



The Foundations of Quantum Computing and Their Relation to Software Engineering

¹ Reyes Juárez-Ramírez, ORCID: 0000-0002-5825-2433 <reyesjua@uabc.edu.mx>

¹ Christian X. Navarro, ORCID: 0000-0002-7220-7006 <cnavarro@uabc.edu.mx>

¹ Samantha Jiménez, ORCID: 0000-0003-0938-7291 <samantha.jimenez@uabc.edu.mx>

² Alan Ramírez, ORCID: 0000-0002-8634-9988 <alandramireznoriega@uas.edu.mx>

³ Verónica Tapia-Ibarra, ORCID: 0000-0002-0501-8600 <veronica.tapia@leon.tecnm.mx>

⁴ César Guerra-García, ORCID: 0000-0002-9290-6170 <cesar.guerra@uaslp.mx>

⁴ Hector G. Perez-Gonzalez, ORCID: 0000-0003-3331-2230 <hectorgerardo@uaslp.mx>

⁵ Carlos Fernández-y-Fernández, ORCID: 0000-0002-1586-8772 <caff@mixteco.utm.mx>

¹ Universidad Autónoma de Baja California, Tijuana, Baja California, México.

² Universidad Autónoma de Sinaloa, Sinaloa, México.

³ Instituto Tecnológico de León, León, Guanajuato, México.

⁴ Universidad Autónoma de San Luis Potosí, San Luis Potosí, SLP, México.

⁵ Universidad Tecnológica de la Mixteca, Huajuapán de León, Oaxaca, México.

Abstract. The principles of quantum mechanics – superposition, entanglement, measurement, and decoherence – form the foundation of quantum computing. Qubits, which are abstract objects having a mathematical expression to implement the rules of quantum physics, are the fundamental building blocks of computation. Software is a key component of quantum computing, along with quantum hardware. Algorithms make up software, and they are implemented using logic gates and quantum circuits. These qualities make quantum computing a paradigm that non-physicists find difficult to comprehend. It is crucial to incorporate a conceptual framework of the principles upon which quantum computing is founded into this new method of creating software. In this paper, we present a kind of taxonomical view of the fundamental concepts of quantum computing and the derived concepts that integrate the emerging discipline of quantum software engineering. Because the systematic review's main goal is to identify the core ideas behind quantum computing and quantum software, we conducted a quasi-systematic mapping as part of the review process. The findings can serve as a starting point for computer science teachers and students to address the study of this field.

Keywords: Quantum computing; quantum principles; qubits; software engineering; taxonomy.

For citation: Juárez-Ramírez R., Navarro C.X., Jiménez S., Ramírez A., Tapia-Ibarra V., Guerra-García C., Perez-Gonzalez H. G., Fernández-y-Fernández C. The Foundations of Quantum Computing and Their Relation to Software Engineering. *Trudy ISP RAN/Proc. ISP RAS*, vol. 36, issue 1, 2024. pp. 73-104. DOI: 10.15514/ISPRAS-2024-36(1)-6.

Full text: R. Juárez-Ramírez, C. X. Navarro, Samantha Jiménez, Alan Ramírez, Verónica Tapia-Ibarra, César Guerra-García, Hector G. Perez-Gonzalez, and Carlos Fernández-y-Fernández. A Taxonomic View of the Fundamental Concepts of Quantum Computing – A Software Engineering Perspective. *Programming and Computer Software*, 2023, Vol. 49, No. 8, pp. 682–704. DOI: 10.1134/S0361768823080108.

Acknowledgements. We would like to thank Universidad de Guanajuato, División de Ciencias e Ingeniería, for their support during the sabbatical stage. Also, thanks to Centro de Investigación en Matemáticas for their support in providing access to electronic databases. Finally, we would like to thank Universidad Autónoma de Baja California, for financing this research through the project 3908_300/6/C/63/23 “Computación Cuántica: Implicaciones en la Ingeniería de Software y las Competencias del Ingeniero de Software”, 23ª Convocatoria

Interna de Proyectos de Investigación. Also, thanks to the team Red Mexicana de Ingeniería de Software who participate in this Project and literature review.

Основы квантовых вычислений и их связь с разработкой программного обеспечения

¹ Рейес Хуарес-Рамирес, ORCID: 0000-0002-5825-2433 <reyesjua@uabc.edu.mx>

¹ Чристиан К. Наварро, ORCID: 0000-0002-7220-7006 <cnavarro@uabc.edu.mx>

¹ Саманта Хименес, ORCID: 0000-0003-0938-7291 <samantha.jimenez@uabc.edu.mx>

² Алан Рамирес, ORCID: 0000-0002-8634-9988 <alandramireznoriega@uas.edu.mx>

³ Вероника Тапия-Ибарра, ORCID: 0000-0002-0501-8600 <veronica.tapia@leon.tecnm.mx>

⁴ Сесар Герра-Гарсия, ORCID: 0000-0002-9290-6170 <cesar.guerra@uaslp.mx>

⁴ Эктор Х. Перес-Гонсалес, ORCID: 0000-0003-3331-2230 <hectorgerardo@uaslp.mx>

⁵ Карлос Фернандес-и-Фернандес, ORCID: 0000-0002-1586-8772 <caff@mixteco.utm.mx>

¹ Автономный университет Нижней Калифорнии, Тихуана, Нижняя Калифорния, Мексика.

² Автономный университет Синалоа, Синалоа, Мексика.

³ Технологический институт Леона, Леон, Гуанахуато, Мексика.

⁴ Автономный университет Сан-Луис-Потоси, Сан-Луис-Потоси, СЛП, Мексика.

⁵ Технологический университет Миштека, Уахуапан де Леон, Оахака, Мексика.

Аннотация. Принципы квантовой механики – суперпозиция, запутанность, измерение и декогерентность – формируют основу квантовых вычислений. Кубиты, которые являются абстрактными объектами, представляемые математическими выражениями, моделирующими законы квантовой физики, являются фундаментальными строительными блоками вычислений. Программное обеспечение является, наряду с квантовым оборудованием, ключевым компонентом квантовых вычислений. Программы строятся на основе алгоритмов, которые реализуются с помощью логических вентилей и квантовых схем. Эти качества делают квантовые вычисления парадигмой, которую трудно понять тем, кто не имеет специального физического образования. Для этой новой парадигмы очень важно выработать концептуальные основы, главные принципы. В статье представлен таксономический взгляд на фундаментальные концепции квантовых вычислений и другие концепции, которые объединяют возникающую дисциплину квантовой программной инженерии. Основной целью систематического обзора является выявление основных идей, лежащих в основе квантовых вычислений и квантового программного обеспечения, в рамках сделанного обзора мы провели квазисистематическое картирование. Результаты могут служить отправной точкой для учителей информатики и студентов при изучении этой предметной области.

Ключевые слова: квантовые вычисления; квантовые законы; кубиты; разработка программного обеспечения; таксономия.

Для цитирования: Хуарес-Рамирес Р., Наварро-Кота Ч.К., Хименес С., Рамирес А., Тапия-Ибарра В., Герра-Гарсия С., Перес-Гонсалес Э.Х., Фернандес-и-Фернандес К. Основы квантовых вычислений и их связь с разработкой программного обеспечения. Труды ИСП РАН, том. 36, вып. 1, 2024. стр. 73-104 (на английском языке). DOI: 10.15514/ISPRAS-2024-36(1)-6.

Полный текст: Хуарес-Рамирес Р., Наварро-Кота Ч.К., Хименес С., Рамирес А., Тапия-Ибарра В., Герра-Гарсия С., Перес-Гонсалес Э.Х., Фернандес-и-Фернандес К. Таксономический взгляд на фундаментальные концепции квантовых вычислений: перспективы разработки программного обеспечения. *Programming and Computer Software*, 2023, т. 49, №. 8, стр. 682–704 (на английском языке). DOI: 10.1134/S0361768823080108.

Благодарности. Мы благодарны Отделению науки и техники Университета Гуанахуато за предоставление творческого отпуска. Также благодарим Центр математических исследований,

предоставивший доступ к электронным базам данных. Наконец, мы благодарим Автономный университет Нижней Калифорнии за финансирование этого исследования в рамках проектов 3908_300/6/C/63/23 «Квантовые вычисления: последствия для программной инженерии и компетенции инженера-программиста», 23^a «Привлечение учащихся к исследовательским проектам», также команду Мексиканской сети программной инженерии, которая участвует в этом проекте и обзоре литературы.

1. Introduction

Quantum computing involves information processing tasks, which are implemented using quantum mechanical systems [1]. To process, store, and transfer the quantum information set represented by qubits, quantum computers are based on quantum mechanics phenomena such as quantum superposition and entanglement. [2]. Quantum computing can significantly cut both execution time and energy usage when compared to conventional digital computing [3]; it sounds simple and obvious, but is it? In terms of hardware, the construction of quantum systems is still facing challenges [4-5]. In terms of software, it is based on the mathematical concepts of quantum mechanics phenomena [6]; this fact makes its implementation more complicated. In this fashion, the authors of [7] argued that, for academicians and practitioners, there is an emergent nature of quantum computer research and an increasing need for interdisciplinarity to address the identified challenges. In general terms, quantum computing is a multidisciplinary field that brings together aspects of physics, mathematics, and computer science [5] and uses quantum mechanics to solve complex problems faster than classical computers [8]. A supporting definition appears in [9]: “*Quantum computing is a fascinating new field at the intersection of computer science, mathematics, and physics, which strives to harness some of the uncanny aspects of quantum mechanics to broaden our computational horizons*”.

Let us introduce the fundamental concepts involved.

The qubit is an abstract object, with a mathematical expression, so it is a mathematical object with certain specific properties [1]; it is a quantum system with two basic states, $|0\rangle$ and $|1\rangle$ [8] manipulated arbitrarily, which are well distinguishable by physical measurements. In contrast to classic binary digits represented as (0,1), qubit as the most fundamental unit of the quantum information set attains a state that is a superposition of 0 and 1 and is represented as $|0\rangle$ and $|1\rangle$ [10-11] where:

$$\begin{aligned} |0\rangle &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ |1\rangle &= \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{aligned}$$

A pure qubit state is a coherent superposition of the basis states. This means that a single qubit (ψ) is a linear combination of $|0\rangle$ and $|1\rangle$ as follows:

$$|\psi\rangle = |\alpha|^2 + |\beta|^2 \quad (1)$$

This gives rise to new logic gates that make new algorithms possible. A quantum gate is a basic quantum circuit that operates on a small set of qubits [8]. The quantum gates allow the implementation of quantum mechanics principles [1], such as superposition and entanglement. A quantum gate is simply an operator that acts on qubits. Unitary matrices will be used to represent these operators [9]. For example, two essential quantum gates are Hadamard (One-qubit gate) gate and CNOT (Multi-qubit gate) gate [8]. Hadamard gate allows the superposition of a qubit, while the CNOT gate allows the entanglement of multiple qubits [12]. This is their matrix representation:

$$\text{Hadamard} = H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$\text{CNOT} = CX = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

A quantum program or algorithm is implemented through a quantum circuit, which is formed by a set of quantum gates. A complete quantum program is expressed in a considerable number of circuits.

Based on this explanation, quantum programming can be seen as the process of assembling sequences of instructions, called quantum circuits, which can run on a quantum computer.

As we can see, the quantum programming model is fundamentally different from traditional computer programming. It is observed in the quantum computing labor market that most positions require Ph.D. degrees due to the elevated level of expertise required by quantum principles.

There is a rapidly growing demand for a quantum workforce [13] educated in the basics of quantum computing, particularly in quantum programming [14].

Therefore, it is necessary to provide a more “intuitive” way to think and write quantum algorithms, thereby simplifying the design and implementation of quantum software [11].

Quantum computing education is ramping up, however, now, there are few computer science specialists to teach these topics, so there are few offerings for non-specialists and little information on best practices for training computer science and engineering students [14].

In [15], the authors suggest bridging the gap between computer professionals and non-physicists by offering the conceptual and notational information that segregates quantum computing from conventional computing. In [16], the authors stated that interest in building dedicated quantum information science and engineering (QISE) education programs have greatly expanded in recent years and that there will be a need for a wide variety of expertise and education levels to create a balanced technical workforce like that seen in other professional scientific and engineering fields.

To contribute to the efforts in making quantum computing literacy more understandable, in this paper, we present a taxonomic view of the fundamental concepts involved in quantum computing, as an initial proposal to integrate a body of knowledge (BOK) for quantum software engineering. We focused on the set of concepts (as the first element of a BOK), which are extracted from the literature, emphasizing the quantum mechanics fundamentals, and the related knowledge of mathematics and computer science. To find the fundamental concepts we performed a quasi-systematic mapping, with ingredients of systematic literature review and multivocal literature review to include grey literature.

The rest of the paper is organized as follows. Section 2 contains a background describing how a body of knowledge is integrated, emphasizing the set of fundamental concepts. Section 3 contains related work, describing proposals of taxonomies in the context of quantum computing, and education initiatives for quantum computing as well as skills required for quantum computing. Section 4 contains the methodology used, describing how the systematic mapping was done. Section 5 contains the results, presenting the taxonomical view of the fundamental concepts of quantum computing and the emerging concepts of quantum software engineering. Section 6 contains a discussion. Finally, section 7 presents the conclusions and future work.

2. Background: Body of knowledge

There are various BOK definitions in the literature. In this section, we present some of them.

A body of knowledge is the complete set of concepts, terms, and activities that make up a professional domain, as defined by the relevant learned society or professional association [17]. The body of knowledge is “generally recognized” by practitioners and may be codified in a variety of ways for a variety of different uses [18].

In [19], the following definition is given for a body of knowledge: (1) “Structured knowledge that is used by members of a discipline to guide their practice or work”; and (2) “The prescribed aggregation of knowledge in a particular area an individual is expected to have mastered to be considered or certified as a practitioner”.

We are going to work with this idea: “A *body of knowledge* is a set of knowledge within a profession or subject area which is agreed as both essential and known” [17]. A body of knowledge is the accepted ontology for a specific domain. Furthermore, a BOK also is considered “the systematic collection of activities and outcomes in terms of their values, constructs, models, principles, and instantiations, which arises from continuous discovery and validation work by members of the profession and enables self-reflective growth and reproduction of the profession” [20].

Integrating a BOK is a challenging task. There are many ways of doing it. The authors of [21] cited some methods: *developing taxonomies, engaging communities of practice, and constructing framing metaphors*.

As we mentioned before, a BOK is integrated by various elements, however, a basic element is the set of fundamental concepts of the referred discipline, so that, this paper is focused on the set of fundamental concepts of quantum computing.

3. Related work

Quantum Ontologies and Quantum Body of Knowledge. In [22], a review of quantum computing literature is presented. The authors proposed a taxonomy of quantum computing, which is used to map various related studies to identify the research gaps. A main taxonomy is presented emphasizing quantum computing technology, which has two specialized branches: (1) time and gates characteristics and (2) algorithmic characteristics. A kind second level of taxonomy is presented for three branches: (1) Software Applications in Quantum Computing, (2) Quantum Annealing-based Software Components, (3) Quantum Software Life Cycle and Associated Terminologies.

The authors did not describe explicitly the systematic approach used to perform the literature review. Furthermore, the proposed taxonomy is not presented in a hierarchical view.

In [15], well-described definitions of quantum mechanics fundamentals are presented, such as superposition, entanglement, and decoherence, as well as the qubit as the fundamental unit of computation. Also, the authors presented a summary of the most used quantum algorithms, quantum technologies, and software tools. A taxonomy of one level is presented, characterizing the uses cases of quantum computing, having two branches: (1) emotions and sentiment analysis, error correction, quantum internet, quantum materials, cryptography, post-quantum cryptography, drug discovery, genetic programming, and navigation; (2) image processing, cloud computing, weather prediction, energy management, open-source software, transport engineering, machine learning, chemistry, and finance.

In the paper, the conceptual part of quantum mechanics fundamentals is well described, as well as the use cases. However, the taxonomical view is presented only at the level of use cases, not at the level of fundamental and supporting quantum computing concepts.

Teaching quantum computing. Quantum computing is a difficult field for non-experts to understand [23-24]. Additionally, most methods for implementing quantum computing are restricted to literature or software implementation.

The authors of [25] propose a Bloch sphere interactive system to visualize quantum computing simulation. This report describes a variety of programming assignments that can be used to teach quantum computing in a practical manner, experiencing most of the software development phases.

In [26], the authors present the results of two semesters of a new undergraduate course on Quantum Computing for Fundamental Sciences and Engineering students. The course was taught employing

a blended learning approach, with a combination of synchronous classes, asynchronous video lessons, and projects using IBM's Qiskit framework.

In both cases [25-26], the authors did not present a complete set of fundamental concepts of quantum computing supported by quantum mechanics.

Skills required for quantum software development. The rapid rise of interest and investments in quantum information science and engineering has led to increasing demand for a quantum-trained workforce [27]. Recent assessments of the needs of the quantum industry [16, 28] identified quantum software engineering and applications development as essential skills required for certain types of industry jobs [27]. Surveys of the university programs offering Master-level education in QISE [16] show that quantum programming is frequently included as either a standalone course or one of the topics in an introductory course [27].

Motivation from related work. This related work gives us a general view of efforts that are focused on trying to organize the knowledge around quantum computing; it is a motivation for us. Especially its limitations detected in terms of a taxonomical view of the fundamental concepts of quantum computing, encourage us to perform the current research work presented in this paper.

4. Methodology

To perform this review, we considered the recommendations for a systematic mapping (SM) from [29-31]. (The review performed is based on SM). Also, we considered recommendations for systematic literature reviews (SLR) from [32-37], especially for establishing selection criteria.

In terms of approaches for carrying out the review and extending the reach of the search, both methods are complementary. Sometimes additional sources are well accepted to enrich the coverture of the review [36], including manual and less structured searches of the Internet and other sources [38] and grey literature as is recommended in [39].

In [32] a comparison is presented, emphasizing the difference in breadth and depth of SM and SLR: *In a systematic mapping study, more articles can be considered as they do not have to be evaluated in such detail.* Therefore, a larger field can be structured. This can be reflected in the search string and inclusion criteria [29]. The search string can be more generic, and the inclusion criteria can be less restrictive [31].

Because the review's goal is to locate the fundamental ideas of quantum computing and quantum software engineering rather than to thoroughly examine the state of the art of each field or the solutions offered to the problems or difficulties that these disciplines face, this ability of SM allows us to perform a systematic mapping.

We decided to follow the recommendations of SLR, especially in terms of the first level of selection criteria, without quality assessment criteria.

There are several formal reports about quantum computing, however, a structured search may not include previous work with significant contributions to our objective. In an unstructured search, we identified work with significant contributions not included in the formal search, so we decided to review these references.

4.1 Definition of Research Questions (Outcome: Research Scope)

Software is an essential element for computation [40]. Quantum software applications are getting popular because the power of quantum computing facilitates the application of this paradigm to solve complex problems in any field of science and the real world [41-44]. Quantum software plays a critical role in exploiting the full potential of quantum computing systems [11]. Quantum applications require the use of a completely different kind of computer and algorithms, which have the potential to solve tasks that we do not even dare dream of today [45]. Considering the significance of software in the quantum world, we state the main objective of the literature review.

The objective of the literature review: To look for what is the impact of quantum computing on software. How do the fundamental concepts of quantum computing and its implementation participate in quantum software development? Furthermore, we intend to organize those fundamental concepts in a kind of taxonomical way for a better understanding of the multidisciplinary approach of quantum computing.

Research questions: Four research questions were formulated:

RQ1: What is the definition of Quantum Computing?

RQ2: What are the fundamental concepts of Quantum Computing?

RQ3: What are the supporting concepts for implementing Quantum Computing?

RQ4: What are the new Software Engineering concepts emerging from Quantum Computing?

Searching phrases: We decided to formulate a search phrase that includes quantum computing as the main topic and its relationship with software engineering.

The search phrase is the following:

“Quantum Computing” AND “Software Engineering”

Databases considered: We considered looking for papers in three databases: ACM, IEEE Xplore, and ScienceDirect.

Developing review protocol: In a session with the participation of two researchers, we defined a kind of general protocol, considering the following aspects: What type/source of papers to consider? What parts of the papers should be revised? How many reviewers are going to review the same set of papers? Defining the filters for selecting the papers, and formats for gathering information.

4.2 Conduct Search for Primary Studies (Outcome: All Papers)

Identifying the relevant research: We tried with a user account from a Mexican institution, with the proper privileges to access the advanced search section on the website of databases and to obtain the corresponding source files of the papers. The result of the search is shown in Table 1. The three databases provide more than 100 items from the search phrase.

Table 1. Results from the search

Database	Number of papers (items) found
ACM (AC)	201
ScienceDirect (SD)	199
IEEE Xplore (XP)	140

4.3 Screening of Papers for Inclusion and Exclusion (Outcome: Relevant Papers)

Selecting the primary studies: We defined a structure for identifying the resulting documents: demographic aspects, inclusion criteria, and exclusion criteria.

Demographics: This attribute identifies the nature of the document.

D1: Type of document (Research paper, communication paper, white paper). We decided to select research papers, unless there may be other types of work with significant contribution.

D2: Origin of the document (Conference, Journal, Book, other.) We decided to select papers from journals and conferences, unless there may be other types of work with significant contributions.

D3: Language – (Results that are written in English/Spanish). We decided to consider only work written in English.

D4: Accessibility – Full texts are accessible by means of institutional accounts.

INCLUSION CRITERIA: We established inclusion criteria on two levels, in the form of filters.

Level 1: Header of the paper

F1: The title of the paper. Contains one significant word of a phrase (e.g., quantum, software), one part of the search phrase, or two parts.

F2: The keyword section. Contains one significant word of a phrase (e.g., quantum, software), one part of the search phrase, or two parts.

F3: Abstract section. Contains an established relationship between “quantum computing” and “software engineering,” for example:

- There are concepts related to one or both areas (“quantum computing,” “software engineering”), explicitly cited.
- A relationship between “quantum computing” and “software engineering” is established.

Note: F3 was the filter with the highest acceptance value because we noticed that in some cases, the title and keywords section did not contain the expected elements, however, the abstract gave signs that the paper contains contribution.

Level 2: Body of the paper

B1: Results that introduce and describe concepts of QC and/or quantum software engineering.

B2: Results that discuss Quantum Computing and some aspects of software engineering, or vice versa.

EXCLUSION CRITERIA: Exclusion criteria are oriented to removing items that we consider do not provide relevant information to the research or do not contain complete information or are not available.

EX01: Remove the duplicates found in the databases.

EX02: Remove items that are not research papers. (Unless there may be other types of work with a significant contribution.)

EX03: Remove papers if only the abstract but not the full text is available.

EX04: Remove results not written in English.

EX05: Results that do not introduce and describe the concepts of quantum computing and/or quantum software engineering.

EX06: Results that do not discuss the association between quantum computing and/or quantum software engineering.

Results of selection: We applied the exclusion criteria EX01, EX02, EX03, and EX04, as well as the filters of level 1, having the selection expressed in Table 2. Appendix A labels the papers consecutively numbered as S1, S2, ...Sn.

Table 2. Results from the selection: Filters Level 1 and exclusion criteria

Database	Papers selected
IEEE Xplore	87
ACM	43
ScienceDirect	13

As we mentioned earlier, we considered all the papers in this selection stage, trying to consider a wider spectrum of literature to detect the fundamental concepts. Furthermore, we include additional sources to complete the report.

4.4. Keywording of Abstracts (Outcome: Classification Scheme)

According to the objective of the research, we created four categories for classifying the selected papers: (1) Quantum computing basics and tendencies, (2) Combination of software engineering and quantum computing, (3) Education in quantum computing, and (4) Skills for quantum computing.

Category 1 is expected to include papers containing descriptions of quantum computing fundamentals and topics related to tendencies in research in quantum computing and quantum technology development.

Category 2 is expected to include papers that emphasize the connection between quantum computing and software engineering, describing advances in the quantum-oriented approach of software engineering concepts.

Category 3 is expected to include papers that explicitly contain proposals or studies for quantum computing or quantum software engineering education.

Category 4 is expected to include papers that explicitly emphasize the required skills and competencies for quantum computing and related topics, such as quantum hardware construction and quantum programming.

4.5. Data Extraction and Mapping of Studies (Outcome: Systematic Map)

In Table 3 we present the grouping of papers in terms of the four categories indicated.

As we can see, Category 1 has the highest frequency of papers, followed by Category 2. As we mentioned in the introductory sections, education in quantum computing is not much attended (Category 3), nor the skills required for quantum computing (Category 4).

The results from the analysis of the mapping are presented in the next section, answering the research questions.

5. Results

In this section we present the results, answering the research questions. For some questions we considered the papers resulting from the search; for other questions, we considered complimentary literature which provides significant contributions to our research project.

RQ1: What is the definition of Quantum Computing?

These definitions were extracted from the reviewed literature and supporting literature. We chose those papers with definitions [46-50] that involve several aspects of quantum computing. Three definitions are presented next.

Definition 1 [46]: “Quantum Computing is a paradigm that intersects computer science, mathematics, and physics. Unlike other computing fields, quantum computing uses the law of quantum mechanics with the goal of achieving high computation efficiency.” The focus of Quantum Computing is the issue of storing, handling and transmitting data stored in quantum mechanical systems. This data mode is therefore referred to as quantum information known as Qubit. Mathematically, a qubit may be denoted with the help of a vector $|\psi\rangle$ in the two-dimensional complex vector area which has a related inner product, such that $|\psi\rangle \in H_2$ [47].

Definition 2: Quantum computing is referred to as “the field of science which directly uses quantum mechanical phenomena like superposition and entanglement to perform operations on data” [48]. Quantum computing studies quantum computers with the quantum mechanics’ phenomenon of superposition, entanglement, tunneling, and annealing to solve problems that cannot be solved in the life span of human beings [49].

Definition 3: Quantum computing works with abstract units named quantum bits, so the basic storage unit is the quantum bit (Qubit) [50]. Quantum bits, or qubits, are considered like quantum particles. The manipulation of qubits by control devices is at the core of a quantum computer’s processing power.

RQ2: What are the fundamental concepts of Quantum Computing?

To answer this question, we present a kind of taxonomy at distinct levels.

For this part of our research, we implemented three steps of the process to build a taxonomy: (1) Determine requirements, (2) identify the concepts, and (3) develop a first version of the taxonomy. The remaining steps are proposed as part of the future work. We considered the following criteria: (a) Terms should be unambiguous and clear, yet not too wordy and long; (b) hierarchical relationships between concepts are of type generic-specific, whole-part, instance; and (c) the structure, for the main concepts' components, depth is about three or four levels, except for entities representing processes or subfields.

The first level integrates the four main supporting disciplines of quantum computing, which are: physics, computer science, mathematics, and electronics. See Fig. 1.

In the second level of taxonomy, we include subfields involved. See Fig. 2-4.

Quantum mechanics is the subfield of physics, which provides the essential principles that quantum computing is based on, and it is strongly related to mathematics. Four subfields of mathematics give support to quantum computing (see Fig. 2): basic logic, linear algebra, numbers theory, and probability. From linear algebra, vectors, matrixes, and operations with them are the core of qubits representation and processing, at the level of qubits and quantum gates. From numbers theory, complex numbers are used to try with the coefficients of the terms of superposed states of qubits.

Table 3. Grouping of papers into categories

Category	Related papers
1.Quantum computing basics and tendencies	S4, S8, S9, S10, S12, S13, S20, S21, S22, S27, S28, S29, S31, S33, S34, S37, S38, S41, S42, S43, S44, S46, S47, S48, S51, S52, S53, S54, S55, S56, S57, S58, S61, S62, S65, S67, S68, S69, S70, S71, S72, S73, S74, S75, S76, S77, S78, S79, S80, S81, S83, S84, S85, S86, S87, S88, S89, S90, S92, S92, S93, S94, S95, S98, S99, S101, S102, S103, S104, S105, S106, S107, S108, S109, S110, S111, S112, S113, S114, S115, S116, S117, S118, S119, S122, S123, S124, S125, S126, S127, S128, S129, S130, S131, S132, S133, S134, S135, S136, S137, S138, S139, S140
2. Software engineering and quantum computing	S1, S2, S5, S6, S7, S9, S14, S15, S17, S19, S19, S23, S24, S26, S30, S32, S35, S36, S39, S40, S45, S49, S50, S56, S59, S60, S63, S64, S66, S82, S91, S96, S97, S100, S120, S121
3. Education in quantum computing	S16, S25, S55
4. Skills for quantum computing	S3

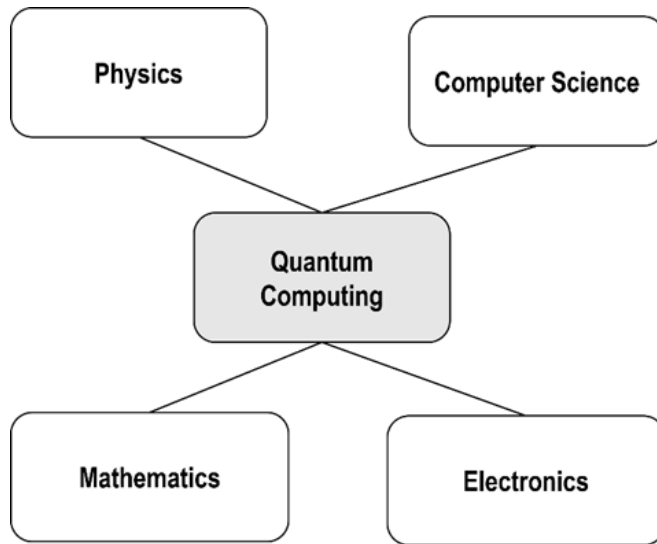


Fig. 1. Disciplines involved in quantum computing

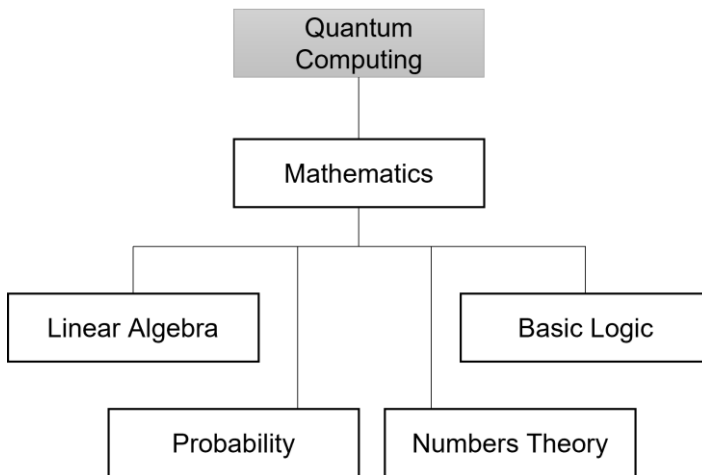


Fig. 2. Areas of mathematics involved in quantum computing

In the case of computer science (see Fig. 3), for the practical implementations of quantum computation, these concepts are involved: computer architecture, programming languages, programming fundamentals, and algorithms. Computer architecture concepts and elements are needed to configure the real expression quantum computation on hardware.

Programming language elements such as syntax, semantics, and capabilities are especially important for the human conceptualization of quantum computing and to indicate computations to do by quantum computers.

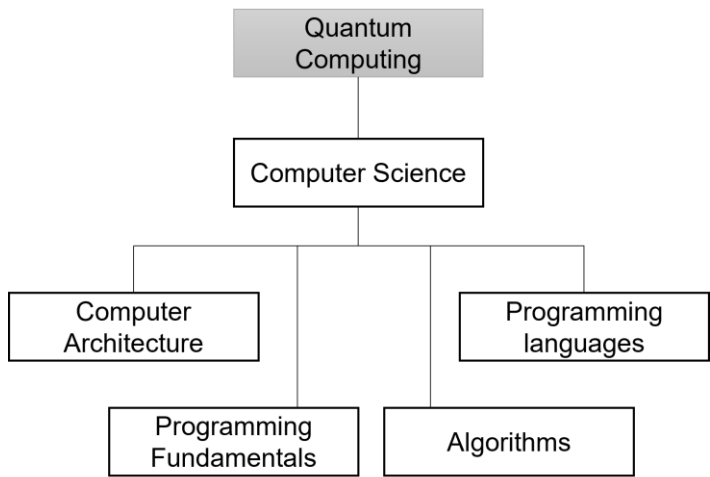


Fig. 3. Areas of computer science involved in quantum computing

Programming fundamentals are the core elements for indicating the logic and the flow of computations, processing information, as well as for visioning the target solution. Algorithms in quantum computing are important tools because they encompass quantum logic and qubits transformations needed to perform quantum computation.

In the case of basic electronics, it gives support for figuring out the practical implementation of quantum computing. Quantum logic gates and circuits allow to implement complete quantum programming and algorithms (see Fig. 4).

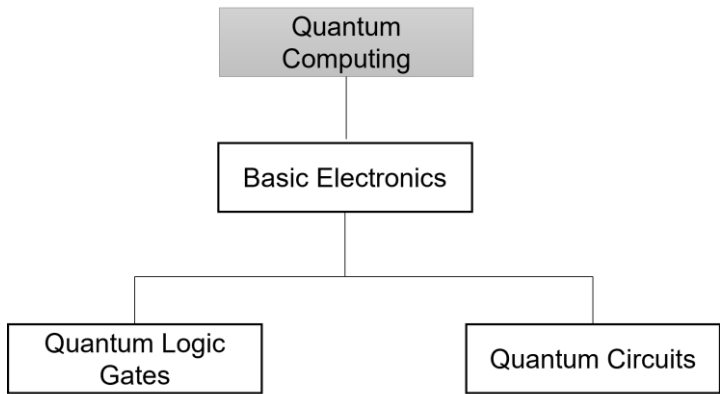


Fig. 4. Areas of electronics involved in quantum computing.

In a third level of taxonomy, we introduce the main concepts of quantum mechanics: Superposition, entanglement, uncertainty principle, probability principle, measurement, decoherence, and non-locality. See Fig. 5.

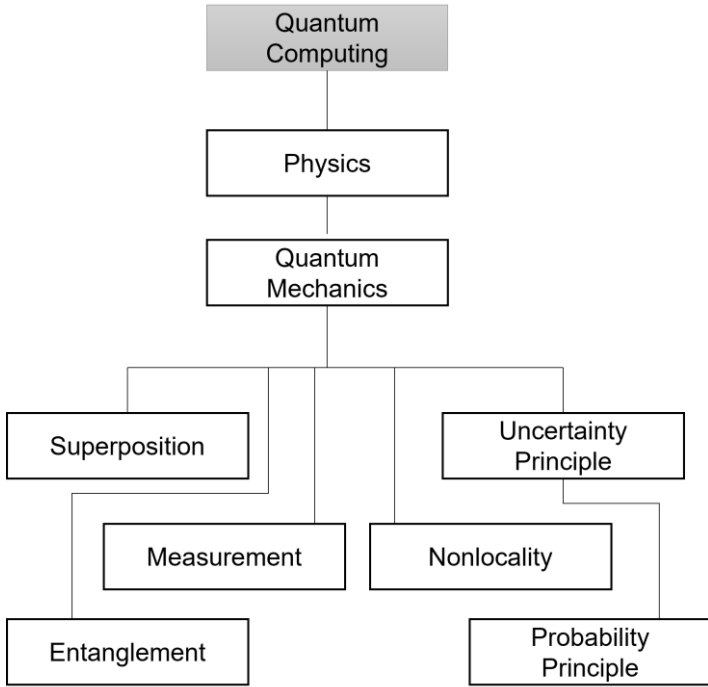


Figure 5. Concepts of quantum mechanics involved in quantum computing.

Superposition: In the quantum mechanics context, the superposition principle is the idea that a system (or an atomic particle) is in all possible states at the same time, until it is measured [51]; after measurement, it then falls to one of the basis states that form the superposition, thus destroying the original configuration. Such phenomenon is applied to the qubits, so that, a qubit can take, in addition to the basic states 0 and 1, a set of states resulting from the linear combination of the basic states [8, 52].

Entanglement: This is a state where two or more particles are generated so that the quantum state of a single particle cannot be destroyed independently. No matter how far the particles are, they will reflect the same quantum state [49]. Any change at one end in the quantum state will be responsible for the change in the other particle. In the case of qubits, entanglement is the ability of qubits to correlate their state with other qubits.

The properties of the two qubits in an entangled state are linked to each other such that by looking (i.e., measuring) one of them, will reveal the other qubit, even when they are at physically large separations [52]. Two or more individually independent quantum objects are said to be entangled when: a) their behavior is random individually, but at the same time, and b) it is strongly correlated despite each object being independent of the other. A multiqubit state that cannot be expressed as a list of the individual constituent qubits is entangled.

Uncertainty principle of Quantum Mechanics: It was formulated in 1926 by Werner Heisenberg, this principle states that an electron, or any other particle, can never have its exact position known, or even specified. The uncertainty principle states that the position and the momentum of an elementary particle (EP) cannot be simultaneously determined with certainty [53]. Quantum systems are so small; it is impossible to measure all properties of a Quantum system without disturbing it [54-55]. As a result, there is no way of accurately predicting all the properties of a particle in a Quantum System.

Probability principle [56-58]: A first general principle in quantum mechanics is that the probability that a particle will arrive at x position when let out at the source s can be represented quantitatively by the absolute square of a complex number called a probability amplitude—in this case, the “amplitude that a particle from a starting point s will arrive at position x .”

In quantum mechanics, particles do not have classical properties like “position” or “momentum,” rather, there is a wave function that assigns a (complex) number, called the “amplitude,” to each measurement outcome. The Born Rule is then quite simple: *it says that the probability of obtaining any possible measurement's outcome is equal to the square of the corresponding amplitude.* (The wave function is just the set of all the amplitudes).

Born Rule: $\text{Probability}(x) = |\text{amplitude}(x)|^2$

In quantum mechanics, a probability amplitude is a complex number used for describing the behavior of systems. The modulus squared of this quantity represents a probability density. Probability amplitudes provide a relationship between the quantum state vector of a system and the results of observations of that system, a link was first proposed by Max Born, in 1926. The interpretation of values of a wave function as the probability amplitude is a pillar of the Copenhagen interpretation of quantum mechanics.

Measurement principle: In quantum physics, measurement is the testing or manipulation of a physical system to yield a numerical result [59]. The predictions that quantum physics makes are in general probabilistic. For each measurement that can be defined, the probability distribution over the outcomes of that measurement can be computed from the density operator.

Decoherence: Quantum decoherence is the loss of quantum coherence [8], the process in which a system's behavior changes from that which can be explained by quantum mechanics to that which can be explained by classical mechanics. Decoherence can be viewed as the loss of information from a system into the environment (often modeled as a heat bath) [60] since every system is loosely coupled with the energetic state of its surroundings.

Non-locality: In [61], it is indicated that non-locality is the most characteristic feature of quantum mechanics, but recent research suggests the possible existence of non-local correlations stronger than those predicted by theory. This raises the question of whether nature is in fact more non-local than expected from quantum theory or, alternatively, whether there could be a yet undiscovered principle limiting the strength of non-local correlations.

RQ3: What are the supporting concepts for implementing Quantum Computing?

Quantum computing as an engineering discipline is still in its infancy [62-63], and only some practical prototypes have been announced. Quantum hardware is going to be implemented, it is a clear objective [64]; the principal manufacturers are creating different technologies.

By today, there are efforts in developing quantum technology applications with practical realizations using photons, atoms, and electrons [65]. In [22], it is presented a list of major hardware candidates for industrial quantum computer and their properties: Qubit Technologies, Trapped Ion Qubits, Super-conducting Qubits, Silicon Qubits, Photonic Qubits, and Topological Qubits.

By this time, there are simulators running on personal computers, which allows the user to run quantum programs. The current implementations of quantum computing are based on quantum gates and quantum algorithms.

Quantum gates are quantum logic objects, and they have a mathematical expression (see examples of Hadamard and CNOT gates in previous sections).

Based on this, we consider as supporting concepts the following: quantum gates, quantum circuits, mathematical expressions, and quantum algorithms. In Fig. 6 we present a kind of quantum gates taxonomy.

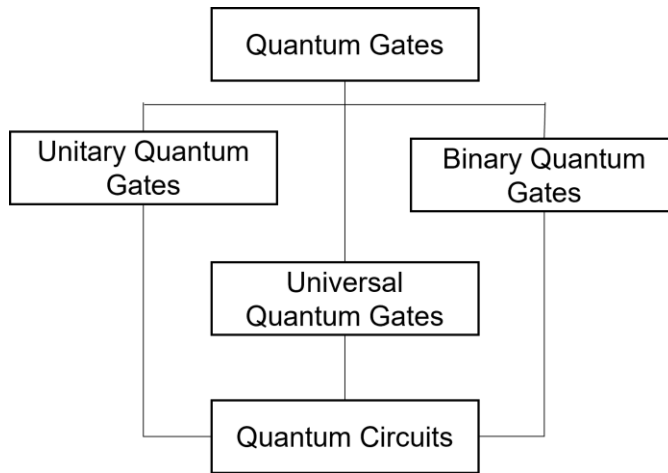


Fig. 6. Concepts of electronics involved in quantum computing

RQ4: What are the new Software Engineering concepts emerging from Quantum Computing?

The connection between quantum computing and software engineering is clearly deduced, software is an essential part of quantum computing, and developing software is the main objective of software engineering. In this section, we present the main concepts detected about software engineering, which emerged from the context of quantum computing.

We present eight elements of quantum software engineering (see Fig. 7, 8): (1) quantum software life cycle, (2) quantum software processes, (3) quantum software modeling, (4) quantum software implementation, (5) quantum software quality assurance, (6) quantum software programming, (7) quantum software tools, and (8) quantum software maintenance. Quantum software modeling is divided into quantum software analysis and quantum software design. Next, we will describe some of them.

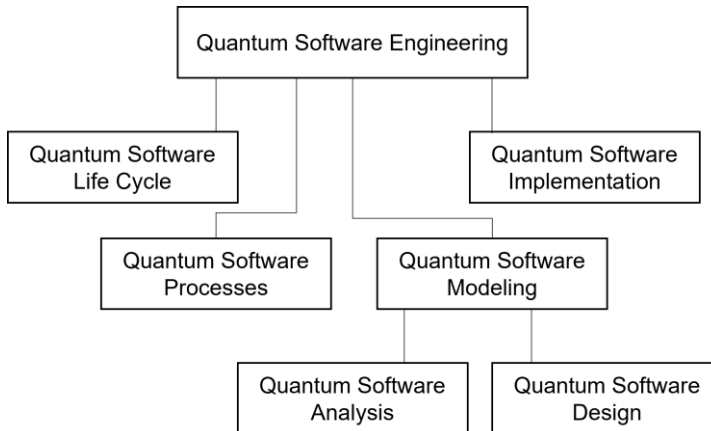


Figure 7. Areas of Quantum Software Engineering. Part 1

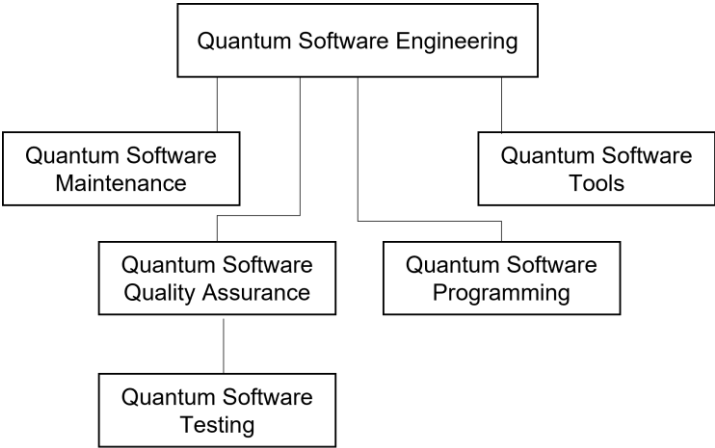


Figure 8. Areas of Quantum Software Engineering. Part 2

Quantum software lifecycle. There are proposals for the lifecycle of quantum software, such as [11, 44, 64]. They propose a kind of traditional main phase: requirements, analysis, design, implementation, testing, and maintenance. Other proposals include specific subphases according to the quantum nature. A generic proposal is presented in [52], which describes the phases according to their quantum nature. The lifecycle for quantum software is divided into ten phases (See Fig. 9, 10).

The principal activities in each phase are cited below:

1. *Quantum-Classical Splitting*: Problem separation in classical and quantum parts. E.g., manual by experts, decision support based on patterns.
2. *Hardware-independent Implementation*: Quantum circuit & classical software artifacts, testing & verification of circuits.
3. *Quantum Circuit Enrichment*: Data preparation/initialization, oracle expansion.
4. *Hardware-independent Optimization*: Removal of unnecessary gates or qubits, based on cost functions (e.g., circuit depth, accuracy).
5. *Quantum Hardware Selection*: Analysis of quantum circuits, selection of suitable hardware (e.g., based on metrics or benchmarks).
6. *Readout-Error Mitigation Preparation*: Analysis of selected hardware, and determination of the error model (e.g., calculation of the correction matrix).
7. *Compilation & Hardware dependent Optimization*: Optimizations based on hardware characteristics, compilation to machine instructions.
8. *Integration*: Deployment of classical software artifacts, and provisioning of quantum resources.
9. *Execution*: On heterogeneous hardware, multiple iterations for variational algorithms, readout-error mitigation.
10. *Result Analysis*: Result verification, return to the user or next iteration for improvement.

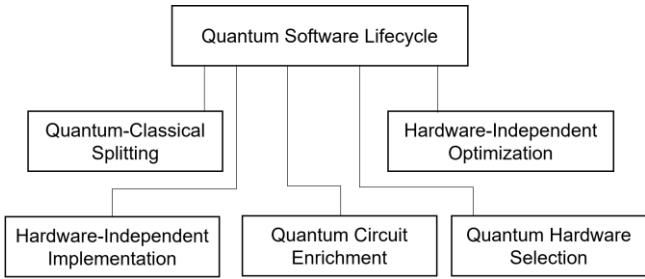


Fig. 9. Stages of the Quantum Lifecycle. Part 1

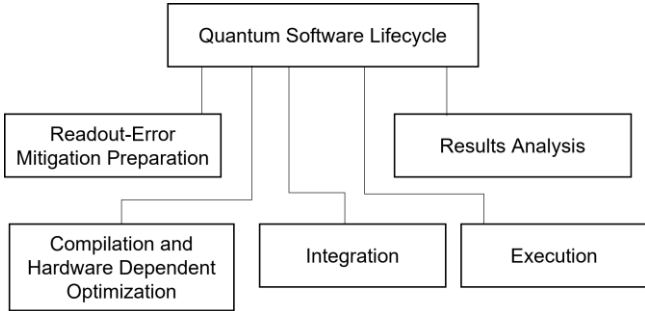


Fig. 10. Stages of the Quantum Life Cycle. Part 2

Quantum software modeling. The design is one of the most cited in the literature, this is because both quantum circuits and quantum algorithms are the basis of quantum software, so this is a kind of low-level design. However, from the software engineering point of view, the design phase is required to be supported by techniques like those for developing classical software, in this case, it is needed to create techniques and tools for quantum requirements analysis and quantum modeling. UML is a representative tool to support software analysis and design, and it can be used as a starting point.

There are a few proposals about modeling techniques for quantum software. Some observations are stated to establish the basis to create supporting tools for the design phase [66]. The central difference between quantum and classical computation is in how it achieves its goals. Quantum computers have access to quantum algorithms [67], and quantum data structures [68], that are unavailable to classical computers—hence their performance advantage. Algorithms and data structures are, however, implementation details. Algorithms are an essential design choice while programming in the small. However, they are not completely ignored in large-scale software architectural design. For instance, UML diagrams seldom portray algorithms and data structures beyond a very high-level design perspective.

In classic Software Engineering, at the software modeling phase, there are in total 14 types of UML diagrams, split into two categories [69]: structure and behavior diagrams.

In [66], the authors propose to create an extension of UML, called Q-UML, which follows the guiding principles behind any quantum software modeling language: Quantum classes, quantum elements (quantum variables, quantum operations), quantum supremacy, and quantum aggregation). These elements make a difference with respect to classical software modeling.

In [69] the author suggests that of the fourteen diagram types in UML, the most widely used (and hence important) diagrams are: use case, class, object, state machine, sequence, and activity diagrams. Based on this, the author presents a proposal for Q-UML diagrams, as we can see in Fig. 11. The Q-UML diagrams contain similar elements to UML diagrams, distinguishing the quantum

elements with highlights such as bold text and double-lines to portray quantum information textually and pictorially.

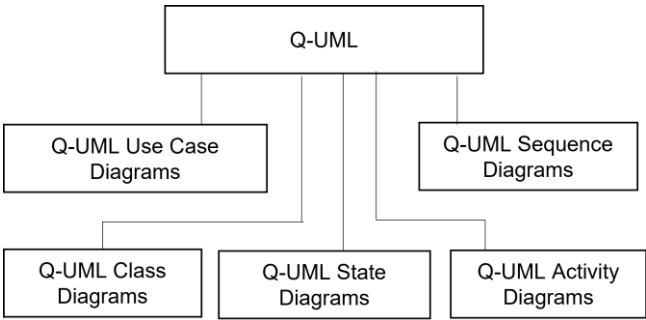


Fig. 11. Q-UML basic diagrams

Quantum programming languages: Quantum programming is the process of assembling sequences of instructions, called quantum circuits, which can run on a quantum computer. Quantum programming languages help express quantum algorithms using high-level constructs [70]. Quantum programming languages are used for controlling existing physical devices, for estimating the execution costs of quantum algorithms on future devices, for teaching quantum computing concepts, or for verifying quantum algorithms and their implementations [71].

In [72] a classification of quantum programming languages is presented (see Fig. 12), which is considered as a starting point for studying quantum programming. This taxonomy has two branches, *programming languages*, and *programming tools*.

Quantum software testing. By nature, quantum software is error susceptible. Quantum physics properties as superposition mean that quantum computers deliver probabilistic measures when classical observations are made on qubits; that is when a qubit in a superposition state is collapsed into a classical value, it takes a given value with a given probability [73]. Some selected publications address quantum computing validation from a probabilistic perspective from circuit and software levels. The behavior of quantum circuits is inherently probabilistic, so while the goal of traditional testing has always been to detect the presence of faults, probabilistic testing aims to estimate fault probability.

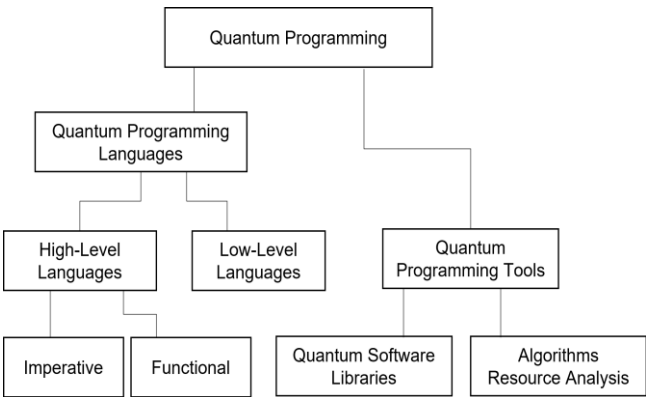


Fig. 12. Quantum programming branches

The testing phase is one of the most covered by hardware producers and scientists. In Fig. 13 we show a classification of types of testing techniques for quantum software. It was extracted from [63, 74-84], which treat specific techniques and tools for debugging and testing.

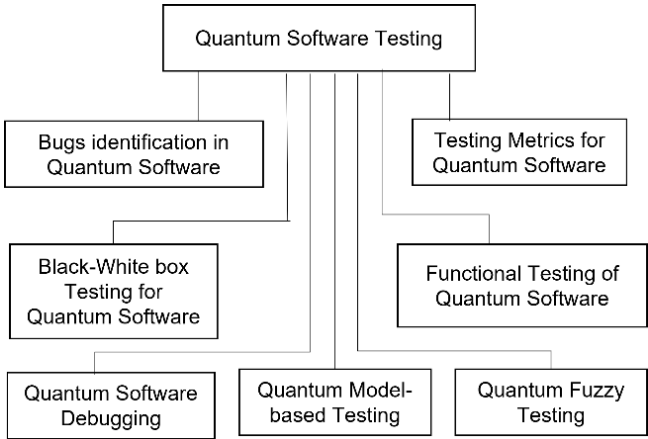


Fig. 13. Quantum programming branches

As we can see, the classification includes bug detection and debugging as part of the quality assurance. White-Black box testing and functional testing are like classical software testing applied to quantum software. Specialized testing is included, such as fuzz testing [80, 84], which is an automated software testing method that injects invalid, malformed, or unexpected inputs into a system to reveal software defects and vulnerabilities. Quantum noise provides an effective built-in fuzzing capability that is centered around the actual answer to a computation. Specialized techniques are introduced for testing quantum software, such as search-based [79], mutation-based [24, 76, 82], and property-based [24, 78].

6. Discussion

Considering the results presented in the previous section we can describe the following facts.

6.1. Tendencies in the coverture of quantum fundamental concepts

- In this sample of literature review, it is seen that there are more sources on quantum computing essentials and tendencies (Category 1). Most research has been done on quantum algorithms, hardware prototyping, and error mitigation.
- Even when these publications attend essential aspects of quantum computing, some of them do not include a full description of the quantum mechanics principles that support quantum computing, commonly they include references to earlier publications and books.
- Few publications exist talking about quantum computing education, as well as few formal publications exist talking about the skills required for practicing quantum computing (Categories 3, 4).

6.2. Tendencies in the coverture quantum software development

- Due to the nature of quantum computing and its exposure to errors, debugging and evaluating such errors has been a problem to address since the beginning of quantum computing practical implementations. Then, more progress is registered for quantum software testing.

- Modeling quantum software shows advances in modeling circuits and quantum algorithms; it is a kind of inherent design required. However, high-level modeling is still in its infancy.
- There is a proliferation in quantum programming languages. At the high-level category, there are several languages based on instructions that implement the quantum mechanics principles, such as superposition, entanglement, and measurement.

6.3. Benefits of the presented results of this review

- The related work cited in this paper presents significant effort in literature review and taxonomical proposals of specific aspects of quantum computing, especially at the level of main aspects, technologies, and use cases. However, the fundamental concepts are presented in a descriptive way, not in a taxonomical view.
- The taxonomical view presented in our paper will allow us to present the knowledge better organized. Also, it represents a starting point to document the principal part of a body of knowledge, the fundamental concepts.

7. Conclusions and future work

A new paradigm of computation known as quantum computing is founded on the ideas of quantum theory, which is concerned with contemporary physics, particularly quantum mechanics. The main goal of quantum mechanics is to describe how matter and energy behave at the atomic and subatomic scales. Quantum computing makes use of quantum phenomena, such as superposition and entanglement to perform data operations. Those principles are becoming popular due to their importance, such as entanglement [85] which is particularly useful in information security issues.

The multi-disciplinarity of quantum computing, especially the quantum mechanics principles and the mathematical expression of qubits and their treatment, make this paradigm difficult to understand by non-physics experts [23-24].

To reduce the complication of understanding and mastering quantum computing, and for addressing the study of this field and the implementation of quantum software, practitioners (academicians and industry professionals) require a kind of taxonomic view of the fundamental concepts of quantum computing and the supporting ones.

In this paper, we presented a quasi-systematic mapping, performed to:

1. Identify the fundamental concepts involved in quantum computing and quantum software engineering.
2. Build a taxonomy to include the main concepts of both disciplines.
3. Identify a set of fundamental concepts to integrate the first element of a body of knowledge.

Software engineering education has implications for the software industry in emerging countries [86] and developed countries. Education and training in software development require considering both hard and soft skills [87]; this is the same for quantum software development.

The proposed taxonomy represents an initial proposal of the body of knowledge of quantum computing, especially for software development, so it can contribute to education in quantum computing and quantum software engineering.

For future work, we identify the next actions:

1. To complete the analysis of the results from all databases recommended for any literature review, and to find how the literature covers the fundamental concepts of both disciplines.
2. To formalize the integration of the taxonomy, with a kind of theoretic validation or with judges from experts.

3. To assess the usefulness of the first levels of the taxonomy, in different scenarios such as guiding curricula creation and course design.
4. To continue with the literature review to gather the remaining elements of a body of knowledge, such as practices and methods.

References

- [1]. M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, 10th edition. Cambridge, United Kingdom: Cambridge University Press, 2016.
- [2]. A. A. Khan et al., "Agile Practices for Quantum Software Development: Practitioners Perspectives," ArXiv, vol. abs/2210.09825, 2022.
- [3]. A. J. et al., "Quantum Algorithm Implementations for Beginners," Apr. 2018, doi: 10.1145/3517340.
- [4]. J. J. García-Ripoll, "Specialty Grand Challenge: Quantum engineering," Frontiers in Quantum Science and Technology, vol. 1, Sep. 2022, doi: 10.3389/frqst.2022.1029525.
- [5]. S. T. Marella and H. S. K. Parisa, "Introduction to Quantum Computing," in Quantum Computing and Communications, Y. Zhao, Ed., Rijeka: IntechOpen, 2020, p. Ch. 5. doi: 10.5772/intechopen.94103.
- [6]. M. Weingärtner and T. Weingärtner, "Quantum Tic-Tac-Toe - learning the concepts of quantum mechanics in a playful way," Computers and Education Open, vol. 4, p. 100125, 2023, doi: <https://doi.org/10.1016/j.caeo.2023.100125>.
- [7]. U. Awan, L. Hannola, A. Tandon, R. K. Goyal, and A. Dhir, "Quantum computing challenges in the software industry. A fuzzy AHP-based approach," Inf Softw Technol, vol. 147, p. 106896, 2022, doi: <https://doi.org/10.1016/j.infsof.2022.106896>.
- [8]. T. G. Wong, Introduction to Classical and Quantum Computing. Omaha, Nebraska: Rooted Grove, 2022.
- [9]. N. S. Yanofsky and M. A. Mannucci, Quantum computing for computer scientists. New York, NY: Cambridge University Press, 2008.
- [10]. E. Rieffel and W. Polak, QUANTUM COMPUTING A Gentle Introduction. Cambridge, Massachusetts: The MIT Press, 2011.
- [11]. J. Zhao, Quantum Software Engineering: Landscapes and Horizons. 2020.
- [12]. P. Zhao, J. Zhao, and L. Ma, "Identifying Bug Patterns in Quantum Programs," in 2021 IEEE/ACM 2nd International Workshop on Quantum Software Engineering (Q-SE), 2021, pp. 16–21. doi: 10.1109/Q-SE52541.2021.00011.
- [13]. P. P. Angara, U. Stege, A. MacLean, H. A. Müller, and T. Markham, "Teaching Quantum Computing to High-School-Aged Youth: A Hands-On Approach," IEEE Transactions on Quantum Engineering, vol. 3, pp. 1–15, 2022, doi: 10.1109/TQE.2021.3127503.
- [14]. M. Mykhailova and K. M. Svore, "Teaching Quantum Computing through a Practical Software-Driven Approach: Experience Report," in Proceedings of the 51st ACM Technical Symposium on Computer Science Education, in SIGCSE '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 1019–1025. doi: 10.1145/3328778.3366952.
- [15]. J. Singh and K. S. Bhangu, "Contemporary Quantum Computing Use Cases: Taxonomy, Review and Challenges," Archives of Computational Methods in Engineering, vol. 30, no. 1, pp. 615–638, 2023, doi: 10.1007/s11831-022-09809-5.
- [16]. C. D. Aiello et al., "Achieving a quantum smart workforce," Quantum Sci Technol, vol. 6, no. 3, p. 030501, Jul. 2021, doi: 10.1088/2058-9565/abfa64.
- [17]. G. R. Oliver, Foundations of the assumed business operations and strategy body of knowledge (BOSBOK) : an outline of shareable knowledge. Sidney: Sydney University Press, 2012.
- [18]. P. Bourque and R. E. Fairley, "Guide to the Software Engineering Body of Knowledge, Version 3.0," 2014.
- [19]. T. I. Ören, "Toward the Body of Knowledge of Modeling and Simulation," in Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2005, 2005.
- [20]. G. Romme, The Quest for Professionalism: The Case of Management and Entrepreneurship. Oxford: Oxford University Press, 2016.
- [21]. H. Hart and C. Baehr, "Sustainable Practices for Developing a Body of Knowledge," Tech Commun, vol. 60, no. 4, pp. 259–266, 2013, [Online]. Available: <https://www.jstor.org/stable/26464355>
- [22]. S. S. Gill et al., "Quantum Computing: A Taxonomy, Systematic Review and Future Directions," Sep. 2020, doi: 10.48550/arxiv.2010.15559.

- [23]. L. S. Barbosa, "Software Engineering for 'Quantum Advantage,'" in Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops, in ICSEW'20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 427–429. doi: 10.1145/3387940.3392184.
- [24]. G. Pontolillo and M. R. Mousavi, "A Multi-Lingual Benchmark for Property-Based Testing of Quantum Programs," in 2022 IEEE/ACM 3rd International Workshop on Quantum Software Engineering (Q-SE), 2022, pp. 1–7. doi: 10.1145/3528230.3528395.
- [25]. Y.-P. Liao, Y.-L. Cheng, Y.-T. Zhang, H.-X. Wu, and R.-C. Lu, "The interactive system of Bloch sphere for quantum computing education," in 2022 IEEE International Conference on Quantum Computing and Engineering (QCE), 2022, pp. 718–723. doi: 10.1109/QCE53715.2022.00097.
- [26]. G. P. Temporão, T. B. S. Guerreiro, P. S. C. Ripper, and A. M. B. Pavani, "Teaching Quantum Computing without prerequisites: a case study," in 2022 IEEE International Conference on Quantum Computing and Engineering (QCE), 2022, pp. 673–676. doi: 10.1109/QCE53715.2022.00090.
- [27]. M. Mykhailova, "Developing Programming Assignments for Teaching Quantum Computing and Quantum Programming," in 2022 IEEE International Conference on Quantum Computing and Engineering (QCE), 2022, pp. 688–692. doi: 10.1109/QCE53715.2022.00092.
- [28]. C. Hughes, D. Finke, D.-A. German, C. Merzbacher, P. M. Vora, and H. J. Lewandowski, "Assessing the Needs of the Quantum Industry," Aug. 2021, doi: 10.48550/arxiv.2109.03601.
- [29]. K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic Mapping Studies in Software Engineering," pp. 1–10, 2008, doi: 10.14236/ewic/EASE2008.8.
- [30]. B. Barn, S. Barat, and T. Clark, "Conducting Systematic Literature Reviews and Systematic Mapping Studies," in Proceedings of the 10th Innovations in Software Engineering Conference, in ISEC '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 212–213. doi: 10.1145/3021460.3021489.
- [31]. K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," *Inf Softw Technol*, vol. 64, pp. 1–18, 2015, doi: <https://doi.org/10.1016/j.infsof.2015.03.007>.
- [32]. B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Technical report, EBSE Technical Report EBSE-2007-01, 2007. [Online]. Available: <https://www.cs.auckland.ac.nz/~norsaremah/2007%20Guidelines%20for%20performing%20SLR%20in%20SE%20v2.3.pdf>
- [33]. B. A. Kitchenham, "Systematic reviews," in 10th International Symposium on Software Metrics, 2004. Proceedings., 2004, pp. xii–xii. doi: 10.1109/METRIC.2004.1357885.
- [34]. J. Hannay, D. Sjøberg, and T. Dybå, "A Systematic Review of Theory Use in Software Engineering Experiments," *Software Engineering, IEEE Transactions on*, vol. 33, pp. 87–107, Mar. 2007, doi: 10.1109/TSE.2007.12.
- [35]. G. Tebes, D. Peppino, P. Becker, and L. Olsina, Enhancing the Process Specification for Systematic Literature Reviews. 2019. doi: 10.13140/RG.2.2.14262.96321/1.
- [36]. P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," *Journal of Systems and Software*, vol. 80, no. 4, pp. 571–583, 2007, doi: <https://doi.org/10.1016/j.jss.2006.07.009>.
- [37]. B. Kitchenham, E. Mendes, and G. Travassos, "Cross versus Within-Company Cost Estimation Studies: A Systematic Review," *Software Engineering, IEEE Transactions on*, vol. 33, pp. 316–329, Jun. 2007, doi: 10.1109/TSE.2007.1001.
- [38]. M. R. W. Hiebl, "Sample Selection in Systematic Literature Reviews of Management Research," *Organ Res Methods*, vol. 26, no. 2, pp. 229–261, Jan. 2021, doi: 10.1177/1094428120986851.
- [39]. V. Garousi, M. Felderer, and M. V. Mäntylä, "Guidelines for including grey literature and conducting multivocal literature reviews in software engineering," *Inf Softw Technol*, vol. 106, pp. 101–121, 2019, doi: <https://doi.org/10.1016/j.infsof.2018.09.006>.
- [40]. A. Sarkar, "Automated Quantum Software Engineering: why? what? how?," Dec. 2022, doi: 10.48550/arxiv.2212.00619.
- [41]. B. Weder, J. Barzen, F. Leymann, and D. Vietz, "Quantum Software Development Lifecycle," Jun. 2021, doi: 10.48550/arxiv.2106.09323.
- [42]. A. A. Khan et al., "Software Architecture for Quantum Computing Systems -- A Systematic Review," Feb. 2022, Accessed: Mar. 18, 2023. [Online]. Available: <http://arxiv.org/abs/2202.05505>

- [43]. M. De Stefano, F. Pecorelli, D. Di Nucci, F. Palomba, and A. De Lucia, "Software Engineering for Quantum Programming: How Far Are We?," Mar. 2022, doi: 10.48550/arxiv.2203.16969.
- [44]. 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, doi: 10.1145/3548679.
- [45]. 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: Springer International Publishing, 2022, pp. 25–38. doi: 10.1007/978-3-031-05324-5_2.
- [46]. M. Openja, M. M. Morovati, L. An, F. Khomh, and M. Abidi, "Technical debts and faults in open-source quantum software systems: An empirical study," *Journal of Systems and Software*, vol. 193, p. 111458, 2022, doi: <https://doi.org/10.1016/j.jss.2022.111458>.
- [47]. S. Singh, M. T. Pandian, A. K. Aggarwal, S. P. Awasthi, H. Bhardwaj, and J. Pruthi, "Quantum learning theory: A classical perspective for quantum image," *Mater Today Proc*, vol. 80, pp. 2786–2793, 2023, doi: <https://doi.org/10.1016/j.matpr.2021.07.039>.
- [48]. G. Arun and V. Mishra, "A review on quantum computing and communication," in *2014 2nd International Conference on Emerging Technology Trends in Electronics, Communication and Networking*, 2014, pp. 1–5. doi: 10.1109/ET2ECN.2014.7044953.
- [49]. J. Singh and M. Singh, "Evolution in Quantum Computing," in *2016 International Conference System Modeling & Advancement in Research Trends (SMART)*, 2016, pp. 267–270. doi: 10.1109/SYSMART.2016.7894533.
- [50]. S. Aralikatti, "Quantum Computing: Challenges and Opportunities," in *2021 Fourth International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, 2021, pp. 1–4. doi: 10.1109/ICECCT52121.2021.9616647.
- [51]. M. Dickson, "NON-RELATIVISTIC QUANTUM MECHANICS," in *Philosophy of Physics*, J. Butterfield and J. Earman, Eds., Amsterdam: North-Holland, 2007, pp. 275–415. doi: <https://doi.org/10.1016/B978-044451560-5/50007-5>.
- [52]. B. Sodhi and R. Kapur, "Quantum Computing Platforms: Assessing the Impact on Quality Attributes and SDLC Activities," in *2021 IEEE 18th International Conference on Software Architecture (ICSA)*, 2021, pp. 80–91. doi: 10.1109/ICSA51549.2021.00016.
- [53]. J. Bub, "Quantum Mechanics as a Principle Theory," *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, vol. 31, no. 1, pp. 75–94, 2000, doi: [https://doi.org/10.1016/S1355-2198\(99\)00032-5](https://doi.org/10.1016/S1355-2198(99)00032-5).
- [54]. J. Hilgevoord, "The uncertainty principle for energy and time," *Am J Phys*, vol. 64, no. 12, pp. 1451–1456, Dec. 1996, doi: 10.1119/1.18410.
- [55]. J. Hilgevoord, "The uncertainty principle for energy and time. II," *Am J Phys*, vol. 66, no. 5, pp. 396–402, May 1998, doi: 10.1119/1.18880.
- [56]. E. Sakai, "On the principles of quantum mechanics," May 2004, doi: 10.48550/arxiv.quant-ph/0405069.
- [57]. D. J. Velleman, "Probability and quantum mechanics," *Am J Phys*, vol. 66, no. 11, pp. 967–969, Nov. 1998, doi: 10.1119/1.19007.
- [58]. W. B. Hodge, S. V. Migirditch, and W. C. Kerr, "Electron spin and probability current density in quantum mechanics," *Am J Phys*, vol. 82, no. 7, pp. 681–690, Jun. 2014, doi: 10.1119/1.4868094.
- [59]. L. Masanes, T. D. Galley, and M. P. Müller, "The measurement postulates of quantum mechanics are operationally redundant," *Nat Commun*, vol. 10, no. 1, p. 1361, 2019, doi: 10.1038/s41467-019-09348-x.
- [60]. D. Bacon, "Decoherence, Control, and Symmetry in Quantum Computers," May 2003, Accessed: Mar. 20, 2023. [Online]. Available: <https://arxiv.org/abs/quant-ph/0305025>
- [61]. S. Popescu, "Nonlocality beyond quantum mechanics," *Nat Phys*, vol. 10, no. 4, pp. 264–270, 2014, doi: 10.1038/nphys2916.
- [62]. N. C. Jones et al., "Layered Architecture for Quantum Computing," *Phys Rev X*, vol. 2, no. 3, p. 31007, Jul. 2012, doi: 10.1103/PhysRevX.2.031007.
- [63]. D. Fortunato, J. CAMPOS, and R. ABREU, "Mutation Testing of Quantum Programs: A Case Study With Qiskit," *IEEE Transactions on Quantum Engineering*, vol. 3, pp. 1–17, 2022, doi: 10.1109/TQE.2022.3195061.
- [64]. R. Van Meter and M. Oskin, "Architectural implications of quantum computing technologies," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 2, pp. 31–63, 2006.

- [65]. A. G. Jordan, "Frontiers of research and future directions in information and communication technology," *Technol Soc*, vol. 30, no. 3, pp. 388–396, 2008, doi: <https://doi.org/10.1016/j.techsoc.2008.05.002>.
- [66]. C. A. Perez-Delgado and H. G. Perez-Gonzalez, "Towards a Quantum Software Modeling Language," Jun. 2020, Accessed: Mar. 18, 2023. [Online]. Available: <https://arxiv.org/abs/2006.16690>
- [67]. P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134, 1994.
- [68]. L. Zhao, C. A. Pérez-Delgado, and J. F. Fitzsimons, "Fast graph operations in quantum computation," *Phys Rev A (Coll Park)*, vol. 93, no. 3, p. 032314, Mar. 2016, doi: [10.1103/PhysRevA.93.032314](https://doi.org/10.1103/PhysRevA.93.032314).
- [69]. C. A. Pérez-Delgado, "A Quantum Software Modeling Language," in *Quantum Software Engineering*, M. A. Serrano, R. Pérez-Castillo, and M. Piattini, Eds., Cham: Springer International Publishing, 2022, pp. 103–119. doi: [10.1007/978-3-031-05324-5_6](https://doi.org/10.1007/978-3-031-05324-5_6).
- [70]. J. A. Miszczak, *High-level Structures in Quantum Computing*. Springer Cham, 2012.
- [71]. B. Heim et al., "Quantum programming languages," *Nature Reviews Physics*, vol. 2, no. 12, pp. 709–722, 2020, doi: [10.1038/s42254-020-00245-7](https://doi.org/10.1038/s42254-020-00245-7).
- [72]. O. Ayoade, P. Rivas, and J. Orduz, "Artificial Intelligence Computing at the Quantum Level," *Data (Basel)*, vol. 7, no. 3, p. 28, Feb. 2022, doi: [10.3390/data7030028](https://doi.org/10.3390/data7030028).
- [73]. A. García de la Barrera, I. García-Rodríguez de Guzmán, M. Polo, and M. Piattini, "Quantum software testing: State of the art," *Journal of Software: Evolution and Process*, vol. n/a, no. n/a, p. e2419, Dec. 2021, doi: <https://doi.org/10.1002/smr.2419>.
- [74]. M. Paltenghi, "Cross-Platform Testing of Quantum Computing Platforms," in *2022 IEEE/ACM 44th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, 2022, pp. 269–271. doi: [10.1145/3510454.3517061](https://doi.org/10.1145/3510454.3517061).
- [75]. N. Costa, J. P. Fernandes, and R. Abreu, "Asserting the Correctness of Shor Implementations Using Metamorphic Testing," in *Proceedings of the 1st International Workshop on Quantum Programming for Software Engineering*, in QP4SE 2022. New York, NY, USA: Association for Computing Machinery, 2022, pp. 32–36. doi: [10.1145/3549036.3562062](https://doi.org/10.1145/3549036.3562062).
- [76]. D. Fortunato, J. Campos, and R. Abreu, "QMutPy: A Mutation Testing Tool for Quantum Algorithms and Applications in Qiskit," in *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*, in ISSA 2022. New York, NY, USA: Association for Computing Machinery, 2022, pp. 797–800. doi: [10.1145/3533767.3543296](https://doi.org/10.1145/3533767.3543296).
- [77]. J. Wang, Q. Zhang, G. H. Xu, and M. Kim, "QDiff: Differential Testing of Quantum Software Stacks," in *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2021, pp. 692–704. doi: [10.1109/ASE51524.2021.9678792](https://doi.org/10.1109/ASE51524.2021.9678792).
- [78]. S. Honarvar, M. R. Mousavi, and R. Nagarajan, "Property-Based Testing of Quantum Programs in Q#," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, in ICSEW'20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 430–435. doi: [10.1145/3387940.3391459](https://doi.org/10.1145/3387940.3391459).
- [79]. X. Wang, P. Arcaini, T. Yue, and S. Ali, "QuSBT: Search-Based Testing of Quantum Programs," in *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings*, in ICSE '22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 173–177. doi: [10.1145/3510454.3516839](https://doi.org/10.1145/3510454.3516839).
- [80]. M. Trinca, J. F. Ferreira, and R. Abreu, "A Preliminary Study on Generating Well-Formed Q# Quantum Programs for Fuzz Testing," in *2022 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2022, pp. 118–121. doi: [10.1109/ICSTW55395.2022.00033](https://doi.org/10.1109/ICSTW55395.2022.00033).
- [81]. J. Campos and A. Souto, "Q Bugs: A Collection of Reproducible Bugs in Quantum Algorithms and a Supporting Infrastructure to Enable Controlled Quantum Software Testing and Debugging Experiments," in *2021 IEEE/ACM 2nd International Workshop on Quantum Software Engineering (Q-SE)*, 2021, pp. 28–32. doi: [10.1109/Q-SE52541.2021.00013](https://doi.org/10.1109/Q-SE52541.2021.00013).
- [82]. D. Fortunato, J. Campos, and R. Abreu, "Mutation Testing of Quantum Programs Written in QISKit," in *2022 IEEE/ACM 44th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, 2022, pp. 358–359. doi: [10.1145/3510454.3528649](https://doi.org/10.1145/3510454.3528649).
- [83]. D. Zhu, S. Wang, J. Han, R. Wei, J. Wu, and L. Song, "Research on method for adding new friends to instant messaging system based on SIP in quantum communication network," in *2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2015, pp. 467–469. doi: [10.1109/ICSESS.2015.7339098](https://doi.org/10.1109/ICSESS.2015.7339098).

- [84]. J. Wang et al., “QuanFuzz: Fuzz Testing of Quantum Program,” ArXiv, vol. abs/1810.10310, 2018.
- [85]. V. V Kornyak, “Dynamic Simulation of Quantum Entanglement in Finite Quantum Mechanics: A Computer Algebra Approach,” *Programming and Computer Software*, vol. 47, no. 2, pp. 124–132, 2021, doi: 10.1134/S0361768821020067.
- [86]. L. Vives, K. Melendez, and A. Dávila, “ISO/IEC 29110 and Software Engineering Education: A Systematic Mapping Study,” *Programming and Computer Software*, vol. 48, no. 8, pp. 745–755, 2022, doi: 10.1134/S0361768822080229.
- [87]. R. Juárez-Ramírez et al., “How COVID-19 Pandemic affects Software Developers’ Wellbeing, and the Necessity to strengthen Soft Skills,” *Programming and Computer Software*, vol. 48, no. 8, pp. 614–631, 2022, doi: 10.1134/S0361768822080047.

APPENDIX A. Selected papers after applying filters of level 1 and exclusion criteria

ID_SR: Source Reference extracted from the search, consecutively numbered.

ID_REF: Reference number from the list of References on the paper.

Note: A complete list of references in APA format for the papers in this Appendix can be found in the file “PCS2023_Appendix A_v6.11.pdf”, at:

[https://drive.google.com/file/d/1qQvusv9-](https://drive.google.com/file/d/1qQvusv9-Jm9iSvYAdVEtABRQXoxbvFmM/view?usp=share_link)

[Jm9iSvYAdVEtABRQXoxbvFmM/view?usp=share_link](https://drive.google.com/file/d/1qQvusv9-Jm9iSvYAdVEtABRQXoxbvFmM/view?usp=share_link)

Table 4. Papers selected from ScienceDirect

ID_SR	ID_REF	PAPER TITLE
S1	[43]	Software engineering for quantum programming: How far are we?
S2	[7]	Quantum computing challenges in the software industry. A fuzzy AHP-based approach
S3	-	Assessing attitudes towards evidence-based software engineering in a government agency
S4	-	A Comprehensive but not Complicated Survey on Quantum Computing
S5	-	Engineering the development of quantum programs: Application to the Boolean satisfiability problem
S6	-	Software modernization to embrace quantum technology
S7	[46]	Technical debts and faults in open-source quantum software systems: An empirical study
S8	-	Automated data validation: An industrial experience report
S9	[47]	Quantum learning theory: A classical perspective for quantum image
S10	-	Rapid solution of logical equivalence problems by quantum computation algorithm
S11	-	Comparing the performance of quantum-inspired evolutionary algorithms for the solution of software requirements selection problem
S12	[65]	Frontiers of research and future directions in information and communication technology
S13	-	East-West paths to unconventional computing

Table 5. Papers selected from ACM

ID_SR	ID_REF	PAPER TITLE
S14	-	Quantum computing for software engineering: prospects
S15	-	Quantum Computing: A New Software Engineering Golden Age
S16	-	Software Engineering Education of Classical Computing vs. Quantum Computing: A Competency-Centric Approach
S17	-	When software engineering meets quantum computing
S18	-	Quantum computing: synergies and opportunities
S19	[23]	Software engineering for 'quantum advantage'
S20	[50]	Quantum computing: challenges and opportunities
S21	-	Making Quantum Computing Open: Lessons from Open Source Projects
S22	-	Towards Higher-Level Abstractions for Quantum Computing
S23	[74]	Cross-platform testing of quantum computing platforms
S24	-	AI4ASE: Quantum artificial intelligence for automotive software engineering
S25	[14]	Teaching Quantum Computing through a Practical Software-driven Approach: Experience Report
S26	-	A Quantum Algorithm for Software Engineering Search
S27	-	Assessing the quantum-computing landscape
S28	-	Bugs in Quantum computing platforms: an empirical study
S29	-	Towards Quantum-algorithms-as-a-service
S30	-	Embracing iterations in Quantum software: a vision

S31	-	Automatic generation of test circuits for the verification of Quantum deterministic algorithms
S32	[75]	Asserting the correctness of Shor implementations using metamorphic testing
S33	-	The Quantum software lifecycle
S34	-	A Backend-agnostic, Quantum-classical Framework for Simulations of Chemistry in C++
S35	[66]	Towards a Quantum Software Modeling Language
S36	-	Quantum Software: Model-driven or Search-driven? A Q-SE 2021 Workshop Report
S37	-	Quantum Annealing-Based Software Components: An Experimental Case Study with SAT Solving
S38	-	Hybrid quantum-classical problem solving in the NISQ era
S39	-	An empirical study on the current adoption of quantum programming
S40	-	Using Quantum computers to speed up dynamic testing of software
S41	-	Quantum optimization for fast CAN bus intrusion detection
S42	-	About a criterion of successfully executing a circuit in the NISQ era: what wd $\ll 1/\epsilon$ eff really means
S43	-	Towards practical quantum applications using hybrid problem solving techniques
S44	[76]	QMutPy: a mutation testing tool for Quantum algorithms and applications in Qiskit
S45	[77]	QDiff: differential testing of quantum software stacks
S46	[3]	Quantum Algorithm Implementations for Beginners
S47	-	Programming quantum computers using 3-D puzzles, coffee cups, and doughnuts
S48	-	Progress and Prospects of Quantum Algorithms
S49	[78]	Property-based Testing of Quantum Programs in Q#
S50	[79]	QuSBT: search-based testing of quantum programs
S51	-	Quito: a coverage-guided test generator for quantum programs
S52	-	Silq2Qiskit - Developing a quantum language source-to-source translator
S53	-	Time-optimal Qubit mapping
S54	-	Quantum Hoare Logic with Classical Variables
S55	-	QuaFL: a typed DSL for quantum programming
S56	-	Invariants of quantum programs: characterizations and generation

Table 6. Papers selected from IEEE Xplore

ID_SR	ID_REF	PAPER TITLE
S55	[27]	Developing Programming Assignments for Teaching Quantum Computing and Quantum Programming
S56	-	Understanding Quantum Software Engineering Challenges An Empirical Study on Stack Exchange Forums and GitHub Issues
S57	-	Poster: Automatically Solving NP-Complete Problems on a Quantum Computer
S58	-	Evolution of Quantum Computing: A Systematic Survey on the Use of Quantum Computing Tools
S59	-	Towards Model-Driven Quantum Software Engineering
S60	-	Modelling Quantum Circuits with UML
S61	-	Quantum Computing: State-of-Art and Challenges
S62	-	1-2-3 Reproducibility for Quantum Software Experiments
S63	[80]	A Preliminary Study on Generating Well-Formed Q# Quantum Programs for Fuzz Testing
S64	[52]	Quantum Computing Platforms: Assessing the Impact on Quality Attributes and SDLC Activities
S65	-	A Tool For Debugging Quantum Circuits
S66	-	Quantum Software Models: The Density Matrix for Classical and Quantum Software Systems Design
S67	-	Quantum-Inspired Immune Memory Algorithm for Self-Structuring Antenna Optimization

S68	-	An Improved Multi-Objective Quantum Genetic Algorithm Based on Cellular Automaton
S69	-	Improved Quantum-Inspired Evolutionary Algorithm and Its Application to 3-SAT Problems
S70	-	Quantum Approximation on Some Classes of Multivarite Functions
S71	-	A Formal Derivation of Grover's Quantum Search Algorithm
S72	-	Explaining the Implicit Parallelism of Genetic Algorithm and Computational Complexity by Quantum Theory
S73	-	Quantum Machine Learning for Software Supply Chain Attacks: How Far Can We Go?
S74	-	Quantum phase estimation-based algorithms for machine learning
S75	-	A Review of Quantum Cybersecurity: Threats, Risks and Opportunities
S76	-	Metamorphic Testing of Oracle Quantum Programs
S77	-	Advance quantum based binary neural network learning algorithm
S78	-	An implementation of compact genetic algorithm on a quantum computer
S79	-	Quantum genetic algorithm optimized BP neural network for high-resolution remote sensing image classification
S80	-	Real-Coded Quantum Evolutionary Algorithm Based on Immune Theory for Multi-modal Optimization Problems
S81	[81]	Q Bugs: A Collection of Reproducible Bugs in Quantum Algorithms and a Supporting Infrastructure to Enable Controlled Quantum Software Testing and Debugging Experiments
S82	-	Some Size and Structure Metrics for Quantum Software
S83	-	Studies on the Quality Assessment of PRNG for Q-Trits Quantum Cryptography Protocols
S84	-	Accelerator circuits for quantum simulation
S85	-	Generalised Quantum Tree Search
S86	-	Multilevel Minimum Cross Entropy Threshold Selection Based on Quantum Particle Swarm Optimization
S87	-	Quantum network based on multiparty quantum secret sharing
S88	-	Quantum Determined Key Distribution Scheme Using Quantum Teleportation
S89	-	Learning quantum operator by quantum adiabatic computation
S90	[82]	Mutation Testing of Quantum Programs Written in QISKit
S91	-	Quantum Software Engineering Supremacy in Intelligent Robotics
S92	-	Compiling Quantamorphisms for the IBM Q Experience
S92	-	Is Your quantum Program Bug-Free?
S93	-	A Quantum Algorithm for Ray Casting using an Orthographic Camera
S94	-	Insight into the operation of NTRU and a comparative study of NTRU, RSA and ECC public key cryptosystems
S95	-	Visualizing time dependent semantics: an application to quantum algorithms
S96	-	Software requirements selection using Quantum-inspired Elitist Multi-objective Evolutionary algorithm
S97	-	Software requirements selection using Quantum-inspired Multi-objective Differential Evolution Algorithm
S98	-	Experimental Implementation of Discrete Time Quantum Walk with the IBM Qiskit Library
S99	-	Queueing theory study of round robin versus priority dynamic quantum time round robin scheduling algorithms
S100	-	Quantum Software as a Service Through a Quantum API Gateway
S101	[63]	Mutation Testing of Quantum Programs: A Case Study with Qiskit
S102	-	An Empirical Study of Optimizers for Quantum Machine Learning
S103	-	Performance Evaluation of TLS 1.3 Handshake on Resource-Constrained Devices Using NIST's Third Round Post-Quantum Key Encapsulation Mechanisms and Digital Signatures

S104	-	Investigating Quantum Cause-Effect Graphs
S105	-	Quantum homomorphic signature using coherent states
S106	-	Approaches to Overpower Proof-of-Work Blockchains Despite Minority
S107	-	Fake News Detection: An Application of Quantum K-Nearest Neighbors
S108	-	Nitride Semiconductor Quantum Dots - Mathematical Models of the Electronic Spectrum and Methods for its Simulation
S109	-	On quantum methods for machine learning problems, part I: Quantum tools
S110	-	On quantum methods for machine learning problems, part II: Quantum classification algorithms
S111	-	A Flowchart Language for Quantum Programming
S112	-	Training of Process Neural Networks Based on Improved Quantum Genetic Algorithm
S113	-	A novel quantum behaved Particle Swarm optimization algorithm with chaotic search for image alignment
S114	-	Quantum-Behaved Particle Swarm Optimization with Normal Cloud Mutation Operator
S115	[83]	Research on method for adding new friends to instant messaging system based on SIP in quantum communication network
S116	-	Quantum Annealing Approach for the Optimal Real-time Traffic Control using QUBO
S117	-	Optimizing Quantum Programs Against Decoherence: Delaying Qubits into Quantum Superposition
S118	-	An Efficient Controlled Quantum Secure Direct Communication Protocol via GHZ-like States
S119	-	Bugs4Q: A Benchmark of Real Bugs for Quantum Programs
S120	-	Hybrid Quantum Applications Need Two Orchestrations in Superposition: A Software Architecture Perspective
S121	-	Reverse Engineering of Hamiltonian Expressions from D-Wave programs
S122	-	A simulator-based performance analysis of multilevel feedback queue scheduling
S123	-	Remarks on concavity of the auxiliary function appearing in quantum reliability function
S124	-	DQRA: Deep Quantum Routing Agent for Entanglement Routing in Quantum Networks
S125	-	A practical quantum public-key encryption model
S126	-	Quantum-Inspired Evolutionary Algorithms applied to numerical optimization problems
S127	[12]	Identifying Bug Patterns in Quantum Programs
S128	-	Performance Comparison of Population-Based Quantum-Inspired Evolutionary Algorithms
S129	-	A Hybrid Adaptive Quantum Behaved Particle Swarm Optimization Algorithm Based Multilevel Thresholding for Image Segmentation
S130	-	GCQW: A Quantum Walk Model for Predicting Missing Links of Complex Networks
S131	-	A Video Steganography Scheme Based on Post-Quantum Cryptography
S132	[24]	A Multi-Lingual Benchmark for Property-Based Testing of Quantum Programs
S133	-	Enhancing data and privacy security in mobile cloud computing through quantum cryptography
S134	-	Quantum Information Security Protocols and Quantum Coding Theory
S135	-	Improved Security of SDN based on Hybrid Quantum Key Distribution Protocol
S136	-	Synthesizing hybrid quantum circuits without ancilla qudits
S137	-	Convergence analysis on a class of quantum-inspired evolutionary algorithms
S138	-	Quantum Control Mechanisms of Superdense Coding
S139	-	Quantum particle swarm optimization for multiobjective combined economic emission dispatch problem using cubic criterion function

S140	-	A hybrid Quantum-Inspired Evolutionary Algorithm for open vehicle routing problem
------	---	---

Информация об авторах / Information about authors

Рейес ХУАРЕС-РАМИРЕС имеет степень PhD по программированию, профессор Автономного университета Нижней Калифорнии (Мексика) с 2002 года. Эксперт в области программной инженерии, в настоящее время является Президентом Мексиканской сети программной инженерии – мексиканской профессиональной ассоциации, продвигающей научные и образовательные инициативы в области программной инженерии. Сфера научных интересов: программная инженерия, человеко-машинное взаимодействие. В последнее время занялся изучением основ квантовых вычислений. Участник Мексиканской национальной исследовательской системы.

Reyes JUÁREZ-RAMÍREZ – PhD in Computer Science, professor at the Universidad Autónoma de Baja California since 2002. Software Engineering expert; he currently is the president of the Red Mexicana de Ingeniería de Software (“Mexican Network of Software Engineering”), a specialized association that addresses research and education initiatives in software engineering in México. His research areas are software engineering, human-computer interaction, and currently starting with quantum computing. He is part of the National Researchers System in Mexico.

Христиан Ксавьер НАВАРРО-КОТА имеет степень PhD по программированию от испанского Университета Кастилии в Ла-Манче. В настоящее время работает профессором Автономного университета Нижней Калифорнии в городе Энсенาดา, штат Нижняя Калифорния, Мексика. Его научные интересы включают образовательные технологии, мобильные и повсеместные вычисления, человеко-машинное взаимодействие, исследования опыта взаимодействия.

Christian Xavier NAVARRO-COTA – PhD in Computer Science from the Universidad de Castilla La Mancha, Spain. Currently, he serves as a professor at Universidad Autónoma de Baja California in Ensenada, Baja California, México His research interests include educational technology, mobile and ubiquitous computing, human-computer interaction, and user experience (UX).

Саманта ХИМЕНЕС имеет основное образование в области вычислительной техники, степень магистра в области обработки данных и степень PhD по программированию. Она является профессором в мексиканском Автономном университете Нижней Калифорнии, город Валье-де-лас-Пальмас, Нижняя Калифорния, Мексика, а также приглашенным преподавателем в Университете Глобальных знаний, Сан-Диего, участвует в мексиканской Национальной системе поддержки исследователей. Сфера её научных интересов: человеко-машинное взаимодействие, диалоговые системы, аффективные вычисления и образовательные системы.

Samantha JIMÉNEZ – PhD in Computer Science and a master’s in data science, complemented by a solid foundation in Computational Systems Engineering. She is a professor at the Universidad Autónoma de Baja California located in Valle de las Palmas, Baja California, México, and a dedicated educator at San Diego Global Knowledge University. Her research interests are human-computer interaction, dialogue systems, affective computing, and educational systems. She is part of the National Researchers System in Mexico.

Алан Давид РАМИРЕС-НОРЬЕГА получил степень магистра прикладных вычислений в Автономном университете Синалоа и степень PhD по программированию в Автономном университете Нижней Калифорнии. Он является штатным профессором и исследователем в Университете Синалоа, членом мексиканской Национальной системы поддержки

исследователей уровня 1 в области IX (междисциплинарные исследования). Имеет несколько публикаций в известных журналах (JCR, SCOPUS), выступал на национальных и международных конференциях по темам, связанным с интеллектуальными системами обучения, разработкой программного обеспечения и добычей данных, последняя из которых является основной областью интересов. Кроме того, руководил подготовкой бакалавров, магистров и докторов по различным направлениям и специальностям.

Alan David RAMÍREZ-NORIEGA – Obtained his master's degree in applied computing from the Autonomous University of Sinaloa and his Ph.D. in computer science from the Universidad Autónoma de Baja California. He is a Full-Time Professor and Researcher at the Facultad de Ingeniería Mochis at the Universidad Autónoma de Sinaloa. He is currently a member of the National System of Researchers level 1 in area IX (Interdisciplinary). He has several publications in high-impact journals (JCR, SCOPUS) and national and international conferences on topics related to Intelligent Tutoring Systems, Software Engineering and Data Mining, the latter being the main areas of interest. In addition, he has participated in various directions and synodalities of Bachelor's, Master's, and Doctorate theses.

Ма Вероника ТАПИЯ-ИБАРРА – инженер, основное образование – вычислительные инженерные системы, завершившееся получением степени магистра. Работает профессором в Технологическом институте Леона, Гуанахуато, Мексика, где преподает курсы по вычислительной технике и программированию.

Ma Veronica TAPIA-IBARRA – She is an engineer with a solid foundation in Computational Systems Engineering, complemented by master's degree studies. She is a professor at the TecNM-Instituto Tecnológico de León located in León, Guanajuato, México, teaching hardware and programming courses.

Сесар Артуро ГЕРРА-ГАРСИЯ – имеет ученую степень PhD Университета Кастилии, Ла-Манча, Испания. Профессор Автономного университета Сан-Луис-Потоси, член мексиканской Национальной системы поддержки исследователей. Сфера научных интересов: инженерия требований, программная инженерия, качество данных и информации, научное волонтерство и информационная безопасность.

César Arturo GUERRA GARCÍA – PhD in computer science, graduated from Universidad de Castilla-La Mancha, Spain. He is a professor at the Universidad Autónoma de San Luis Potosí. His research interests are requirements engineering, software engineering, data and information quality, citizen science and informatic security. He is part of the National Researchers System in México.

Эктор Херардо ПЕРЕС-ГОНСАЛЕС – штатный профессор-исследователь Автономного университета Сан-Луис-Потоси (Мексика), имеет ученую степень доктора компьютерных наук. Автор научных статей и глав в книгах по автоматизации проектирования программного обеспечения и человеко-машинного взаимодействия, выступал с научными докладами на международных конференциях в США, Канаде, Великобритании, Португалии и в Сингапуре, участвует в мексиканской Национальной системе поддержки исследователей. Область научных интересов: проектирование программного обеспечения, преподавание методов разработки программного обеспечения, обработка цифровых изображений, разработка программного обеспечения для квантовых компьютеров.

Hector Gerardo PEREZ-GONZALEZ – Full-time research professor at Universidad Autónoma de San Luis Potosi, Mexico. PhD in Computer Science from the University of Colorado in 2003. Author of research articles and book chapters on Automatic Software Design and Human-Computer Interaction. He has been a speaker at international conferences in the USA, Canada, UK, Portugal,

and Singapore. His research areas are software design, computer science education, and quantum software engineering. He is a member of the National Researchers System in Mexico.

Карлос Альберто ФЕРНАНДЕС-И-ФЕРНАНДЕС имеет степень PhD университета Шеффилда по программированию, эксперт в области программирования. В настоящее время возглавляет Институт вычислений в Технологическом университете в мексиканском регионе Миштека, координирует магистерские программы по прикладным аспектам вычислительных технологий. Сфера научных интересов: визуальное моделирование, гибкие технологии разработки и формальные спецификации программного обеспечения.

Carlos Alberto FERNÁNDEZ-Y-FERNÁNDEZ – Software Engineering expert with a Ph.D from the University of Sheffield. He currently leads the Institute of Computing at Universidad Tecnológica de la Mixteca and coordinates the Master's program in Applied Computing Technologies. His research interests include visual modeling, agile methods, and formal software specification.