systems Journal*, vol. 26, no. 3, pp. 276–292, 1987, doi: https://doi.org/10.1147/sj.263.0276.

[49] H. Jonkers, E. Proper, and M. Turner, "TOGAF and ArchiMate: A future together," The Open Group, Reading, U.K., White Paper W192, Nov. 2009. [Online]. Available: https://publications.opengroup.org/w192

[50] J. Corbin and A. Strauss, *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 3rd ed. Thousand Oaks, CA, USA: SAGE Publications, 2008, doi: https://doi.org/10.4135/9781452230153. [Online]. Available: https://methods.sagepub.com/book/mono/basics-of-qualitative-research/