QUANTUM SOFTWARE ENGINEERING:

PAST, PRESENT & FUTURE

Authors

Giuseppe Bisicchia, José Garcia-Alonso,
Juan M. Murillo, and Antonio Brogi

"What kind of computer are we going to use to simulate physics?" It was Nobel laureate Richard Feynman who raised this question in his visionary speech to the Department of Physics at the California Institute of Technology in 1982, beginning the history of quantum computing.[1,2]

This question is rooted in a series of crises and revolutions that shook the world of physics to its foundations between 1900 and 1925. The result of that tumultuous period was a theory of physics that describes the behavior of nature at subatomic levels: *quantum mechanics*.

In the 1980s, Yuri Manin and Feynman, among others, were primarily concerned about the difficulties of modeling quantum systems. In such systems, the number of variables required to represent them increases exponentially with their complexity and with the number of particles involved.[3]

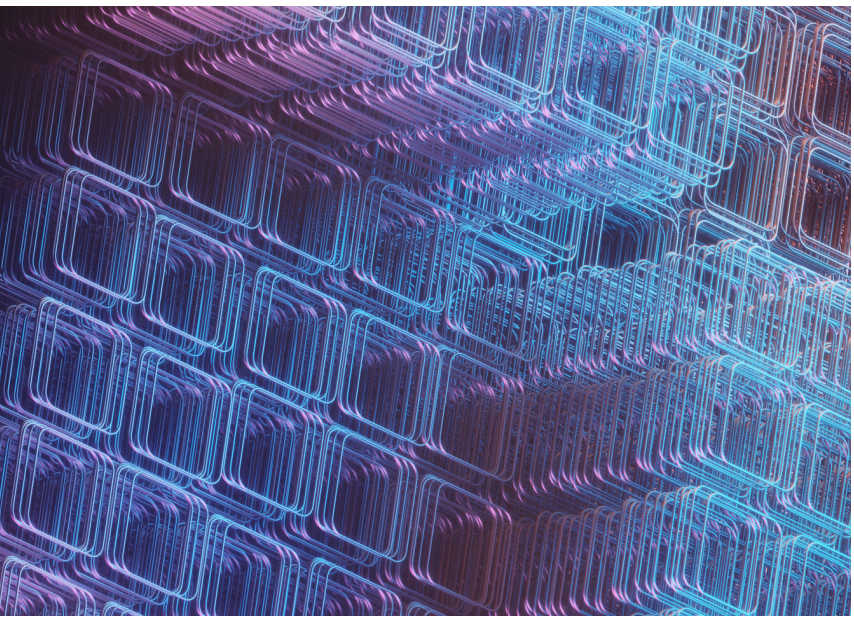
In 1985, physicist David Deutsch, in his seminal work, suggested a deeper connection between computing and physics, stating a stronger "physical version" of the Church-Turing thesis.[4] This thesis, called the "Church-Turing-Deutsch principle," states that: "Every finitely realizable physical system can be perfectly simulated by a universal model computing machine operating by finite means."[5]

With this interpretation, Deutsch brought attention to an often-neglected fact about computation. Every algorithm is performed by a physical system, whether that's an electronic calculator, a mechanical apparatus, or a human being. Computation is ultimately a physical process, so a *universal* computer (that is also a physical system) must be able to simulate the dynamics of every possible physical system.

The consequences of the physics revolution in the early 20th century led scientists to postulate that the fundamental nature of physics is ultimately quantum mechanical. Unfortunately, classical systems seem to be ineffective in efficiently simulating quantum mechanical systems. Deutsch then proposed a universal computing device based on the principles of quantum mechanics to overcome the limitations of classical computers, and the quantum computer was born.

> **EVERY ALGORITHM IS PERFORMED BY A PHYSICAL SYSTEM, WHETHER THAT'S AN ELECTRONIC CALCULATOR, A MECHANICAL APPARATUS, OR A HUMAN BEING**

Soon, the potential of quantum computers began to be, as Deutsch surmised, far more impactful than just simulating physical systems. In 1992, Deutsch, in collaboration with Richard Jozsa, formulated a problem that (even if of little practical interest) can be solved more efficiently by quantum devices than by any classical or stochastic algorithm. In 1993, Ethan Bernstein and Umesh Vazirani proposed another problem that showed the advantage of quantum devices over classical ones, even when small errors are allowed. In the same work, Bernstein and Vazirani designed a quantum version of the Fourier transform.[6]



In 1994, leveraging the quantum Fourier transform and the work of Daniel Simon, who showed that a quantum computer could find the period of a function with an exponential speedup, Peter Shor presented an efficient quantum algorithm for computing discrete logarithms. Only a few days later, Shor formulated an efficient quantum algorithm for factoring large numbers, too. Both problems are believed to be intractable on classical computers and are commonly used in cryptographic protocols.

Just two years later, Seth Lloyd proved quantum computers could simulate quantum systems without the exponential overhead present in classical simulations, confirming Feynman's 1982 conjecture. In the same year, Lov Grover presented a quantum algorithm achieving an optimal quadratic speedup for unstructured search. Shor and Grover's breakthroughs proved a strong impetus to research quantum algorithms, demonstrating the existence of useful problems that benefit from a quantum speedup.

Meanwhile, research into working quantum computers began in earnest. In 1993, Lloyd proposed a method for building a potentially realizable quantum computer through electromagnetic pulses. In 1995, Juan Cirac and Peter Zoller suggested an implementation of a quantum computer employing cold ionized atoms. One year later, David DiVincenzo formalized five minimal requirements for creating a working quantum computer. They include the availability of scalable qubits highly isolated from the external environment; the ability to initialize, manipulate, and entangle their state; and the ability to "strongly" measure the state of each qubit.[7-9] A further advance came from Yasunobu Nakamura and collaborators between 1991 and 2001 in the form of a working, controllable superconducting qubit.

During those years, however, decoherence threatened to dash any hopes of ever having usable quantum computers. Decoherence is the phenomenon that, under typical conditions, prevents complex many-particle quantum systems from exhibiting quantum behavior for a long time, stranding the dream of a quantum computer. Once again, it was Shor who offered hope and brought new life to the field. In 1995, he demonstrated that it was possible to reduce the destructive effects of decoherence through the quantum analogue of error-correcting codes and fault-tolerant methods for executing reliable quantum computations on noisy quantum computers.[10,11]

The work of Shor and subsequent researchers confirmed that it is possible, at least in principle, to suppress the error rate of a quantum computer to arbitrarily low levels, thanks to error correction schemes (and as long as the error rate is below a certain threshold).[12] This is called the "threshold theorem."

Significant developments have been made since those first steps in quantum software and hardware. In 2011, the first commercially available quantum computer was presented by D-Wave: the D-Wave One, a 128-qubit quantum annealer.[13] In 2016, IBM put its first five-qubit, gate-based superconducting quantum computer online, making quantum computing publicly available through the cloud.[14]

In 2018, the first commercial quantum computer employing trapped ions was launched by IonQ. A year later, Google claimed the achievement of quantum supremacy with Sycamore, its 54-qubit, superconducting processor.[15] However, doubts arose shortly afterward and, eventually, classical devices beat Google's result.

The last record in the quantum race was set in 2023 by IBM, which announced evidence for the utility of quantum computing even with noisy hardware, showing it is possible to produce reliable results even without fault-tolerant quantum computers and at a scale beyond brute-force classical computation. However, the scientific community does not entirely agree.

Although the supremacy and utility of quantum computers have not yet been established beyond a shadow of a doubt, there is no denying we are at the gates of a new era.

Even if quantum and classical computers feature the same computational power (i.e., they can solve the same class of problems), it is believed (and some evidence has arisen) that quantum computers can solve some problems asymptotically faster than what is possible with classical resources.

In fact, cutting-edge applications are emerging, promising to revolutionize numerous industries and sectors (and with a potentially immeasurable impact on society). Among the most researched areas are medicine, chemistry and pharmacy, biology and agriculture, engineering, energy and logistics, economy and finance, meteorology, manufacturing, and cybersecurity.

THE DAWN OF QSE

Despite recent progress, current quantum computers cannot scale beyond dimensions of a few tens (or in the best cases, hundreds) of qubits. Quantum devices are also very sensitive to external interference (noise), which can easily disrupt an ongoing computation. Because of these limitations, quantum computers are usually referred to as "noisy intermediate-scale quantum devices," highlighting their capacity to execute only quantum programs featuring a small number of qubits and consecutive steps.

However, this is not the first time in history that computer scientists have faced limitations on computing devices. Several authors compare the current quantum computing landscape to that of classical computing during the 1960s and argue for a similar roadmap.[16]

The idea is to view the primary role of quantum software engineering (QSE) as exploiting the full potential of commercial quantum computer hardware when it arrives.[17] In that role, QSE will define the best quantum software development and application management lifecycles. It will also coherently employ and operate quantum methodologies and tools as they are developed.

Researcher and quantum expert Jianjun Zhao emphasizes that adopting proven engineering methods in quantum software isn't just about technology — it's a strategic move for businesses. Organizations can transform complex quantum capabilities into reliable, efficient, profitable solutions by carefully designing, building, and managing quantum software with discipline and purpose. This empowers companies to tap into quantum computing's full potential, translating innovation into tangible competitive advantage and sustained business growth while delivering real-world impact.[18]

Some experts believe quantum computing will lead to a golden age of software engineering. They believe software engineering already provides proven methods and best practices that can accelerate quantum software development. Businesses entering the quantum space should certainly leverage these established approaches to reduce risks and enhance productivity. However, quantum software has unique challenges, creating opportunities to develop specialized techniques. Recognizing this balance between proven practices and innovation is key to success in QSE.[19]

A good example is the "Talavera Manifesto for Quantum Software Engineering and Programming," a foundational document summarizing essential principles and commitments that guide the emerging field of QSE. Its importance lies in clearly defining the conceptual framework and best practices for developing robust, reliable quantum software, thus providing a common ground for researchers and practitioners worldwide.

The manifesto is considered by many researchers as a milestone because it marks one of the earliest organized efforts to formalize the core values, goals, and standards within the relatively young discipline of QSE. Researchers and practitioners can leverage the Talavera Manifesto by adopting its principles as a baseline, extending its guidelines, and systematically applying them to future quantum software development projects as a way to push QSE toward greater maturity and practical impact.[20,21] It has been signed by more than 200 researchers and practitioners from more than 20 countries.[22]

THE FUTURE OF QSE

QSE will not replace classical software engineering; it will coexist and integrate with it. As quantum computers progress from research prototypes to production-ready platforms, we anticipate hybrid quantum-classical pipelines becoming standard practice. Thus, developing robust methodologies and frameworks that seamlessly combine the two paradigms will be essential:

- **Hybrid architectures and workflows.** Many quantum algorithms depend on iterative procedures in which a classical computer is used to run optimization loops that feed results back to a quantum device. Formalizing best practices in designing, implementing, and optimizing these hybrid workflows could help unify quantum and classical software engineering. This might include new software lifecycle models that explicitly account for quantum-classical feedback and optimize data exchange between the two worlds.
- **API standardization.** With multiple quantum hardware manufacturers offering cloud-based APIs, there is a clear need for interoperability standards.[23] Standardized APIs and data exchange protocols for quantum backends could lower the learning curve and prevent vendor lock-in, accelerating broader adoption of quantum solutions.
- **Toolchain integration.** Transitioning from proof-of-concept quantum code to enterprise-grade applications will require tight integration of quantum development tools (e.g., high-level domain-specific languages, simulators, and compilers) into established continuous integration/continuous delivery pipelines.[24] Ensuring compatibility with classical development tools (e.g., continuous integration servers, version-control systems, and automated testing suites) will reduce friction for developers and enable more mature software engineering practices in the quantum domain.

LANGUAGE ABSTRACTIONS & HIGHER-LEVEL PRIMITIVES

To build on the impetus to move away from low-level gate operations, QSE will need domain-specific languages and libraries that cater to specific application areas, ranging from quantum chemistry simulations to quantum machine learning (ML):

- **Domain-specific quantum languages.** Specialized libraries for quantum chemistry, finance, ML, or cryptography will eliminate the need for developers to understand the details of quantum gate manipulation. Developers will benefit from libraries that speak the domain's "language," making quantum development more accessible to subject matter experts without deep quantum expertise.
- **Declarative quantum programming.** Instead of explicitly describing how to manipulate qubits and gates, developers will be able to focus on the "what" of the problem. Declarative quantum languages (in which one specifies the desired outcome or high-level algorithmic structure) could help shift quantum coding from a specialized skill to a more universally approachable paradigm.
- **Automation and optimization.** High-level abstractions will inevitably be matched with sophisticated compilers and optimizers capable of translating abstract quantum instructions into efficient gate-level operations. These compilers

may use AI-driven optimizations, iteratively learning to compile code for different quantum architectures and hardware constraints.

QUANTUM SOFTWARE DEBUGGING, VISUALIZATION & VERIFICATION

Although debugging on quantum hardware remains intrinsically challenging, continued research may yield innovative approaches that enable practical, rigorous testing:

- **Advanced visualization techniques.** Beyond standard circuit diagrams, we may see the development of 3D or interactive visual models that depict qubit interactions, entanglement patterns, and error propagation in real time. Such immersive techniques could aid developers in pinpointing the root causes of unexpected behavior.
- **Probabilistic debugging methods.** Given the nondeterministic nature of quantum measurement, debugging tools could rely on statistical methods to gather information about the system's state. This approach may involve repeated runs of the same circuit under different conditions or sampling a subset of qubits to minimize measurement disturbances.
- **Formal verification for quantum systems.** Borrowing principles from classical formal verification, quantum program verification could involve the use of formal logic systems and model checking specialized for quantum. The aim would be to mathematically prove certain properties (correctness, security, or reliability) without requiring a full measurement of the quantum state. As quantum programs scale in complexity, such methods may become indispensable to ensure correctness in safety-critical applications.

DISTRIBUTED & HETEROGENEOUS QUANTUM COMPUTATIONS

As quantum computers diversify in qubit implementation (e.g., superconducting, ion-trap, photonic), harnessing that heterogeneity through distributed quantum computing could become a crucial strategy:

- **Networked quantum environments.** Research on quantum networks and interconnects is already advancing, pointing to a future where qubits can be transferred or teleported between remote quantum processors. Such quantum networks would enable multi-computer protocols, distributed entanglement, and resource sharing, effectively increasing overall computational capacity.
- **Task-oriented compilers and schedulers.** In a world where multiple quantum backends exist, each with unique advantages (speed, fidelity, qubit count, connectivity), specialized compilers and schedulers could dynamically partition programs.[25] Some qubits or tasks could be offloaded to a superconducting processor for specific gates; others might be reserved for an ion-trap system that excels at different operations. This approach has parallels to high-performance computing frameworks in which tasks are distributed among central processing units, graphics processing units, and other accelerators.
- **Runtime adaptation.** Quantum hardware is prone to noise and varying fidelity across qubits. A future quantum runtime environment might adapt in real time — monitoring error rates and automatically routing subtasks to the most reliable qubits or devices across a distributed network.[26] This adaptive orchestration could significantly enhance performance and reliability.

INTO THE QUANTUM FUTURE

As quantum hardware advances, the demand for robust, scalable, developer-friendly tools and practices will intensify. Tackling these challenges will require a concerted effort across academia, industry, and government and between physicists, computer scientists, and software engineers. The pathways we've outlined (high-level language abstractions, distributed quantum systems, adaptive runtime environments, and quantum DevOps) hint at the multifaceted nature of this emerging discipline.

In the next few years, breakthroughs in quantum hardware fidelity and qubit count will undoubtedly usher in unanticipated applications. To clear these hurdles, QSE will need to stay agile, incorporating novel computational models and addressing newly uncovered ethical and security concerns. Successful paradigm shifts in computing tend to be driven by accessible abstractions, robust tooling, and a rich ecosystem of supportive infrastructures. For quantum computing, creating this ecosystem is not merely a challenge, it is a profound opportunity to shape a new technological frontier.

REFERENCES

[1] Feynman, Richard P. "Simulating Physics with Computers." *International Journal of Theoretical Physics*, Vol. 21, June 1982.

[2] To be fair, a few years earlier, in 1980, mathematician Yuri Manin pointed out the problems in simulating quantum systems with classical computers. Independently, between 1980 and 1982, physicist Paul Benioff proposed a quantum mechanical model of Turing machines, a topic also discussed by Feynman in 1985.

[3] On the other hand, the number of variables to describe classical physical systems grows only linearly with their complexity.

[4] Here is a version formulated by Deutsch for completeness: "Every function which would naturally be regarded as computable can be computed by the universal Turing machine."

[5] Deutsch, David. "Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer." *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 400, No. 18188, July 1985.

[6] The quantum Fourier transform serves as the quantum counterpart to the discrete Fourier transform, offering an efficient quantum algorithm for implementing Fourier transform operations. Crucial in various quantum algorithms, it plays a fundamental role in extracting and translating purely quantum information stored within qubits into classically measurable outcomes.

[7] A qubit is the computational unit of a quantum computer (as opposed to the classical bit). A qubit state can be 0, 1, or in a superposition (i.e., linear combination) of both. In the latter, when measured, it will be only 0 or 1, with different probabilities according to its superposition.

[8] "Strong" measurement means a measure capable of collapsing the state of a qubit.

[9] DiVincenzo, D.P. "Topics in Quantum Computers." In *Mesoscopic Electron Transport*, edited by L.L. Sohn, L.P. Kouwenhoven, and G. Schön. Springer, Dordrecht, 1997.

[10] Shor, Peter W. "Scheme for Reducing Decoherence in Quantum Computer Memory." *Physical Review A*, Vol. 52, October 1995.

[11] Shor, Peter W. "Fault-Tolerant Quantum Computation." *Proceedings of 37th Conference on Foundations of Computer Science*. IEEE Computer Society Press, 1996.

[12] The assumptions made regarding the computational capability have a significant impact on the precise value of the threshold, but it is considered in the range 10^{-4} to 10^{-6}.

[13] A quantum annealer is a specialized form of quantum computer. Unlike universal quantum computers, quantum annealers are non-Turing, complete devices tailored specifically for solving optimization problems such as energy minimization problems. Roughly speaking, quantum annealers ensure that each qubit eventually settles into a classical state that reflects the minimum energy configuration of the problem.

[14] Unlike quantum annealers, gate-based quantum computers are universal computing machines. A gate-based quantum computer operates by manipulating qubits through the quantum analogue of classical logical gates.

[15] Quantum supremacy refers to the goal of performing tasks with controlled quantum systems, going beyond what can be achieved with ordinary digital computers.

[16] Moguel, Enrique, et al. "A Roadmap for Quantum Software Engineering: Applying the Lessons Learned from the Classics." *Proceedings of the 1st International Workshop on Software Engineering & Technology (Q-SET'20)*. CEUR Workshop Proceedings, 2020.

[17] Clark, John, and Susan Stepney. "Quantum Software Engineering." University of York, Wayback Machine, accessed 2025.

[18] Zhao, Jianjun. "Quantum Software Engineering: Landscapes and Horizons." Cornell University, 31 December 2021.

[19] Piattini, Mario, Guido Peterssen, and Ricardo Pérez-Castillo. "Quantum Computing: A New Software Engineering Golden Age." *ACM SIGSOFT Software Engineering Notes*, Vol. 45, No. 3, July 2020.

[20] Piattini, Mario, et al. "The Talavera Manifesto for Quantum Software Engineering and Programming." *Proceedings of the 1st International Workshop on the QuANtum SoftWare Engineering & pRogramming (QANSWER).* CEUR Workshop Proceedings, 2020.

[21] De Stefano, Manuel, et al. "The Quantum Frontier of Software Engineering: A Systematic Mapping Study." *Information and Software Technology*, Vol. 175, November 2024.

[22] "Talavera Manifesto's Endorsers." aQuantum Software Engineering, accessed 2025.

[23] Bisicchia, Giuseppe, et al. "From Quantum Software Handcrafting to Quantum Software Engineering." *2024 IEEE International Conference on Software Analysis, Evolution and Reengineering — Companion (SANER-C).* IEEE, 2024.

[24] Bisicchia, Giuseppe, et al. "Distributing Quantum Computations, by Shots." In *Service-Oriented Computing*, edited by Flavia Monti et al. Springer, 2023.

[25] Bisicchia, Giuseppe, et al. "Dispatching Shots Among Multiple Quantum Computers: An Architectural Proposal." *2023 IEEE International Conference on Quantum Computing and Engineering (QCE).* IEEE, 2023.

[26] Bisicchia, Giuseppe, et al. "Distributing Quantum Computations, Shot-Wise." Cornell University, 25 November 2024.

About the authors

Giuseppe Bisicchia is a PhD student in computer science, jointly pursuing his degree at the University of Pisa, Italy, and the University of Extremadura, Spain. His research interests span multiple domains, including quantum computing, cloud computing, software engineering, and the Internet of Things (IoT). Mr. Bisicchia also serves as a teaching assistant for a cloud computing course at the University of Pisa and has been actively involved in mentoring introductory programming courses for middle school students. He is also a longstanding volunteer mentor at a local CoderDojo, inspiring young learners to explore the world of coding and technology. Mr. Bisicchia earned a dual master of science degree in computer science, graduating cum laude, from both the University of Pisa and the University of Málaga, Spain. He can be reached at giuseppe.bisicchia@phd.unipi.it.

José Garcia-Alonso is Associate Professor at the University of Extremadura, Spain. In the last 10 years, his research interests have focused on quantum software engineering, e-health continuum, and gerontechnology. These interests have led to numerous publications in indexed journals and international conference proceedings, as well as participation in several research projects. Dr. Garcia-Alonso has also contributed to the founding of three technology-based companies. He earned a PhD in computer science from the University of Extremadura. He can be reached at jgaralo@unex.es.

Juan M. Murillo is Professor of Software Engineering at the University of Extremadura, Spain, and cofounder of start-ups Gloin and Viable. His research interests include quantum software engineering, software architectures, mobile computing, and quantum computing. Dr. Murillo earned a PhD in computer science from the University of Extremadura. He can be reached at juanmamu@unex.es.

Antonio Brogi is Professor of Computer Science at the University of Pisa, Italy, and leads the service-oriented, cloud, and fog computing research group. He also serves as coordinator of the computer science PhD program jointly offered by the Universities of Pisa, Florence, and Siena. Dr. Brogi's research interests span software engineering, symbolic AI, cloud-edge computing, sustainability and ICT, and quantum software engineering. He has published his research results in approximately 300 papers for international journals and conference proceedings. Dr. Brogi earned a PhD in computer science from the University of Pisa. He can be reached at antonio.brogi@unipi.it.