23rd International Conference on
Software Engineering and Formal Methods

SEFM 2025

Quantum Software in Action
Challenges and Opportunities in Software Engineering

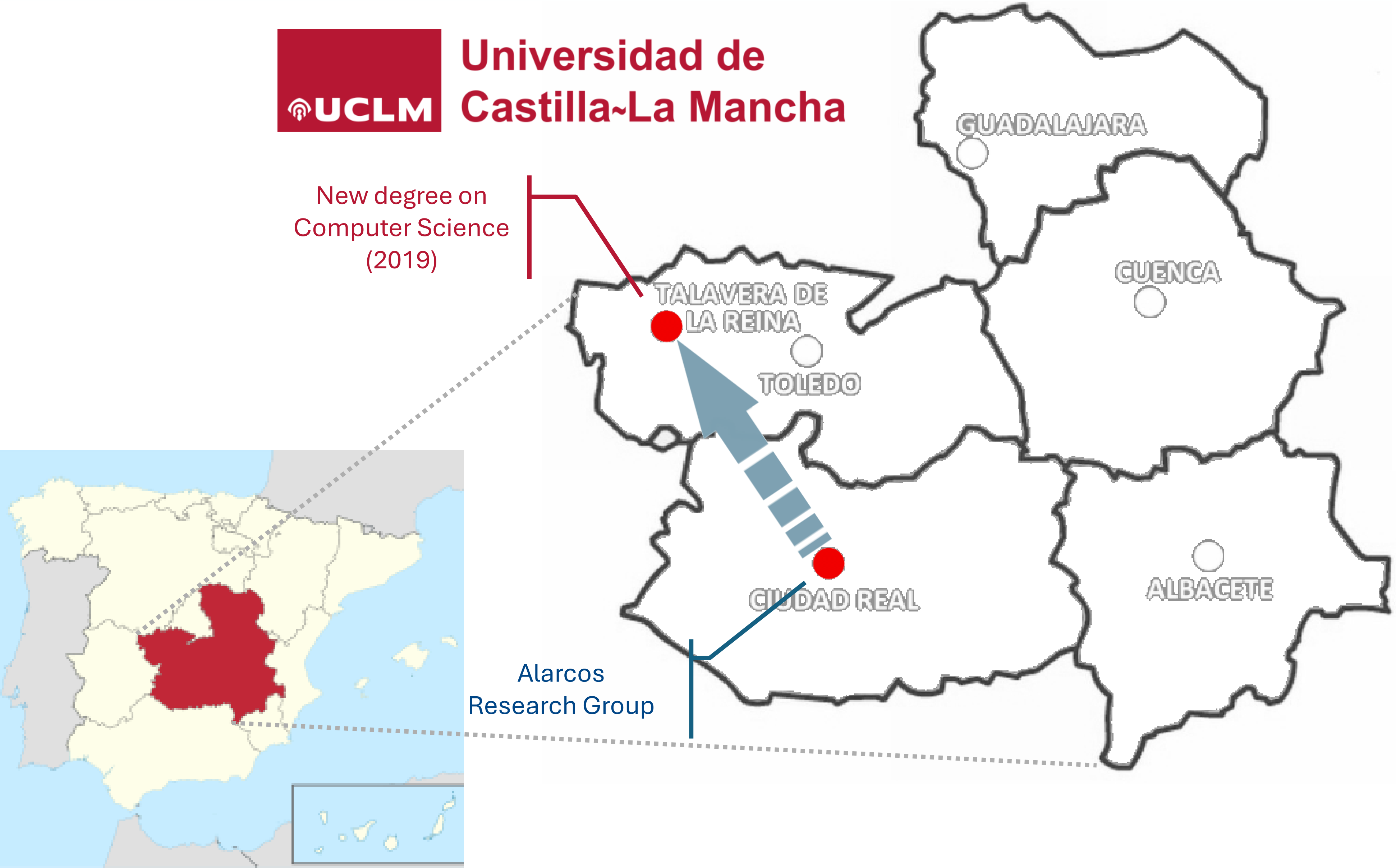
Ricardo Pérez del Castillo

UCLM Universidad de Castilla~La Mancha

10-14 November 2025, Toledo, Spain

Background

Universidad de
Castilla~La Mancha



Facultad de
Ciencias Sociales y
Tecnologías de la Información
Talavera de la Reina. UCLM

Agenda

- Quantum Computing
- Quantum Computing Applications
- Quantum Software & Quantum Software Engineering
- Our Research Journey
- Conclusions

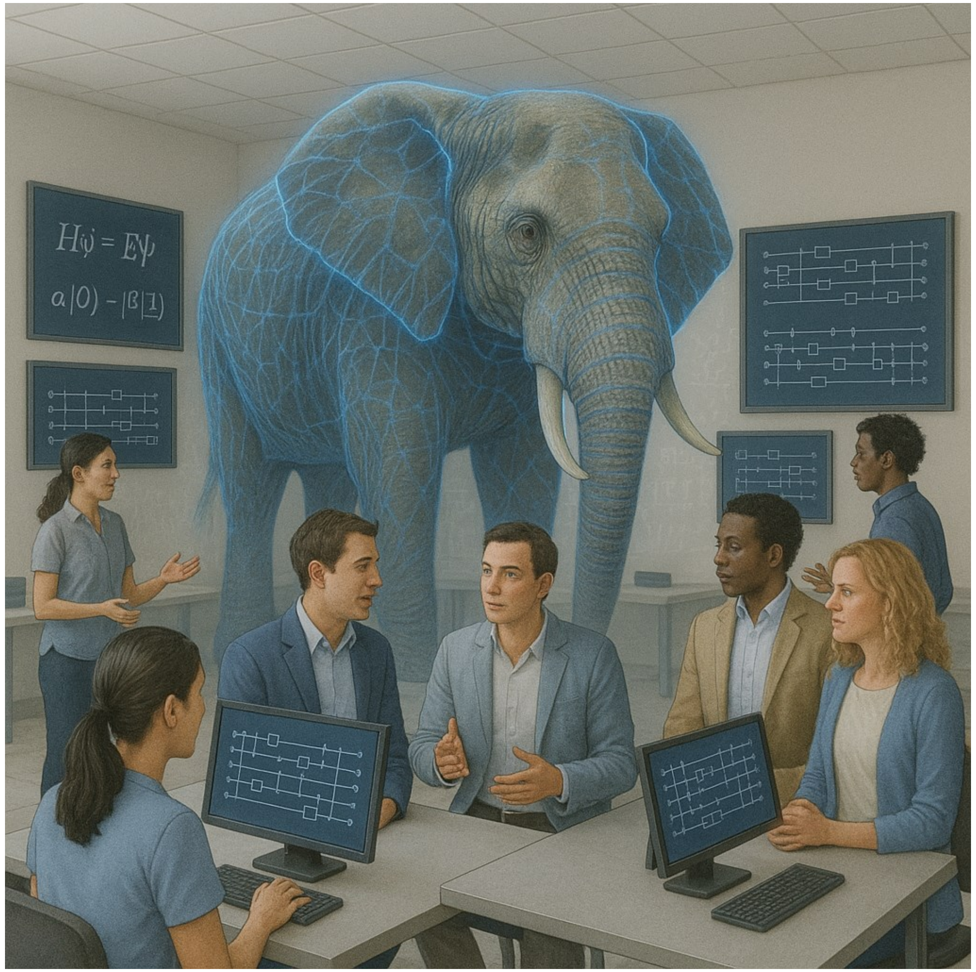
Scan for slides

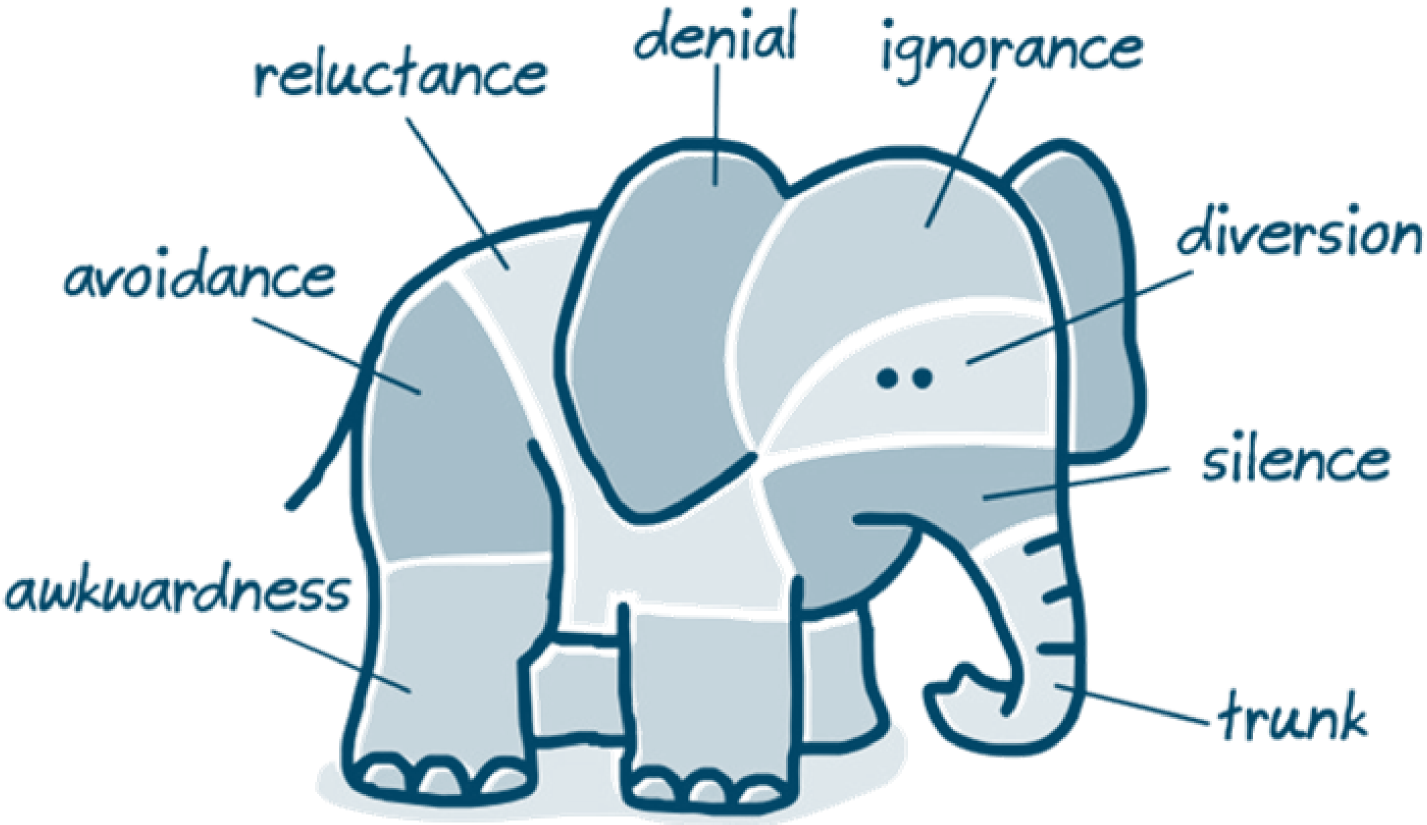
Agenda

- **Quantum Computing**
- Quantum Computing Applications
- Quantum Software & Quantum Software Engineering
- Our Research Journey
- Conclusions

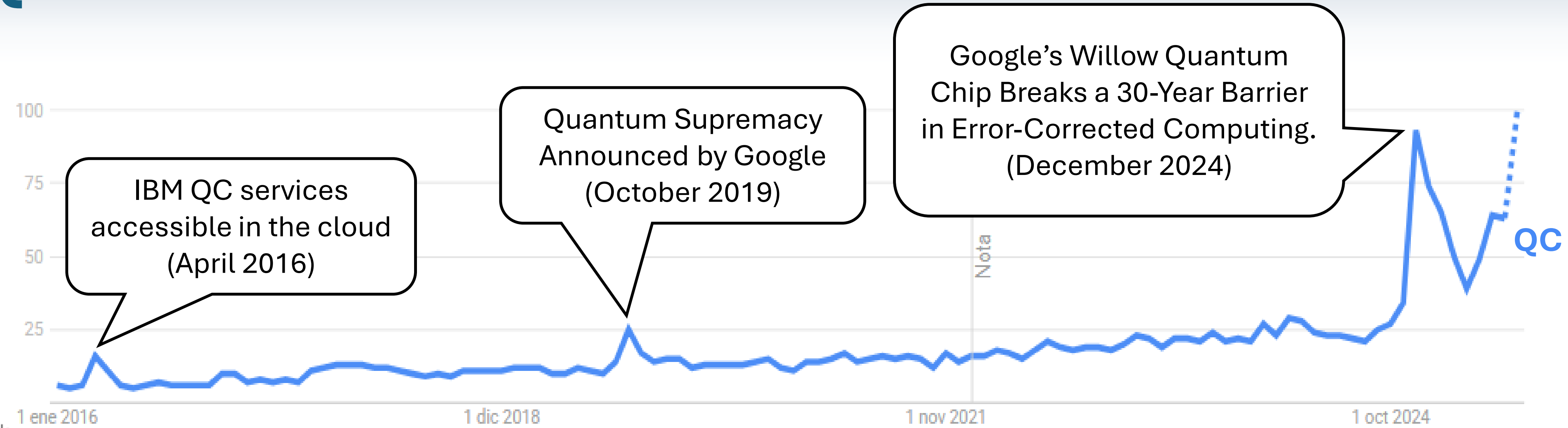
Scan for slides

Quantum Computing is in the Room



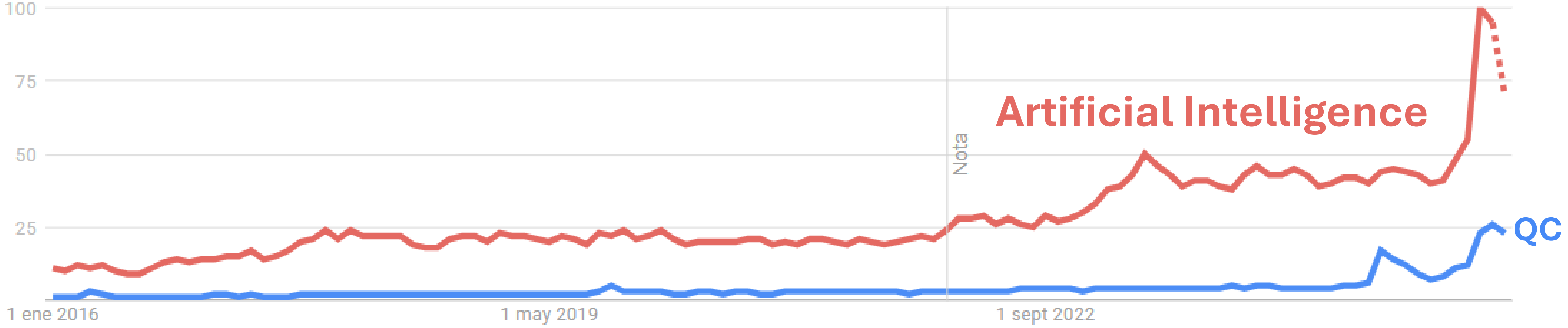


QC interest


IBM QC services accessible in the cloud (April 2016)
Quantum Supremacy Announced by Google (October 2019)
Google's Willow Quantum Chip Breaks a 30-Year Barrier in Error-Corrected Computing. (December 2024)
100
75
50
25
Nota
QC
1 ene 2016
1 dic 2018
1 nov 2021
1 oct 2024


Google Trends
GPT-4 Release (March 2024)
GenAI ?
100
75
50
25
Nota
1 ene 2016
1 abr 2019
1 jul 2022

QC interest



Google Trends

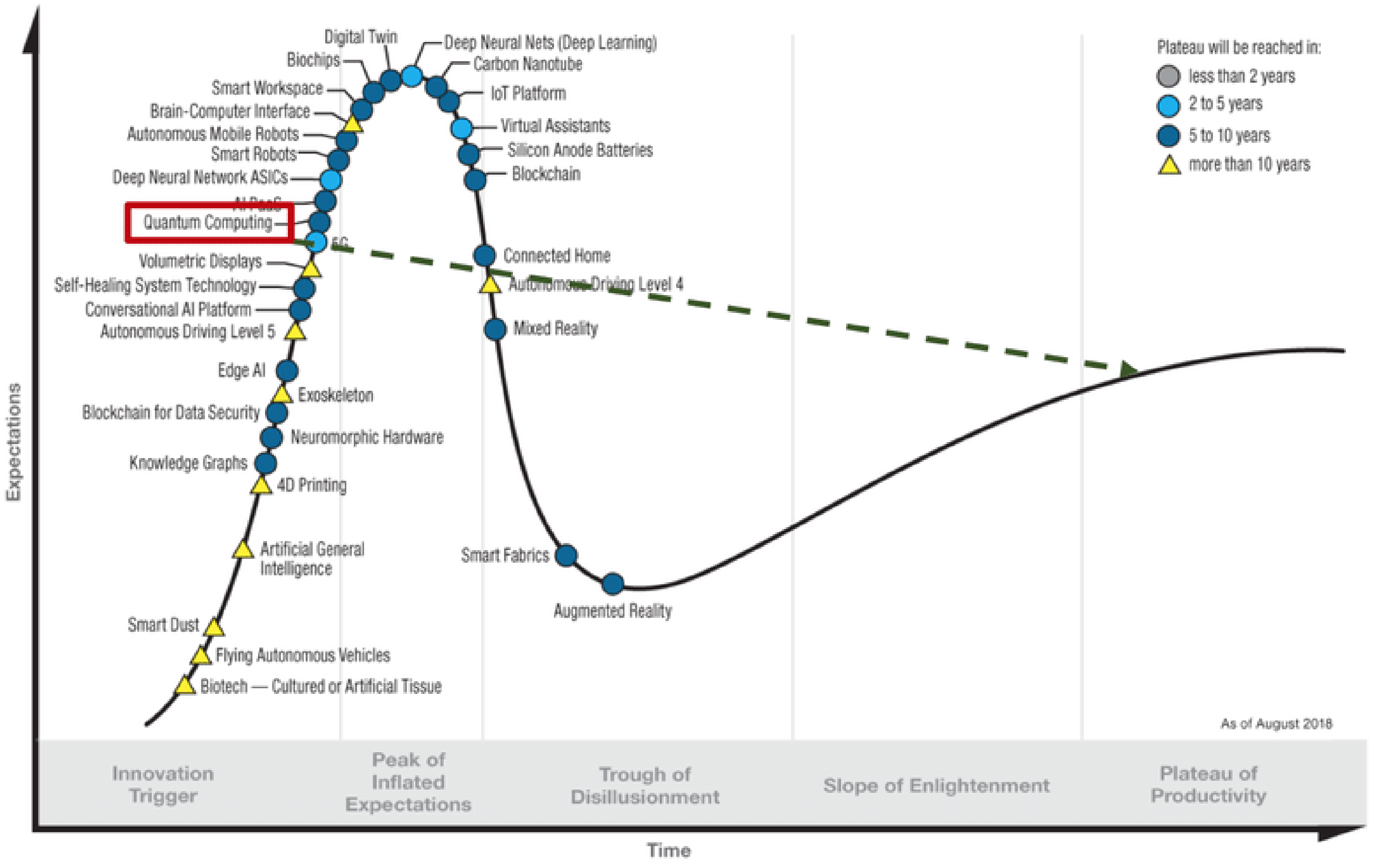
Some thoughts

> *Quantum information is a radical departure in information technology, more fundamentally different from current technology than the digital computer is from the abacus*

William Daniel Phillips, 1997

Today (and Tomorrow) Quantum Computing Expectations

Hype Cycle for Emerging Technologies 2018



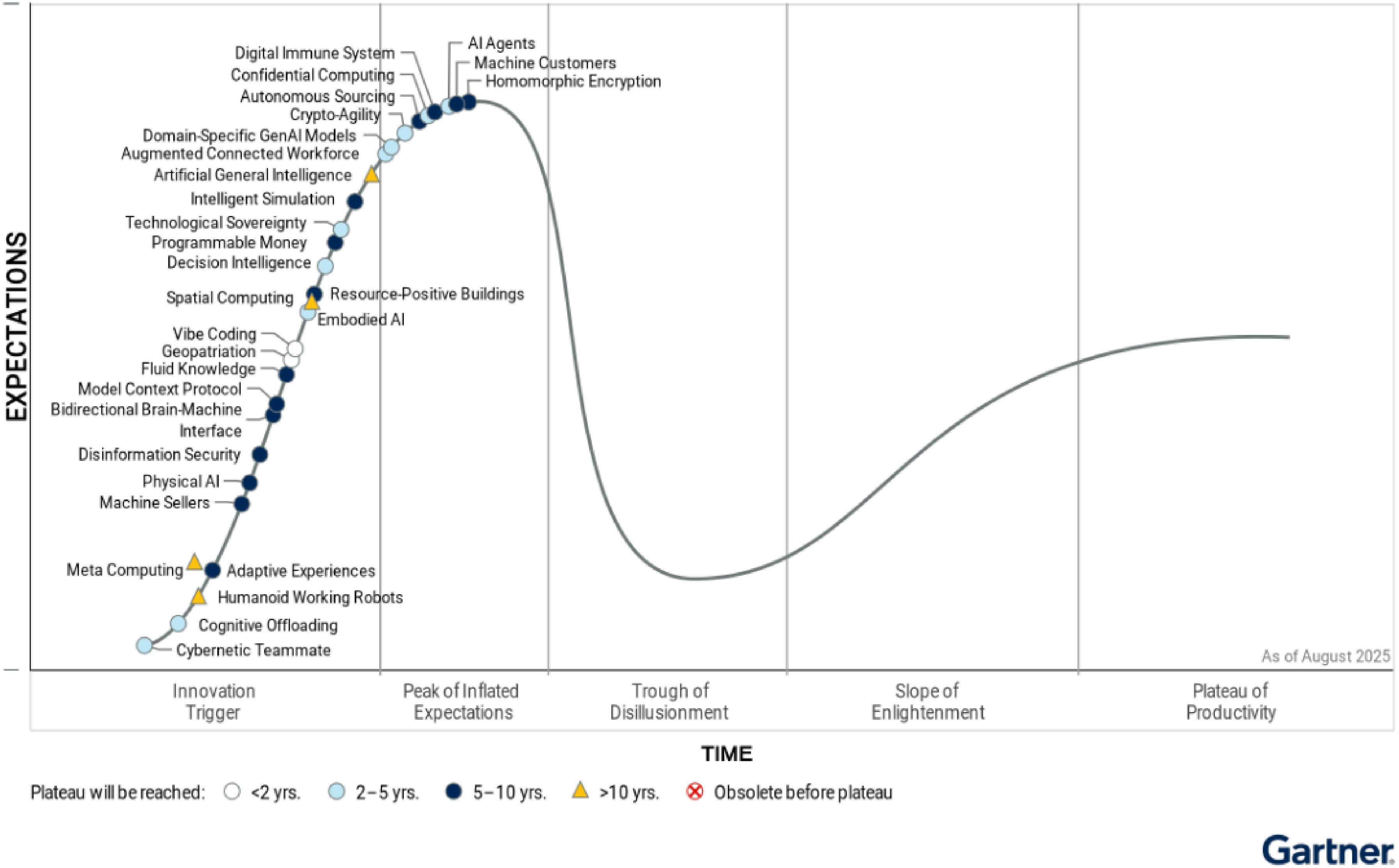
gartner.com/SmarterWithGartner

Source: Gartner (August 2018)

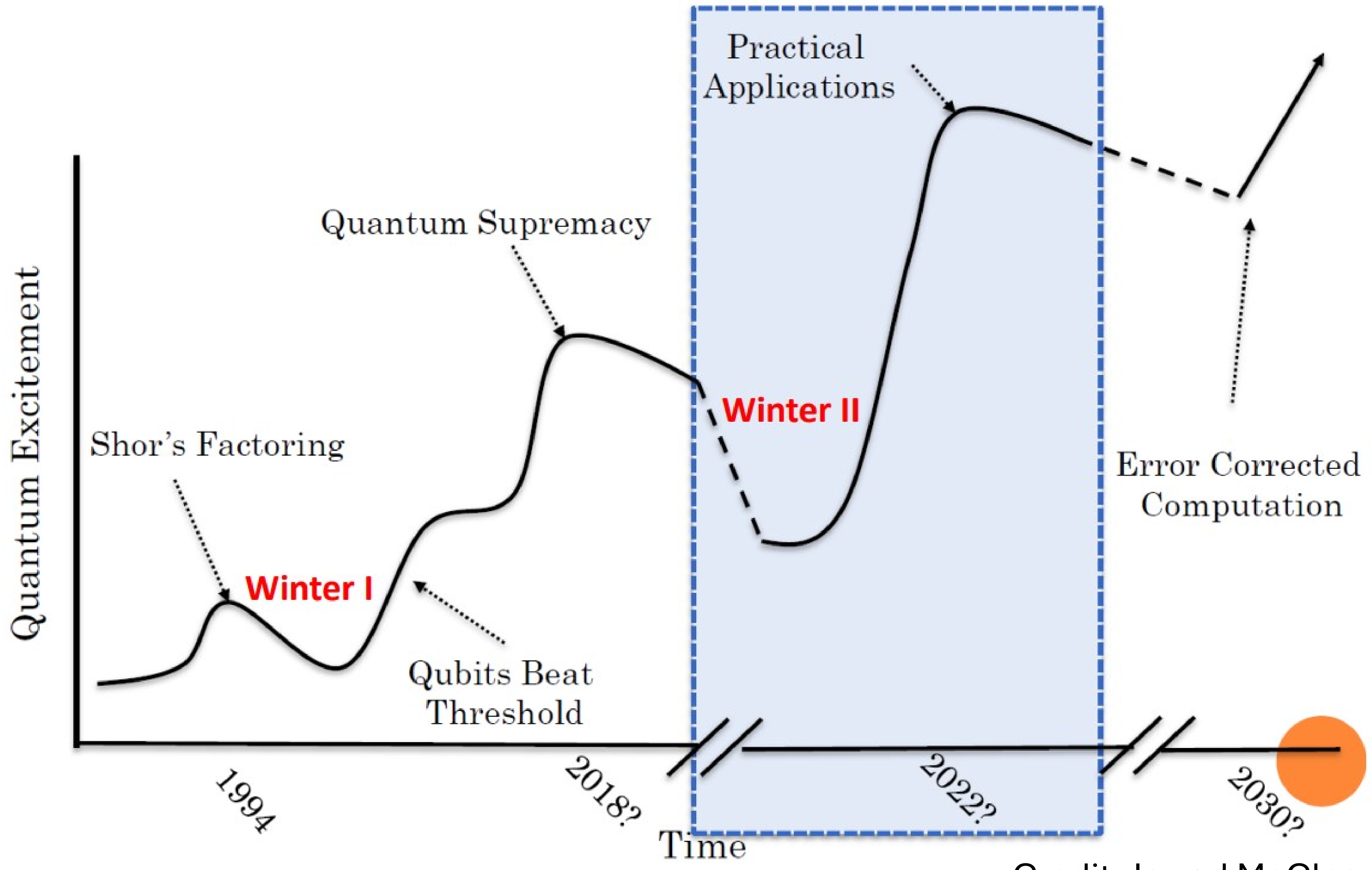
Gartner

Today (and Tomorrow) Quantum Computing Expectations

Hype Cycle for Emerging Technologies 2025

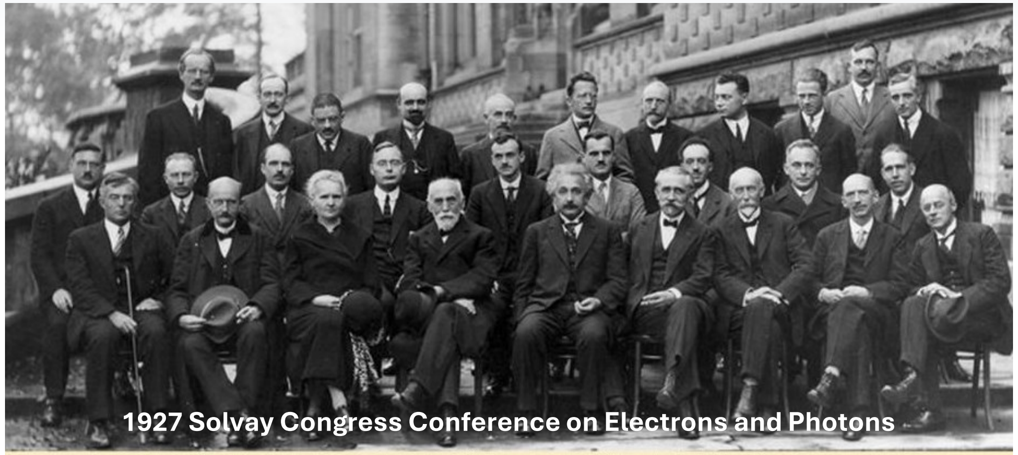


Quantum Computing Hype Cycles



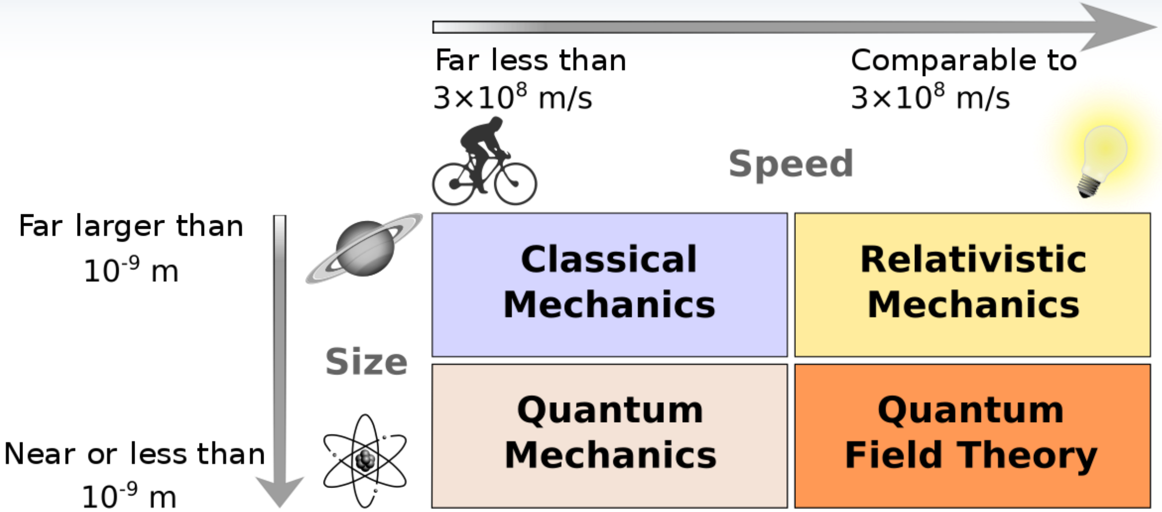
Credit: Jarrod McClean

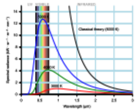
Quantum Mechanics Foundations



1927 Solvay Congress Conference on Electrons and Photons

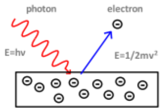
A. Piccard, E. Henriot, P. Ehrenfest, E. Herzen, Th. De Donder, E. Schrödinger, J.E. Verschaffelt, W. Pauli, W. Heisenberg, R.H. Fowler, L. Brillouin;
P. Debye, M. Knudsen, W.L. Bragg, H.A. Kramers, P.A.M. Dirac, A.H. Compton, L. de Broglie, M. Born, N. Bohr;
Langmuir, M. Planck, M. Curie, H.A. Lorentz, A. Einstein, P. Langevin, Ch. E. Guye, C.T.R. Wilson, O.W. Richardson

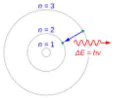




1900 1918
Max Planck
black body radiation
energy quanta
Planck constant

1905 1921
Albert Einstein
photoelectric effect





1913 1922
Niels Bohr
hydrogen atom model

1922 1944
Stern-Gerlach experiment
atoms angular momentum

1924 1929
Louis de Broglie
wave-particule duality

1924 1945
Wolfgang Pauli
exclusion principle

1925
Uhlenbeck-Goudsmit
electron spin

1926 1933
Erwin Schrödinger
wave function

$$i\hbar \frac{\partial \Psi(x,t)}{\partial t} = H\Psi(x,t)$$

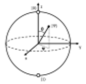
1926 1954
Max Born
quantum probablities

1927 1932
Werner Heisenberg
indetermination

1935
Einstein, Podolski, Rosen
EPR paradox

1935
Erwin Schrödinger
cat

1937
Etore Majorana
fermion



1946 1952
Felix Bloch
sphere

1947 1956
William Shockley
John Bardeen
Walter Brattain
transistors

1953-1960
Gordon Gould
Theodore Maiman
Nikolay Basov 1964
Alexander Prokhorov 1964
Charles Hard Townes 1964
Arthur Leonard Schawlow 1981
maser and laser

1957
John Bardeen
Leon Cooper
John Robert Schrieffer
superconductivity

1962 1973
Brian Josephson
Josephson effect

1964
John Stewart Bell
Bell inequalities

1970
Dieter Zeh
quantum decoherence

1980
Yuri Manin
quantum computing

1980
Tommaso Toffoli
Toffoli gate

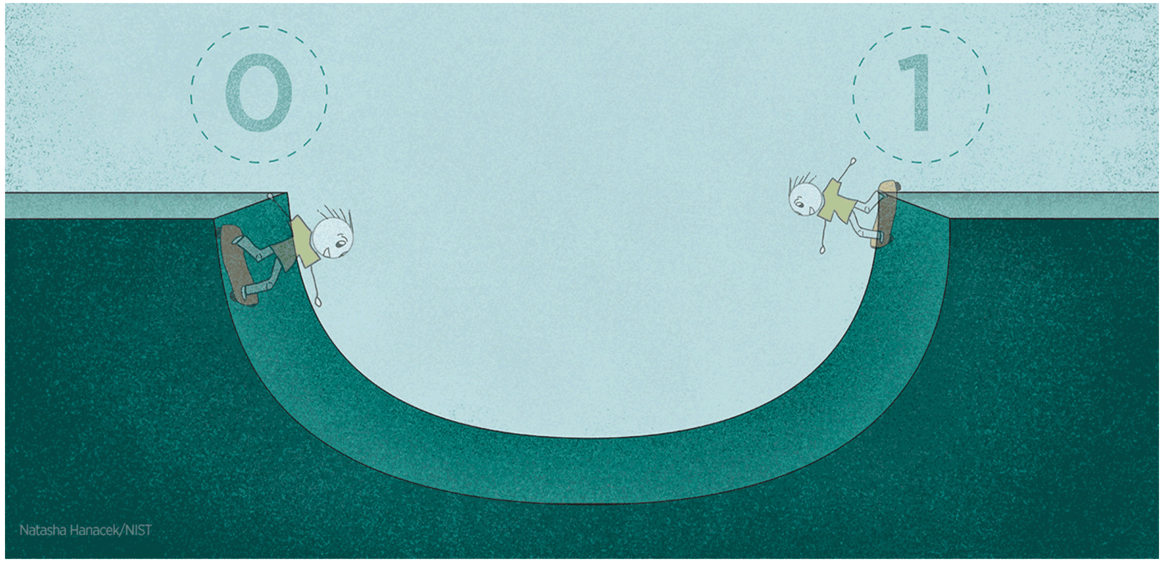
1981
Richard Feynman
quantum simulator

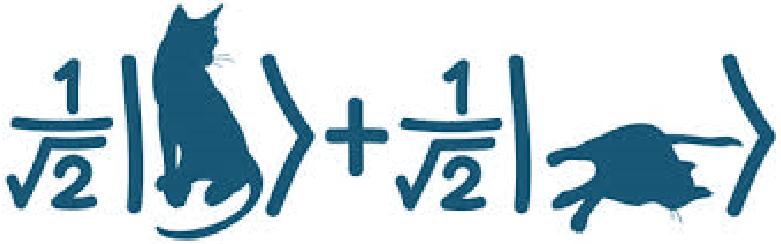
1982
William Wooters
Wojciech Zurek
non cloning theorem

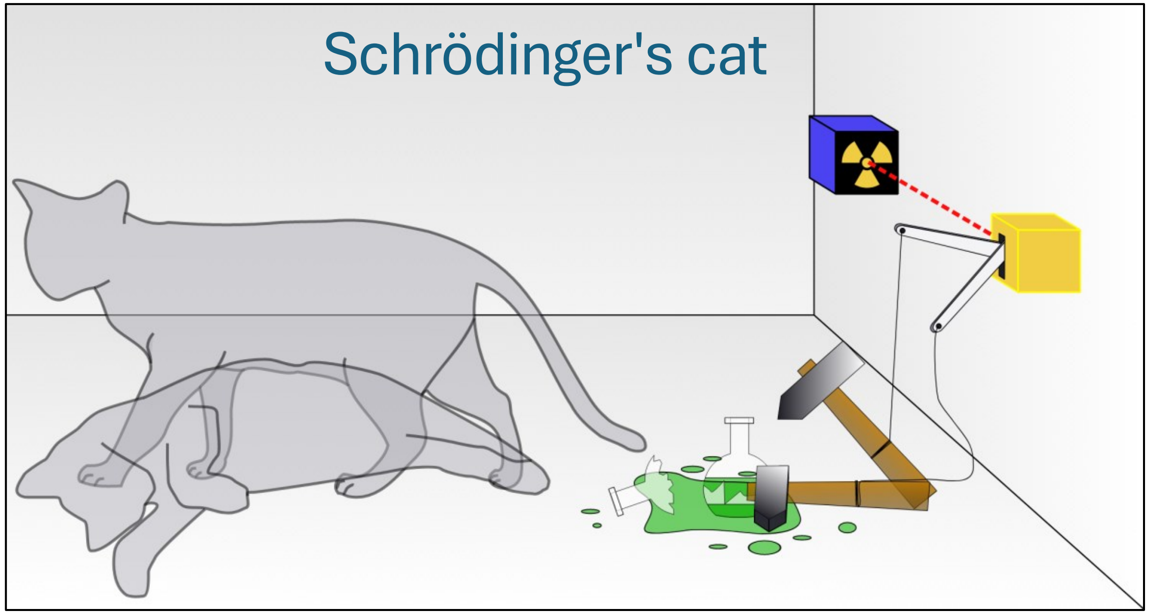
1982
Alain Aspect
Jean Dalibard
Gérard Roger
Philippe Grangier
entanglement

1900 1905 1910 1915 1920 1925 1930 1935 1940 1945 1955 1960 1965 1970 1975 1980 1985 1990

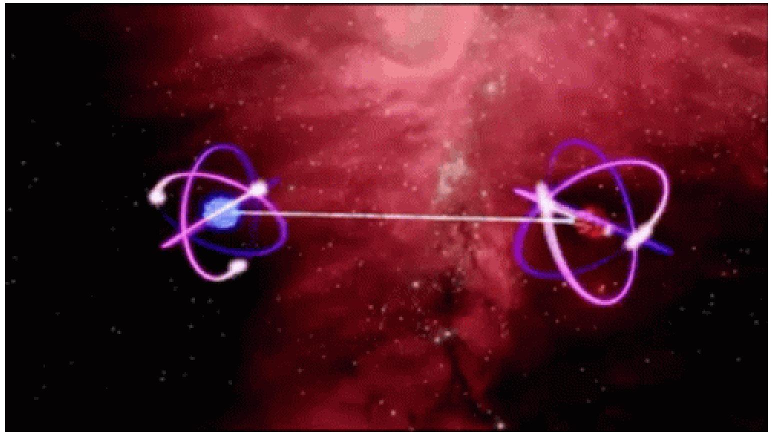
Foundations: Superposition

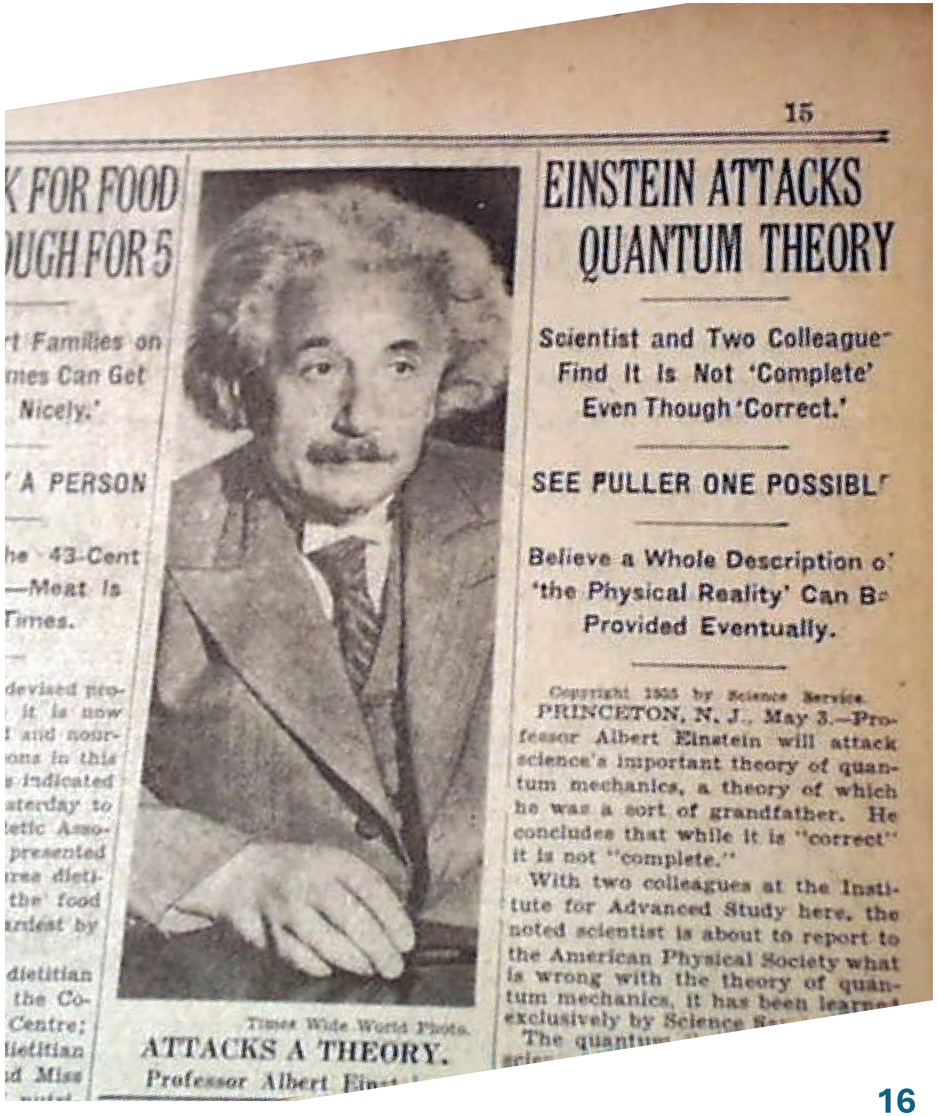






Foundations: Entanglement




15

K FOR FOOD
UGH FOR 5

t Families on
mes Can Get
Nicely.'

A PERSON

he 43-Cent
—Meat Is
Times.

EINSTEIN ATTACKS QUANTUM THEORY

Scientist and Two Colleague
Find It Is Not 'Complete'
Even Though 'Correct.'

SEE FULLER ONE POSSIBL

Believe a Whole Description o
'the Physical Reality' Can B
Provided Eventually.

Copyright 1935 by Science Service.

PRINCETON, N. J., May 3.—Professor Albert Einstein will attack science's important theory of quantum mechanics, a theory of which he was a sort of grandfather. He concludes that while it is "correct" it is not "complete."

With two colleagues at the Institute for Advanced Study here, the noted scientist is about to report to the American Physical Society what is wrong with the theory of quantum mechanics, it has been learned exclusively by Science S

The quantum

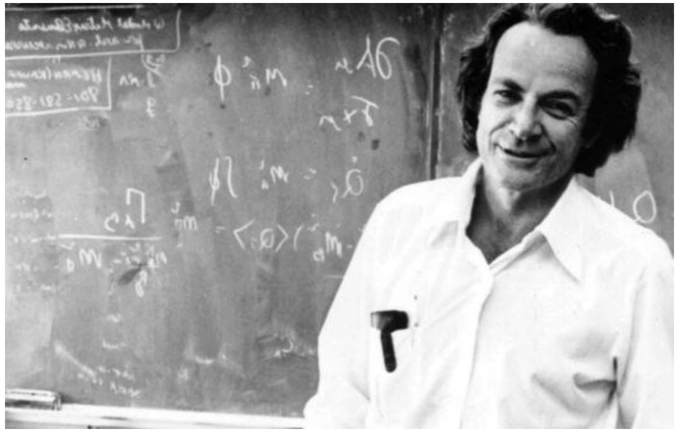
Times Wide World Photo.

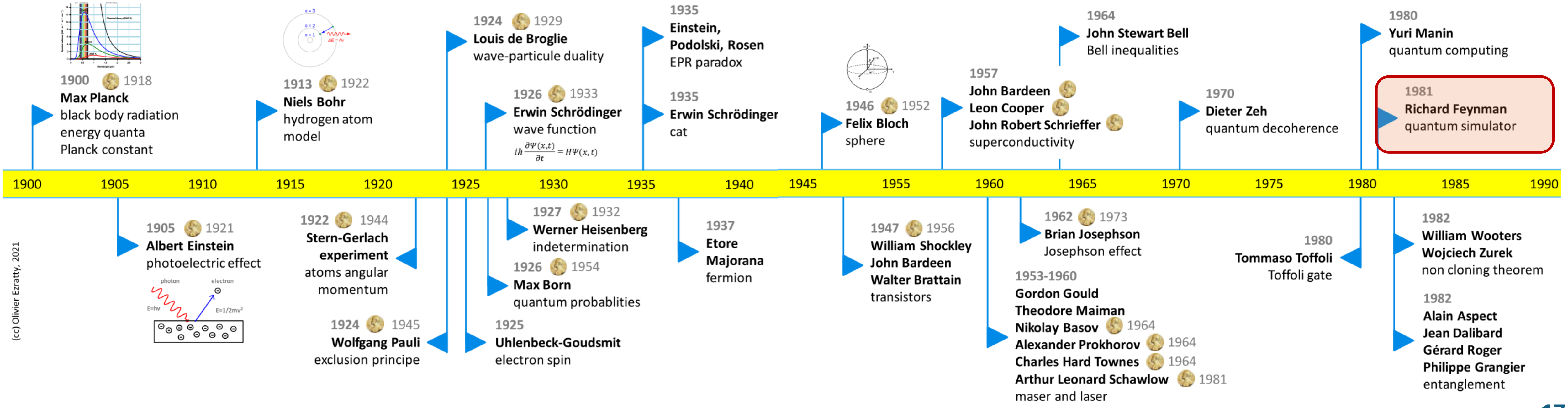
ATTACKS A THEORY.

Professor Albert Ein

Quantum Computing Foundations

> *What kind of computer are we going to use to simulate physics?*
>
> Richard Feyman, 1982





Quantum Computing Foundations

VOLUME 74, NUMBER 20 PHYSICAL REVIEW LETTERS 15 MAY 1995

Quantum Computations with Cold Trapped Ions

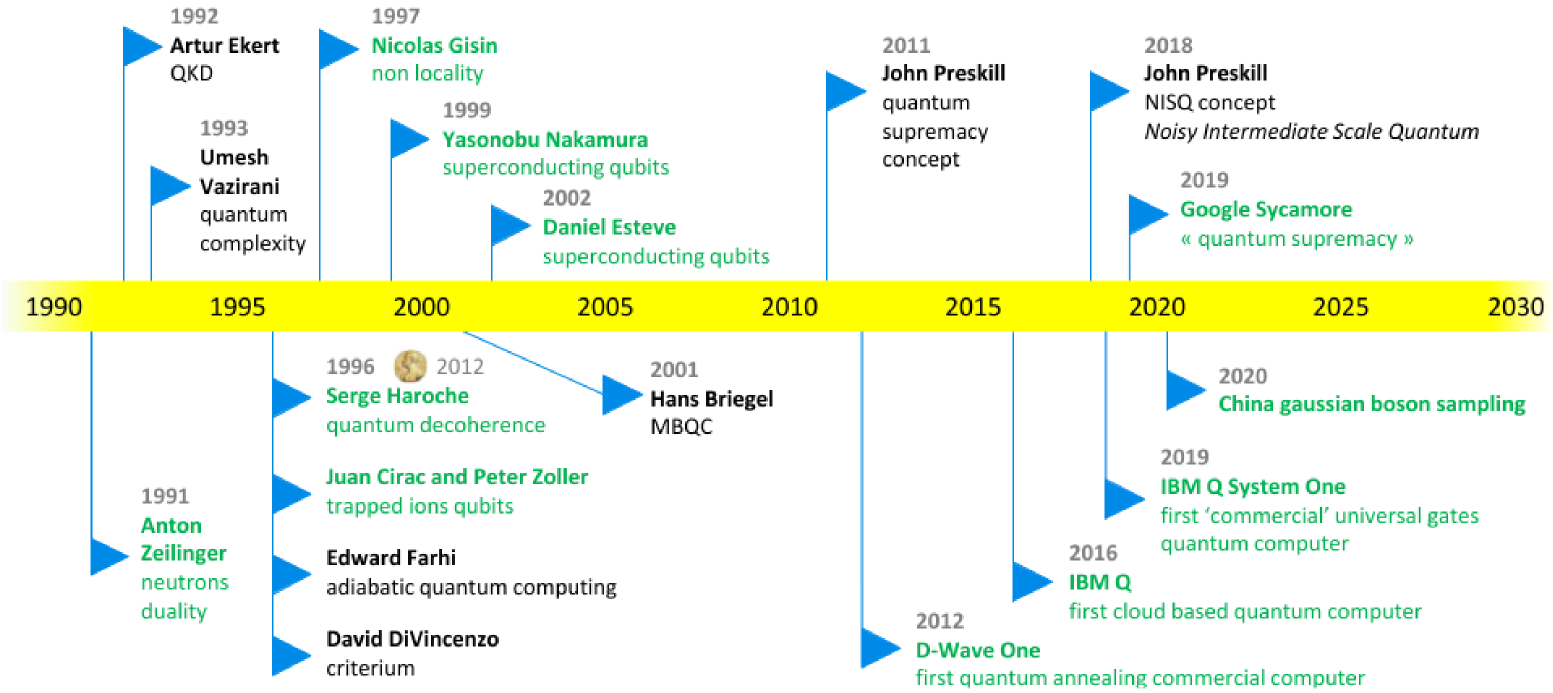
J. I. Cirac and P. Zoller*

Institut für Theoretische Physik, Universiät Innsbruck, Technikerstrasse 25, A-6020 Innsbruck, Austria

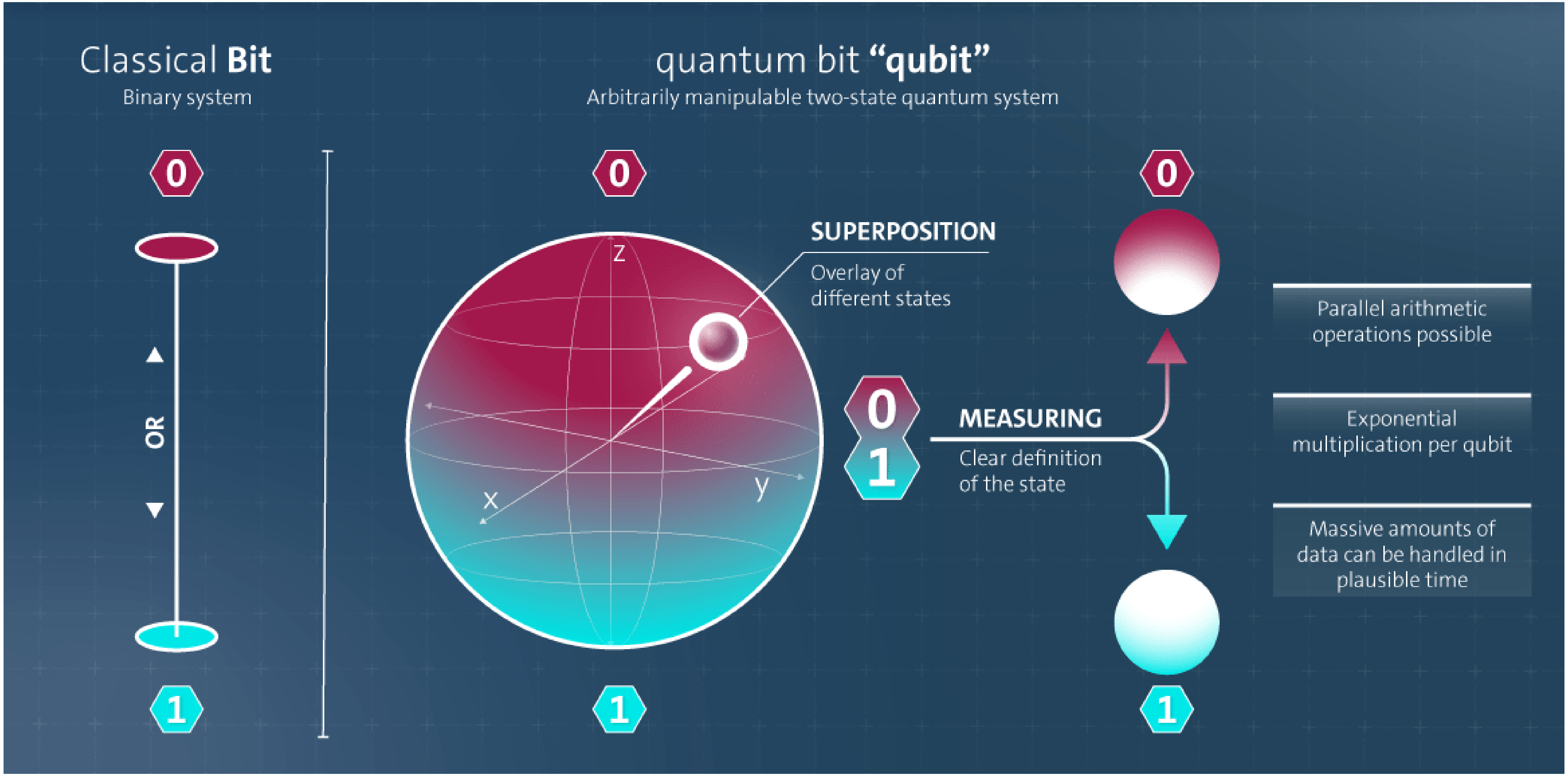
(Received 30 November 1994)

A quantum computer can be implemented with cold ions confined in a linear trap and interacting with laser beams. Quantum gates involving any pair, triplet, or subset of ions can be realized by coupling the ions through the collective quantized motion. In this system decoherence is negligible, and the measurement (readout of the quantum register) can be carried out with a high efficiency.

Juan Ignacio Cirac

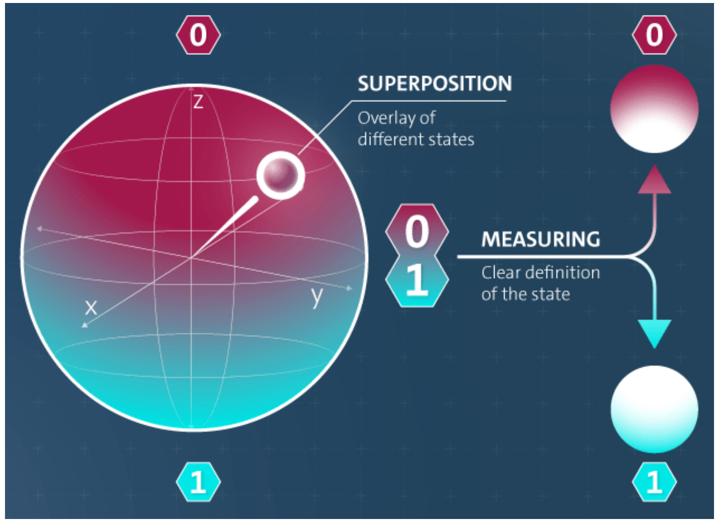


Qubits



Classical **Bit**
Binary system
0
OR
1
quantum bit **“qubit”**
Arbitrarily manipulable two-state quantum system
0
z
x
y
1
SUPERPOSITION
Overlay of different states
0
1
MEASURING
Clear definition of the state
0
1
Parallel arithmetic operations possible
Exponential multiplication per qubit
Massive amounts of data can be handled in plausible time

Quantum Speedups

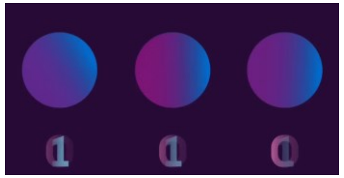


n bits register	n qubits register
2^n possible states once at a time	2^n possible states simultaneously
evaluable	partially evaluable
independent copies	no copy
individualy erasable	non individualy erasable
non destructive readout	value changed after readout
deterministic	probabilistic

101

n=3 example

000
001
010
011
100
101
110
111



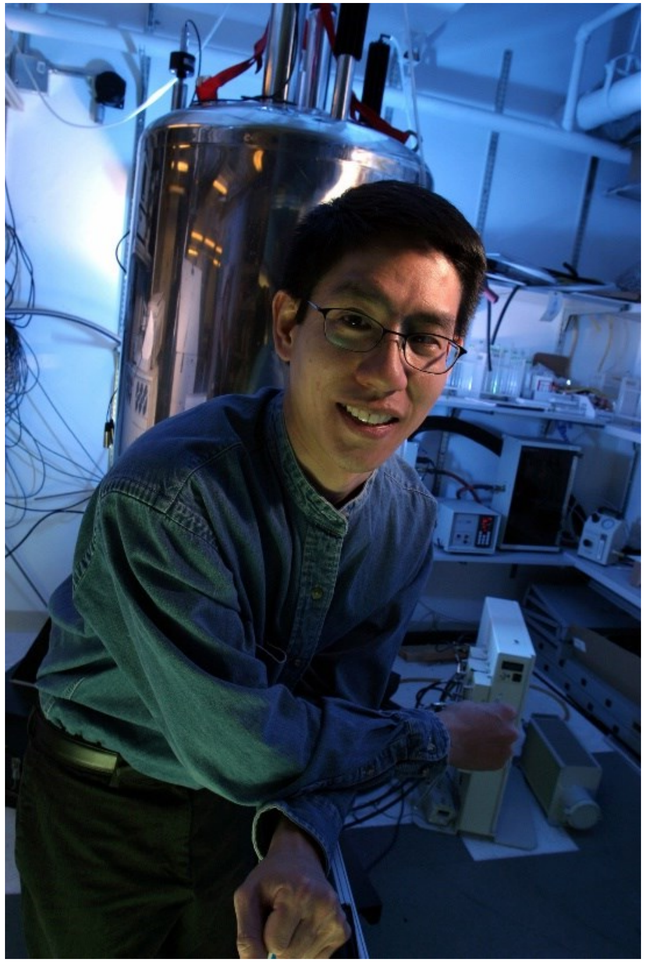
aka register pure states

QC is not a tomorrow's technology, it is real

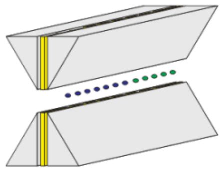
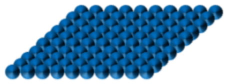
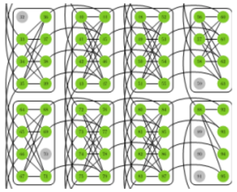
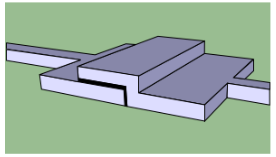
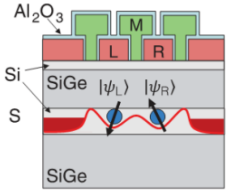
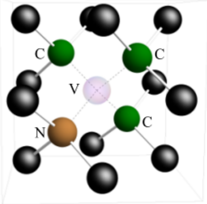
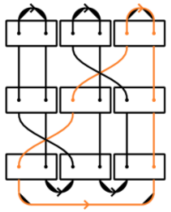
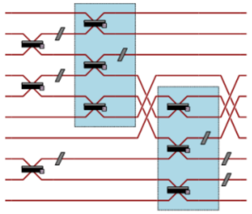
"

The thing driving the hype is the realization that quantum computing is actually real.
*It is no longer a physicist's dream — **it is an engineer's nightmare**.*

Isaac Chuang, 2018



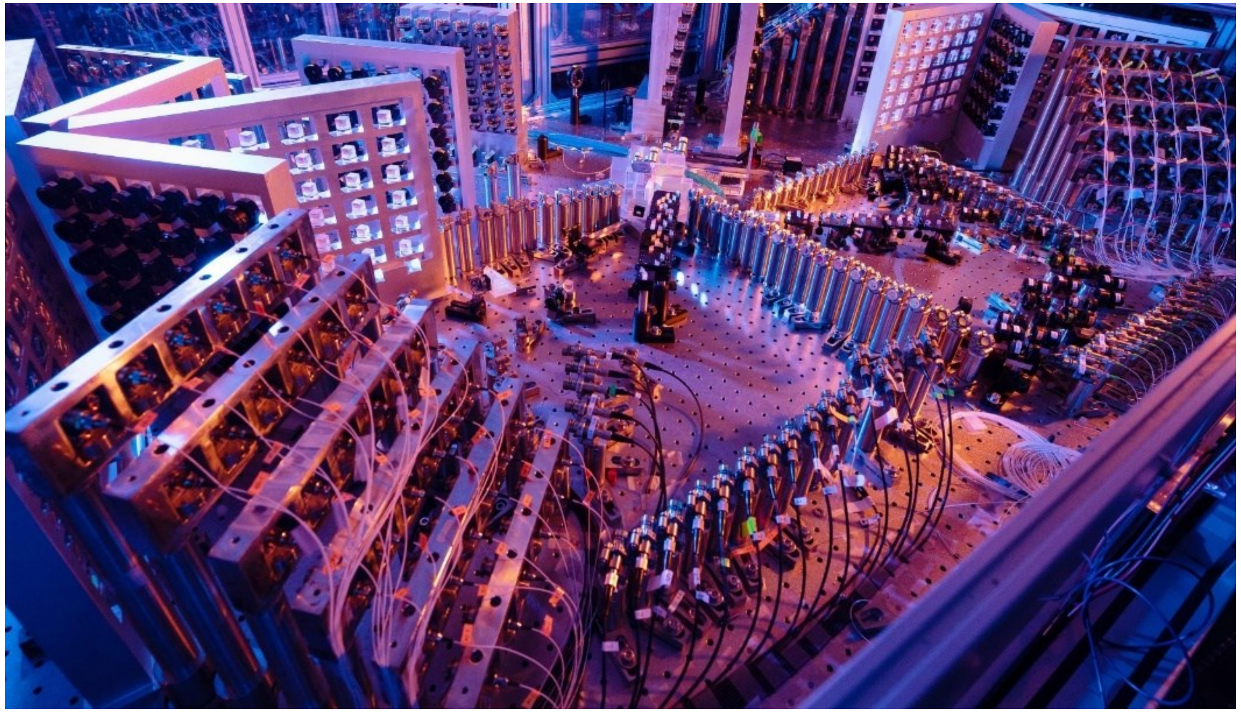
Physical Realization of Qubits

atoms		electrons *controlled spin and microwave cavities*					photons
							
trapped ions	**cold atoms**	**annealing**	**superconducting**	**silicon**	**vacancies**	**topological**	**photons**
1: Quantinuum, AQT, NEQXT, ZuriQ, Quantum Art, Foxconn, HYQ 华翊量子	4: QuEra, Pasqal, Infleqtion, GQI, QUANTier, planqc, logiqal	D-Wave The Quantum Computing Company, Qilimanjaro Quantum Tech, NEC	7: IBM, rigetti, IQM, Google, 国盾, 本源量子 Origin Quantum, Anyon, OQC, Fujitsu, QOLAB, QuamCore, Peak Quantum	9: diraq, quobly, Qπ AI, Quantum Motion	12: Quantum Brilliance, Turing, XEED, SaxonQ, Quantum Transistors, Fujitsu	Microsoft, Quoherent, Quantum Gravity Research, Nokia	14: PsiQuantum, Quandela, QuiX Quantum, ORCA Computing, Duality Quantum Photonics, aegiq, Tundra Systems Global
2: IonQ, eleQtron, 国仪量子, Qudora Technologies	5: atom computing		8: Alice & Bob, Nord Quantique, amazon, qci, Atlantic Quantum, Z-Axis Quantum	10: Groove Quantum, equal1.labs, ARQUE, intel	13: photonic		15: Xanadu, Quanfluence
3: Crystal Quantum Computing	6: Atom Quantum Labs			11: Silicon Quantum Computing, EeroQ, C12, Archer			16: QCi, QBoson

Olivier Ezratty, September 2025.

Quantum Computing Hardware

Superconducting



Photonics

Physical Realization of Qubits

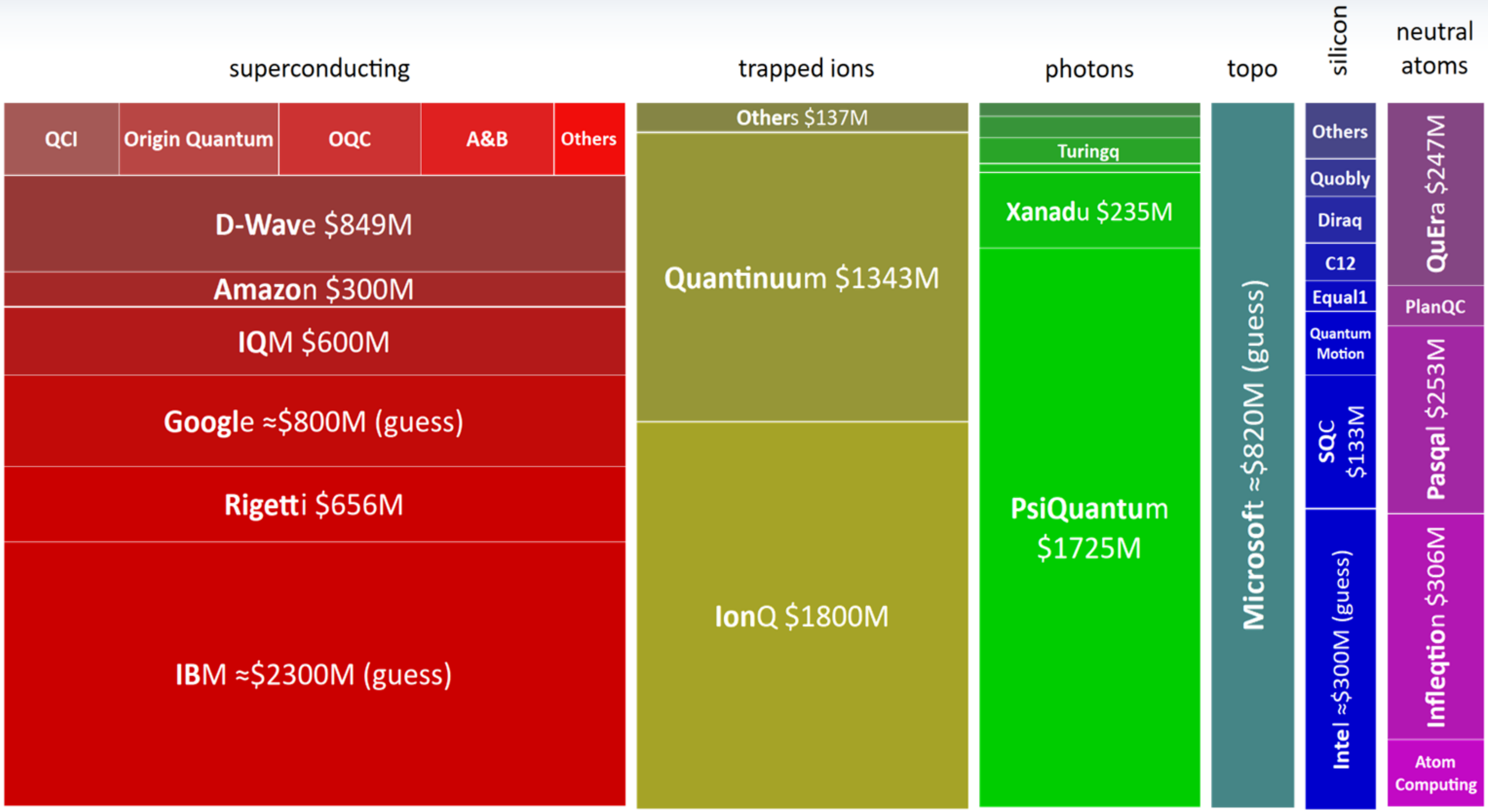
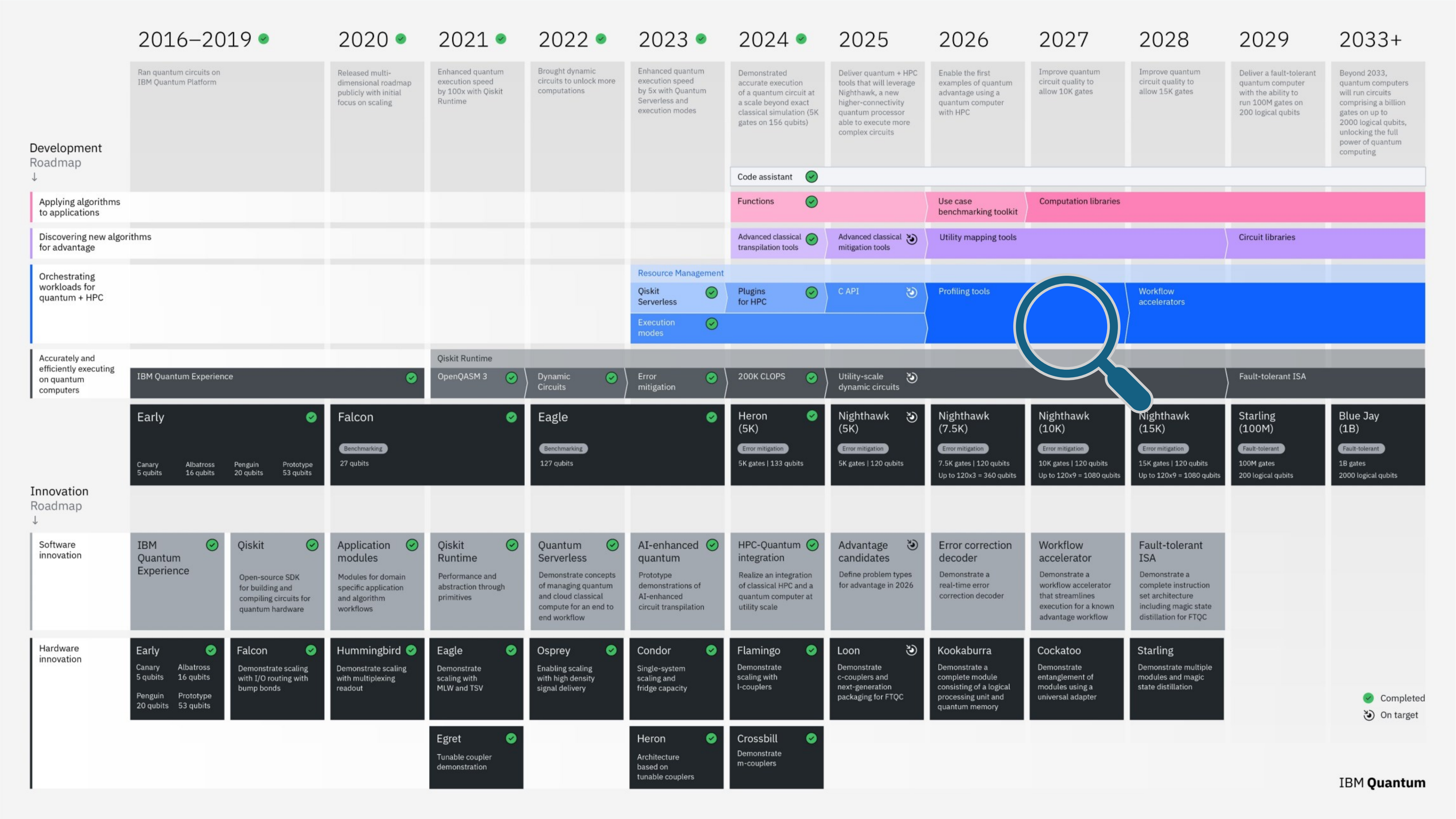


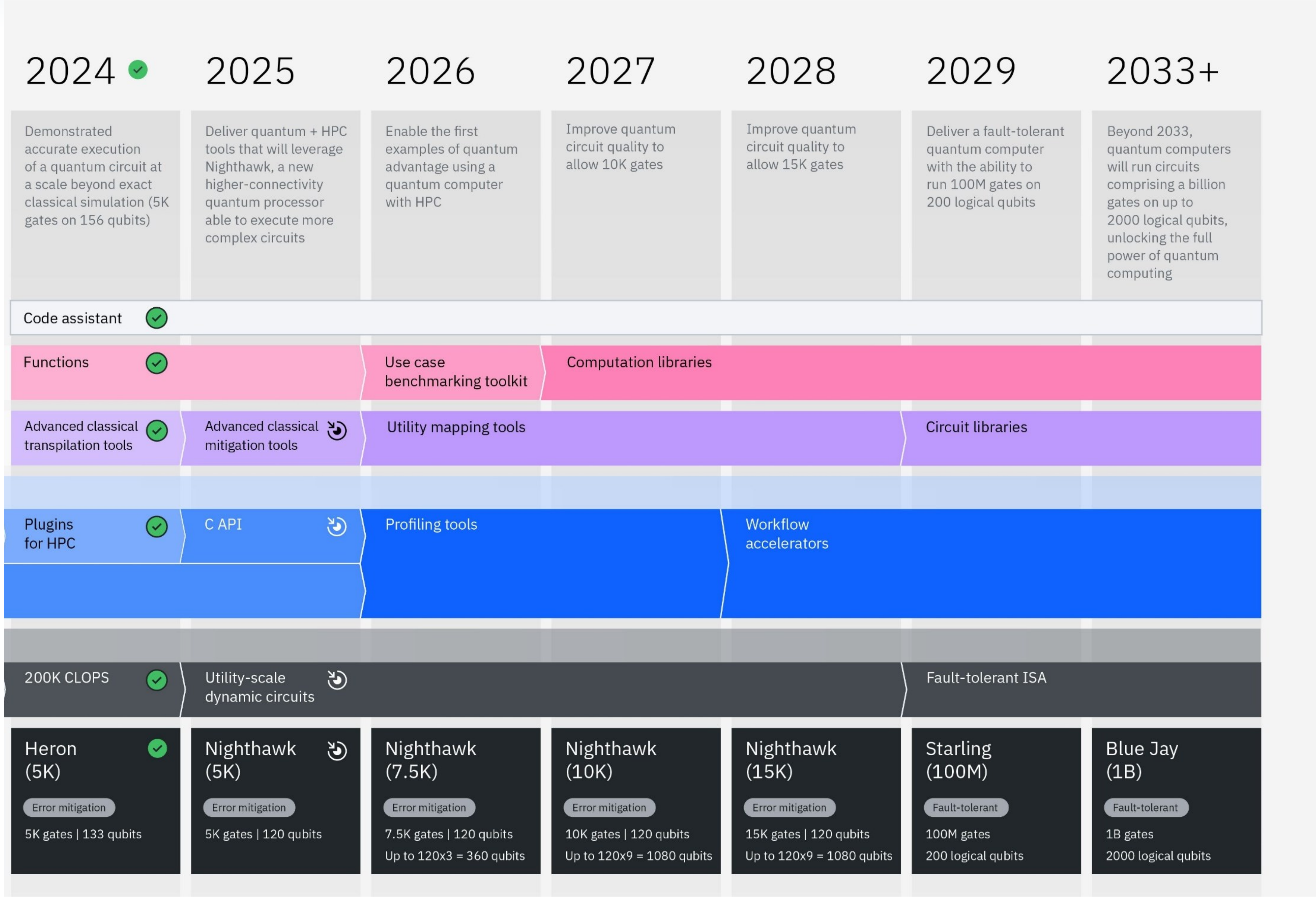
Figure 300: Fuzzy logic assessment of industry investments per qubit type mixing capital investment for startups and internal investments for legacy companies is "life to date" (LTD). (cc) Olivier Ezratty, September 2025.

IBM Roadmap

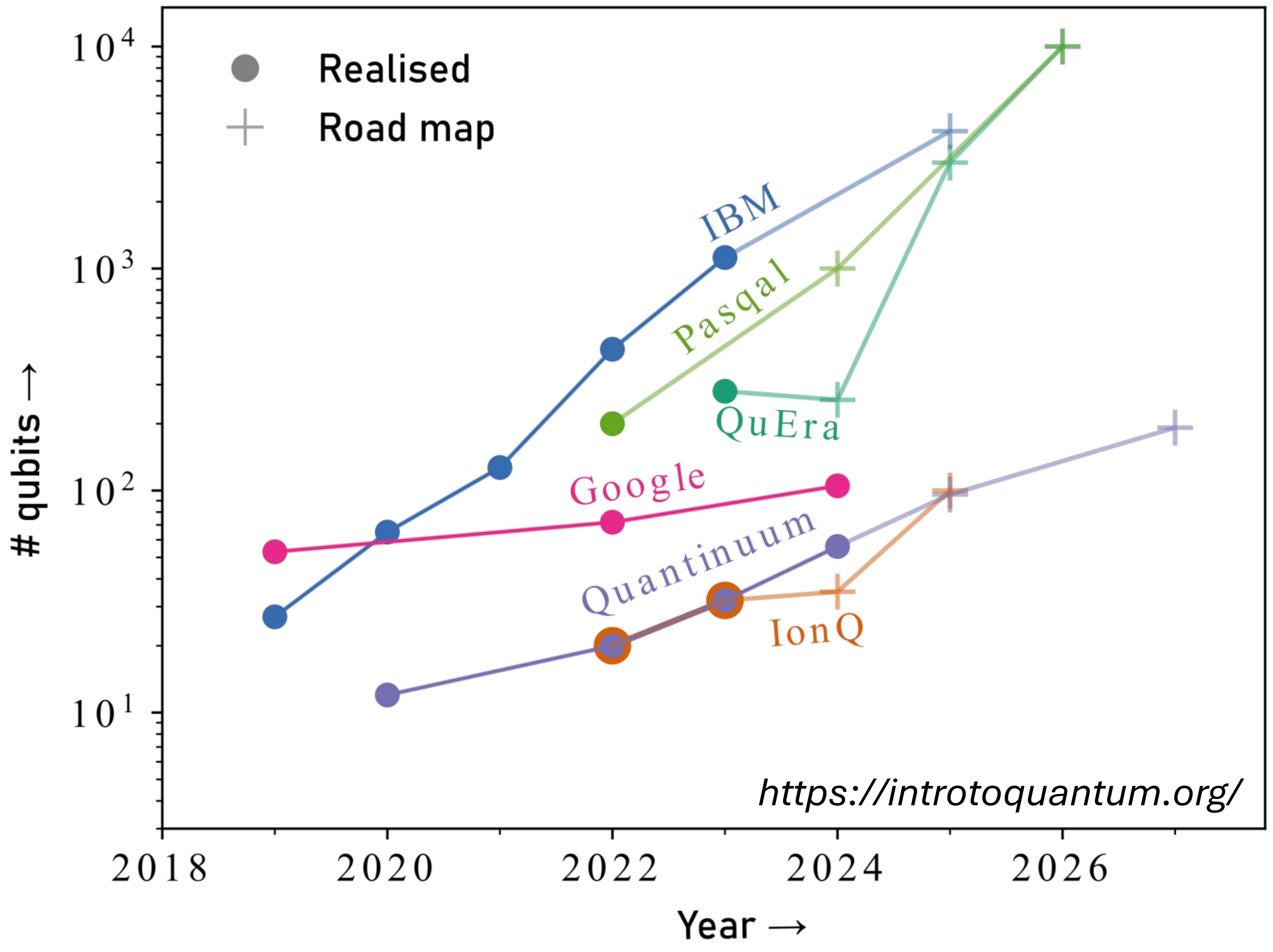


2016–2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2033+
Development Roadmap
Ran quantum circuits on IBM Quantum Platform
Released multi-dimensional roadmap publicly with initial focus on scaling
Enhanced quantum execution speed by 100x with Qiskit Runtime
Brought dynamic circuits to unlock more computations
Enhanced quantum execution speed by 5x with Quantum Serverless and execution modes
Demonstrated accurate execution of a quantum circuit at a scale beyond exact classical simulation (5K gates on 156 qubits)
Deliver quantum + HPC tools that will leverage Nighthawk, a new higher-connectivity quantum processor able to execute more complex circuits
Enable the first examples of quantum advantage using a quantum computer with HPC
Improve quantum circuit quality to allow 10K gates
Improve quantum circuit quality to allow 15K gates
Deliver a fault-tolerant quantum computer with the ability to run 100M gates on 200 logical qubits
Beyond 2033, quantum computers will run circuits comprising a billion gates on up to 2000 logical qubits, unlocking the full power of quantum computing
Code assistant
Applying algorithms to applications
Functions
Use case benchmarking toolkit
Computation libraries
Discovering new algorithms for advantage
Advanced classical transpilation tools
Advanced classical mitigation tools
Utility mapping tools
Circuit libraries
Orchestrating workloads for quantum + HPC
Resource Management
Qiskit Serverless
Plugins for HPC
C API
Profiling tools
Workflow accelerators
Execution modes
Accurately and efficiently executing on quantum computers
Qiskit Runtime
IBM Quantum Experience
OpenQASM 3
Dynamic Circuits
Error mitigation
200K CLOPS
Utility-scale dynamic circuits
Fault-tolerant ISA
Early
Canary 5 qubits
Albatross 16 qubits
Penguin 20 qubits
Prototype 53 qubits
Falcon
Benchmarking
27 qubits
Eagle
Benchmarking
127 qubits
Heron (5K)
Error mitigation
5K gates | 133 qubits
Nighthawk (5K)
Error mitigation
5K gates | 120 qubits
Nighthawk (7.5K)
Error mitigation
7.5K gates | 120 qubits
Up to 120x3 = 360 qubits
Nighthawk (10K)
Error mitigation
10K gates | 120 qubits
Up to 120x9 = 1080 qubits
Nighthawk (15K)
Error mitigation
15K gates | 120 qubits
Up to 120x9 = 1080 qubits
Starling (100M)
Fault-tolerant
100M gates
200 logical qubits
Blue Jay (1B)
Fault-tolerant
1B gates
2000 logical qubits
Innovation Roadmap
Software innovation
IBM Quantum Experience
Qiskit
Open-source SDK for building and compiling circuits for quantum hardware
Application modules
Modules for domain specific application and algorithm workflows
Qiskit Runtime
Performance and abstraction through primitives
Quantum Serverless
Demonstrate concepts of managing quantum and cloud classical compute for an end to end workflow
AI-enhanced quantum
Prototype demonstrations of AI-enhanced circuit transpilation
HPC-Quantum integration
Realize an integration of classical HPC and a quantum computer at utility scale
Advantage candidates
Define problem types for advantage in 2026
Error correction decoder
Demonstrate a real-time error correction decoder
Workflow accelerator
Demonstrate a workflow accelerator that streamlines execution for a known advantage workflow
Fault-tolerant ISA
Demonstrate a complete instruction set architecture including magic state distillation for FTQC
Hardware innovation
Early
Canary 5 qubits
Albatross 16 qubits
Penguin 20 qubits
Prototype 53 qubits
Falcon
Demonstrate scaling with I/O routing with bump bonds
Hummingbird
Demonstrate scaling with multiplexing readout
Eagle
Demonstrate scaling with MLW and TSV
Osprey
Enabling scaling with high density signal delivery
Condor
Single-system scaling and fridge capacity
Flamingo
Demonstrate scaling with l-couplers
Loon
Demonstrate c-couplers and next-generation packaging for FTQC
Kookaburra
Demonstrate a complete module consisting of a logical processing unit and quantum memory
Cockatoo
Demonstrate entanglement of modules using a universal adapter
Starling
Demonstrate multiple modules and magic state distillation
Egret
Tunable coupler demonstration
Heron
Architecture based on tunable couplers
Crossbill
Demonstrate m-couplers
Completed
On target
IBM Quantum

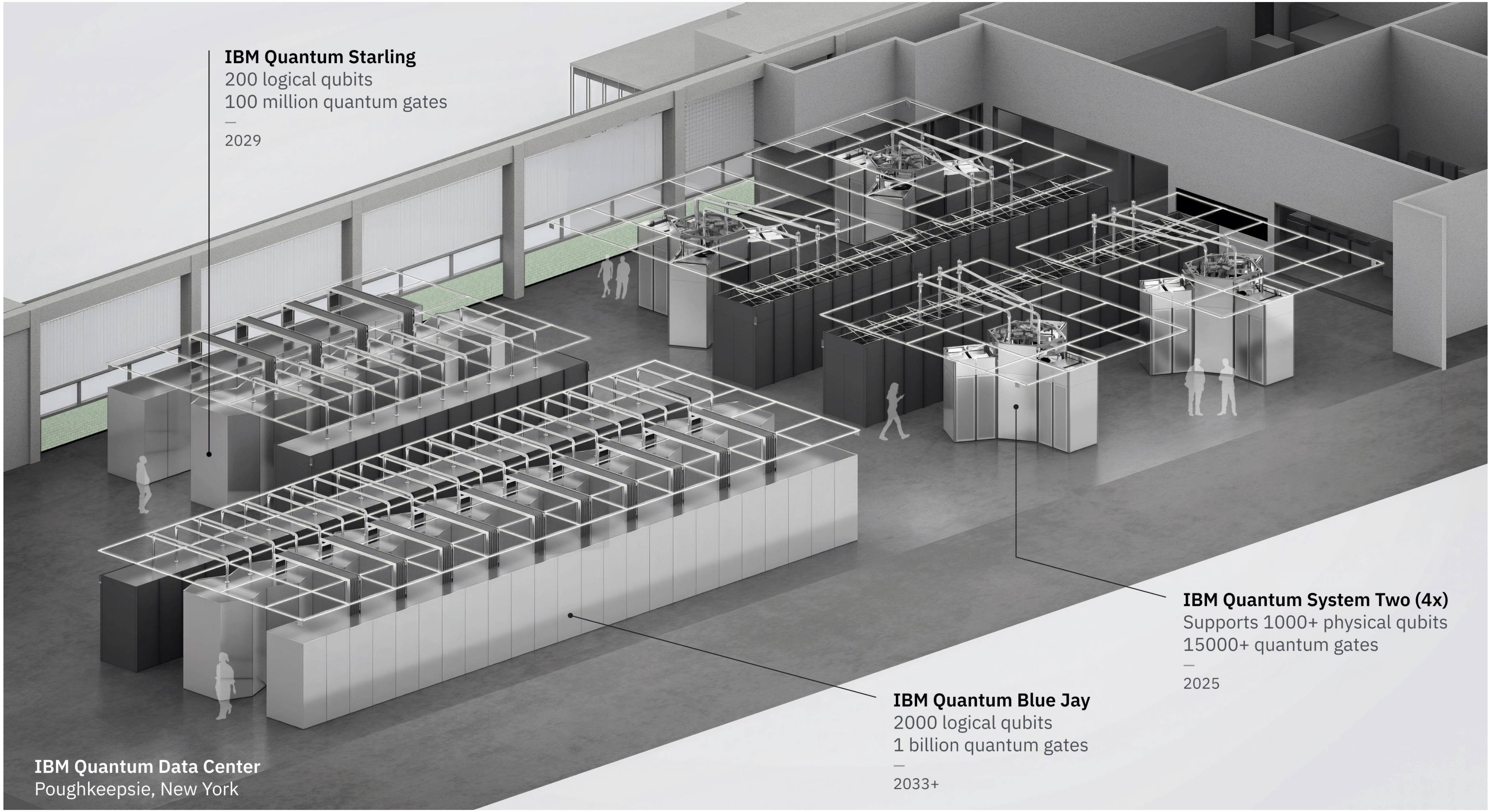
IBM Roadmap


2024
2025
2026
2027
2028
2029
2033+
Demonstrated accurate execution of a quantum circuit at a scale beyond exact classical simulation (5K gates on 156 qubits)
Deliver quantum + HPC tools that will leverage Nighthawk, a new higher-connectivity quantum processor able to execute more complex circuits
Enable the first examples of quantum advantage using a quantum computer with HPC
Improve quantum circuit quality to allow 10K gates
Improve quantum circuit quality to allow 15K gates
Deliver a fault-tolerant quantum computer with the ability to run 100M gates on 200 logical qubits
Beyond 2033, quantum computers will run circuits comprising a billion gates on up to 2000 logical qubits, unlocking the full power of quantum computing
Code assistant
Functions
Use case benchmarking toolkit
Computation libraries
Advanced classical transpilation tools
Advanced classical mitigation tools
Utility mapping tools
Circuit libraries
Plugins for HPC
C API
Profiling tools
Workflow accelerators
200K CLOPS
Utility-scale dynamic circuits
Fault-tolerant ISA
Heron (5K)
Error mitigation
5K gates | 133 qubits
Nighthawk (5K)
Error mitigation
5K gates | 120 qubits
Nighthawk (7.5K)
Error mitigation
7.5K gates | 120 qubits
Up to 120x3 = 360 qubits
Nighthawk (10K)
Error mitigation
10K gates | 120 qubits
Up to 120x9 = 1080 qubits
Nighthawk (15K)
Error mitigation
15K gates | 120 qubits
Up to 120x9 = 1080 qubits
Starling (100M)
Fault-tolerant
100M gates
200 logical qubits
Blue Jay (1B)
Fault-tolerant
1B gates
2000 logical qubits

Evolution of Number of Qubits

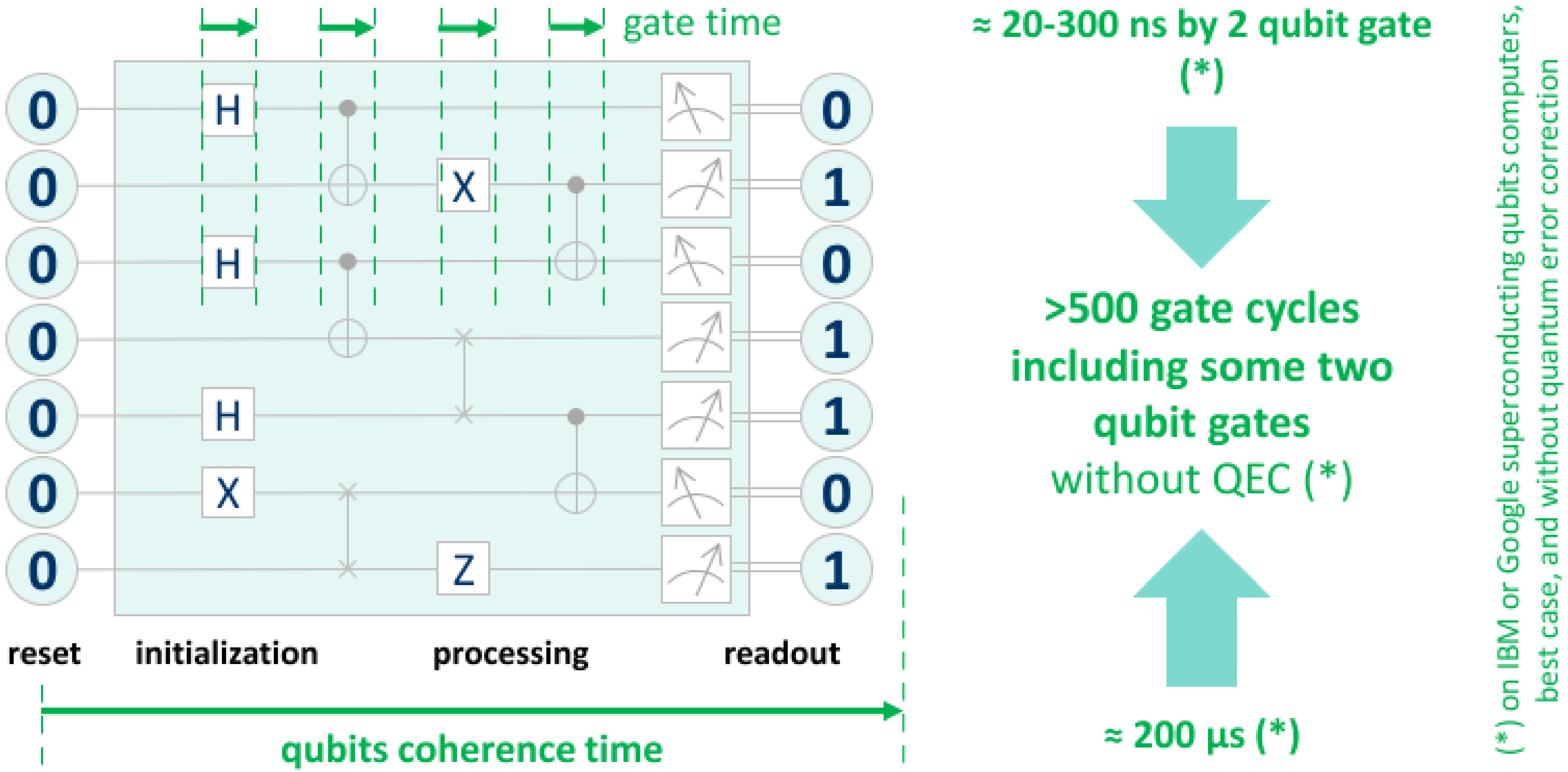


QC in HPC (High Performance Computing)

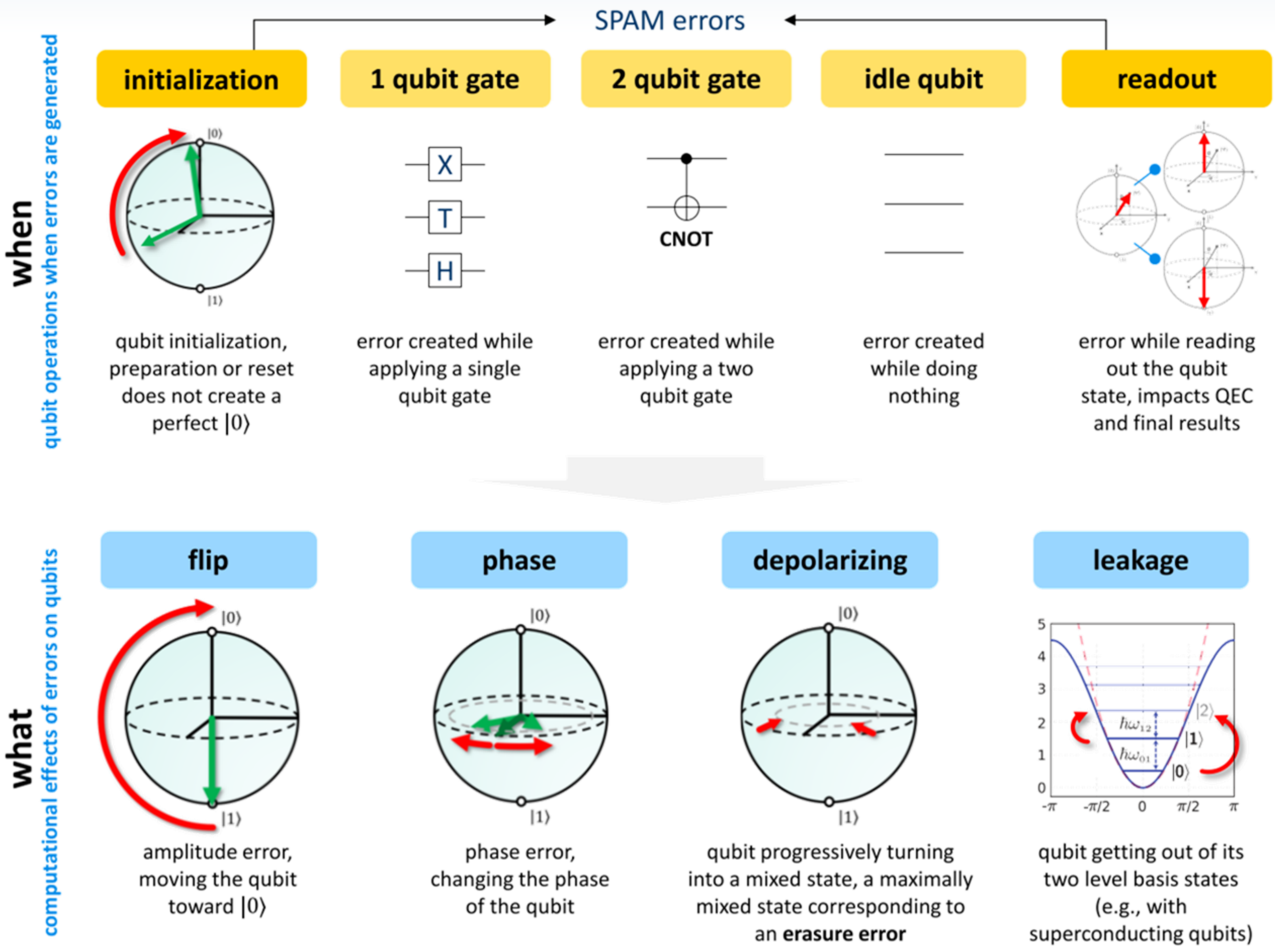


Quantum computing challenges: Decoherence

- Limited timeframe where qubits are useful



Quantum computing challenges: Error Correction



Causes:

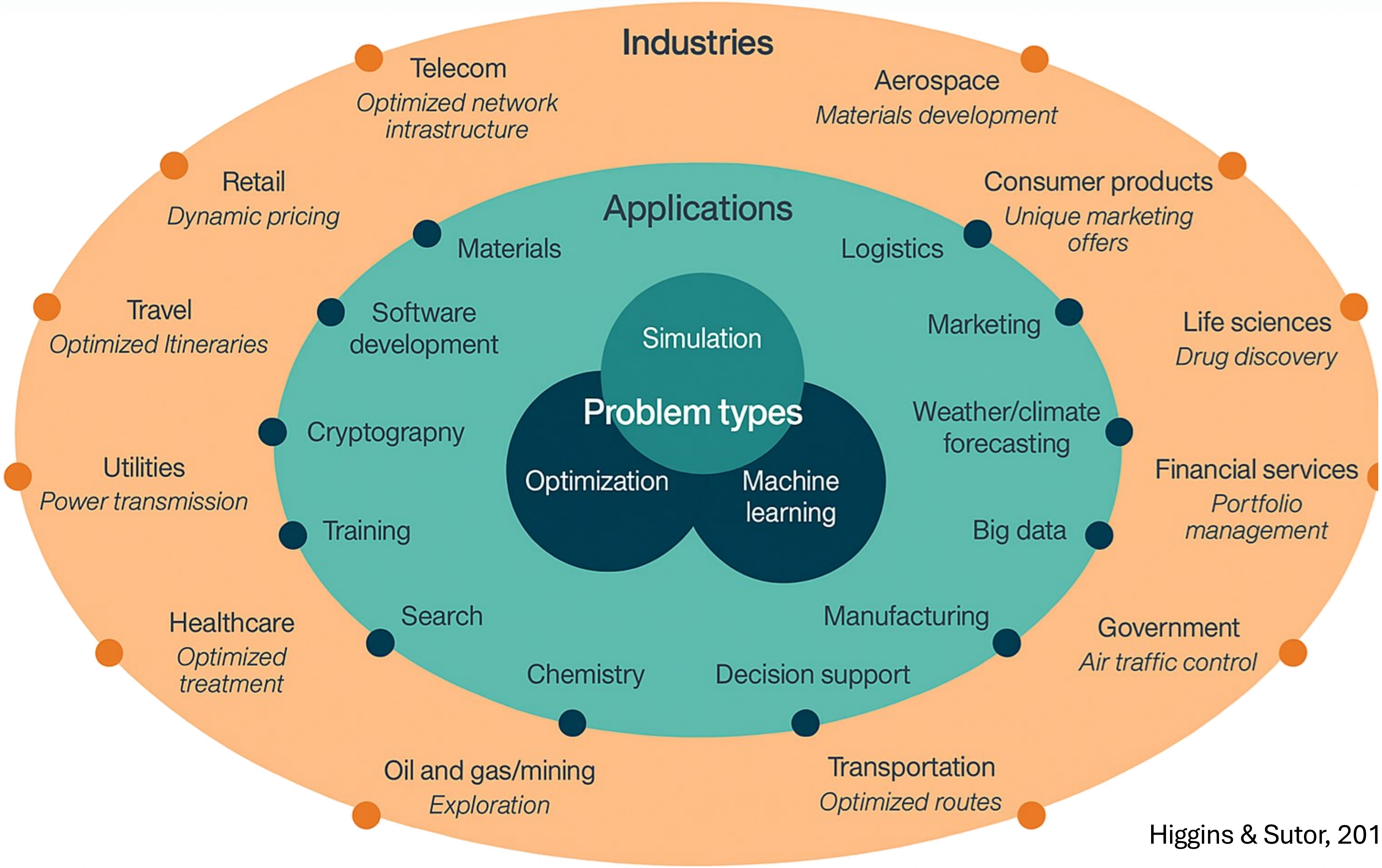
- Control
- Calibration
- Material defects
- Cosmic rays
- Multiple interactions
- Thermal noise
- Electromagnetic noise
- Photon los
- ...

Agenda

- Quantum Computing
- **Quantum Computing Applications**
- Quantum Software & Quantum Software Engineering
- Our Research Journey
- Conclusions

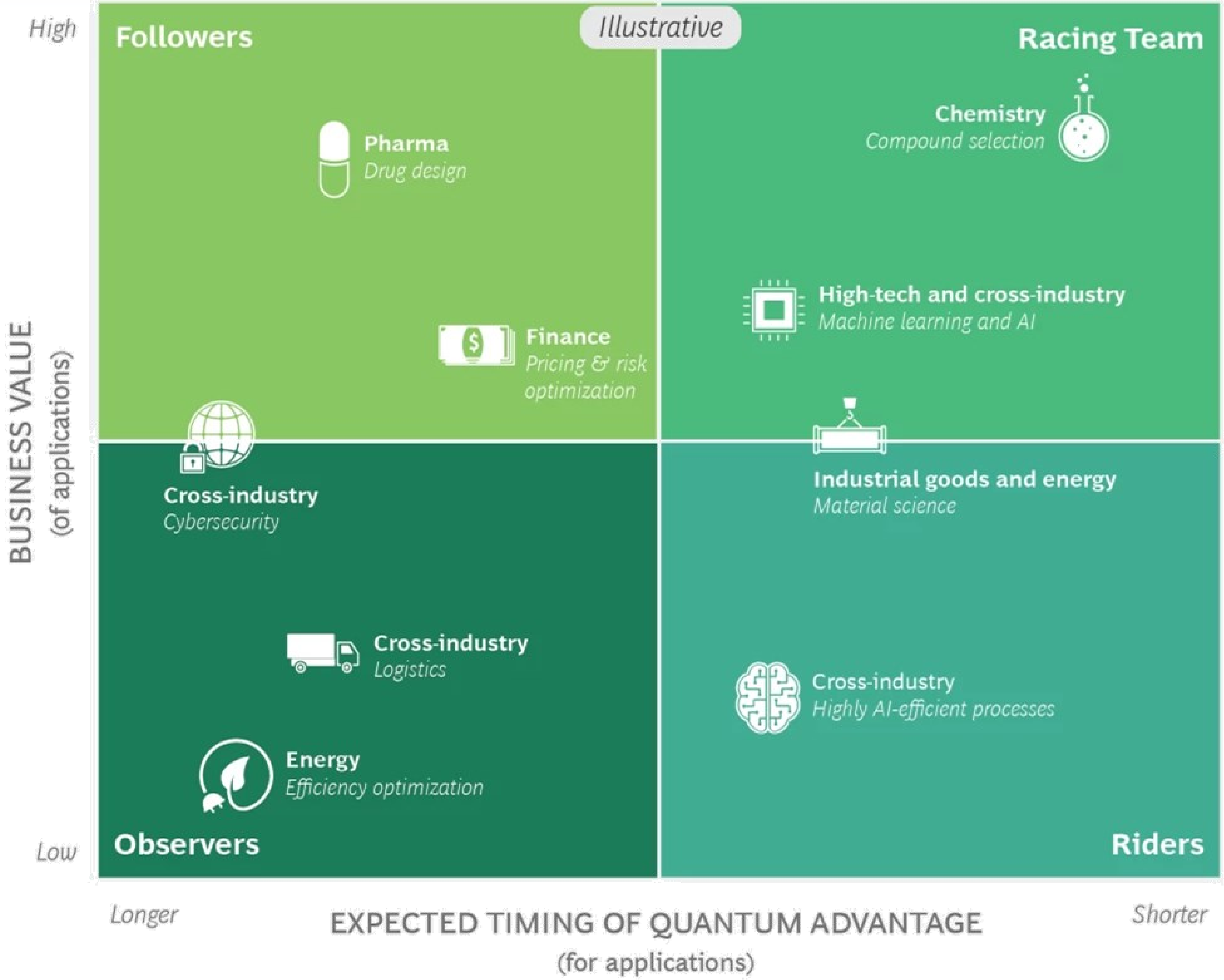
Scan for slides

Expected QC Applications



Higgins & Sutor, 2018

Expected QC Applications



HOW TO PLAY

Racing Team
Build superior QC network
Launch new offerings

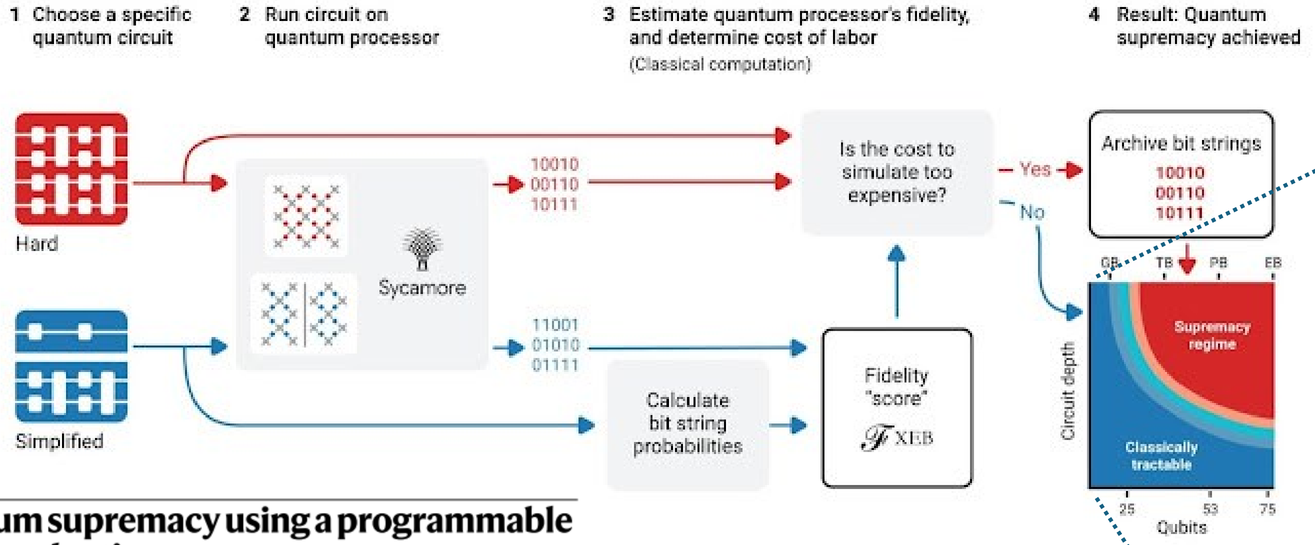
Riders
Engage with QC ecosystem
Lead own effort

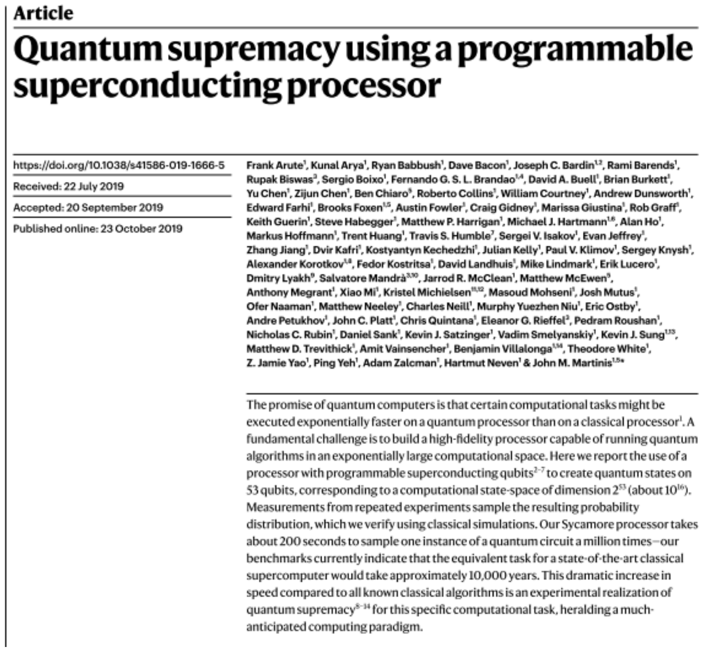
Followers
Participate in QC networks
Gain experience

Observers
Monitor QC ecosystem
Analyze business potential

Sources: BCG analysis.

Quantum Supremacy?





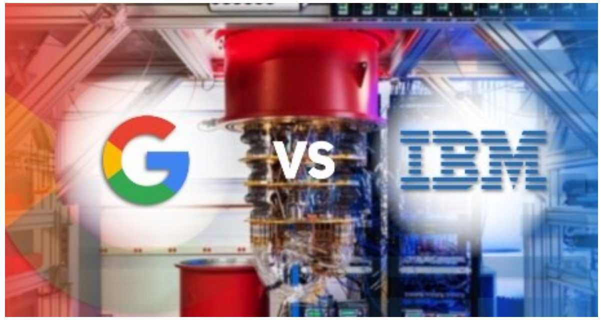
Article

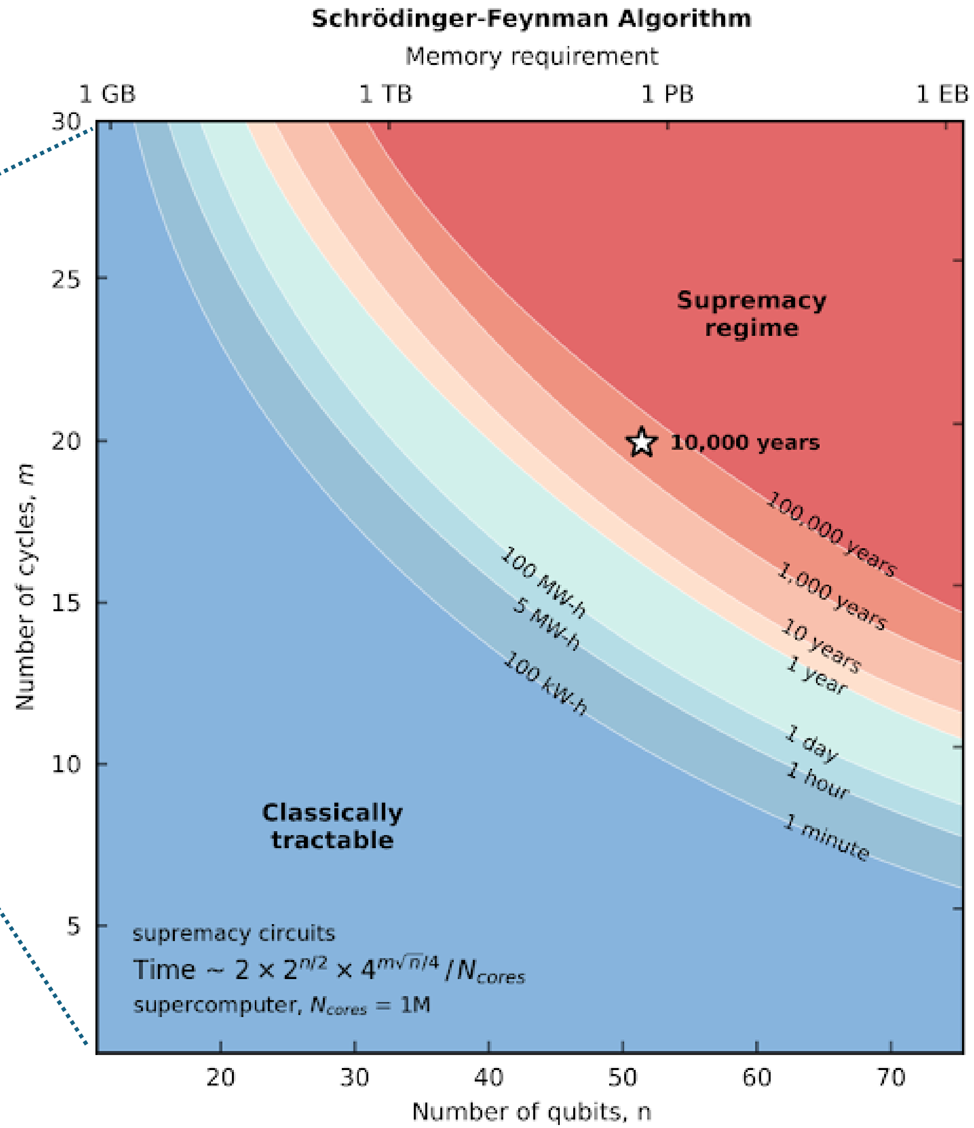
Quantum supremacy using a programmable superconducting processor

https://doi.org/10.1038/s41586-019-1666-5
Received: 22 July 2019
Accepted: 20 September 2019
Published online: 23 October 2019

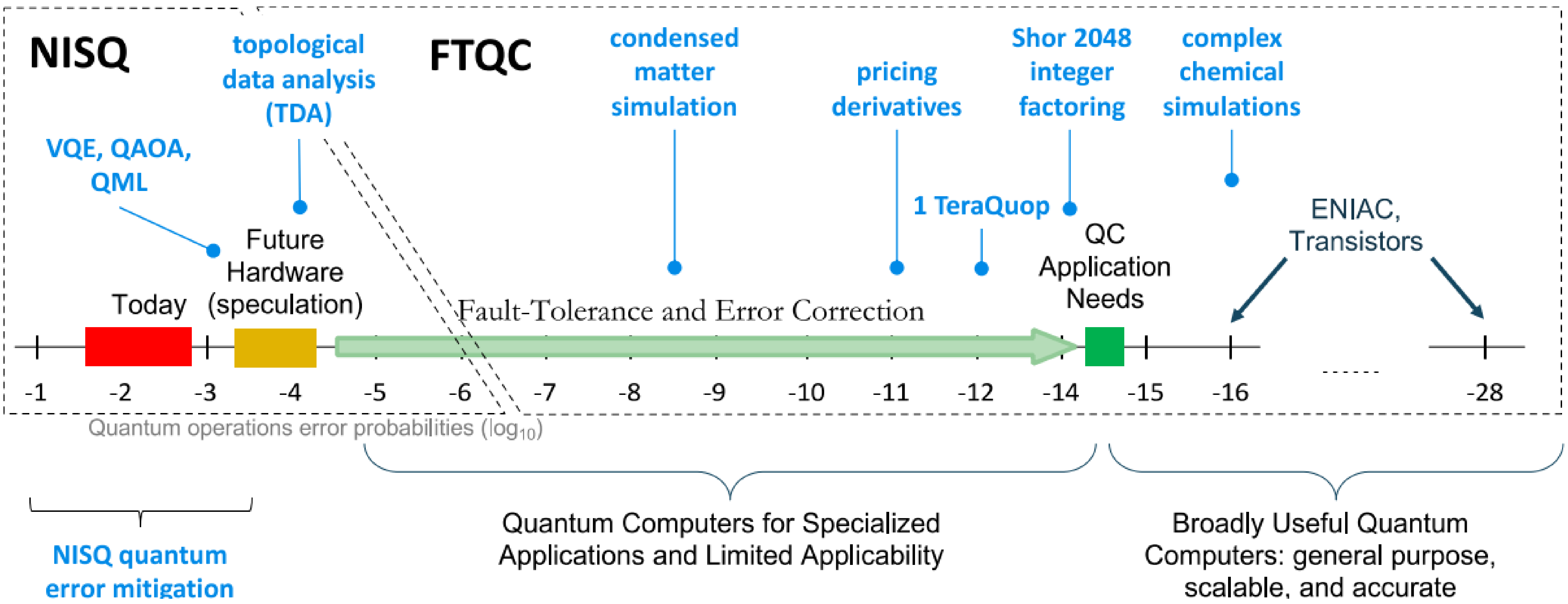
Frank Arute[1], Kunal Arya[1], Ryan Babbush[1], Dave Bacon[1], Joseph C. Bardin[1,2], Rami Barends[1], Rupak Biswas[3], Sergio Boixo[1], Fernando G. S. L. Brandao[1,4], David A. Buell[1], Brian Burkett[1], Yu Chen[1], Zijun Chen[1], Ben Chiaro[5], Roberto Collins[1], William Courtney[1], Andrew Dunsworth[1], Edward Farhi[1], Brooks Foxen[1,5], Austin Fowler[1], Craig Gidney[1], Marissa Giustina[1], Rob Graff[1], Keith Guerin[1], Steve Habegger[1], Matthew P. Harrigan[1], Michael J. Hartmann[1,6], Alan Ho[1], Markus Hoffmann[1], Trent Huang[1], Travis S. Humble[7], Sergei V. Isakov[1], Evan Jeffrey[1], Zhang Jiang[1], Dvir Kafri[1], Kostyantyn Kechedzhi[1], Julian Kelly[1], Paul V. Klimov[1], Sergey Knysh[1], Alexander Korotkov[1,8], Fedor Kostritsa[1], David Landhuis[1], Mike Lindmark[1], Erik Lucero[1], Dmitry Lyakh[9], Salvatore Mandrà[3,10], Jarrod R. McClean[1], Matthew McEwen[5], Anthony Megrant[1], Xiao Mi[1], Kristel Michielsen[11,12], Masoud Mohseni[1], Josh Mutus[1], Ofer Naaman[1], Matthew Neeley[1], Charles Neill[1], Murphy Yuezhen Niu[1], Eric Ostby[1], Andre Petukhov[1], John C. Platt[1], Chris Quintana[1], Eleanor G. Rieffel[3], Pedram Roushan[1], Nicholas C. Rubin[1], Daniel Sank[1], Kevin J. Satzinger[1], Vadim Smelyanskiy[1], Kevin J. Sung[1,13], Matthew D. Trevithick[1], Amit Vainsencher[1], Benjamin Villalonga[1,14], Theodore White[1], Z. Jamie Yao[1], Ping Yeh[1], Adam Zalcman[1], Hartmut Neven[1] & John M. Martinis[1,5]*

The promise of quantum computers is that certain computational tasks might be executed exponentially faster on a quantum processor than on a classical processor[1]. A fundamental challenge is to build a high-fidelity processor capable of running quantum algorithms in an exponentially large computational space. Here we report the use of a processor with programmable superconducting qubits[2–7] to create quantum states on 53 qubits, corresponding to a computational state-space of dimension 2^{53} (about 10^{16}). Measurements from repeated experiments sample the resulting probability distribution, which we verify using classical simulations. Our Sycamore processor takes about 200 seconds to sample one instance of a quantum circuit a million times—our benchmarks currently indicate that the equivalent task for a state-of-the-art classical supercomputer would take approximately 10,000 years. This dramatic increase in speed compared to all known classical algorithms is an experimental realization of quantum supremacy[8–14] for this specific computational task, heralding a much-anticipated computing paradigm.

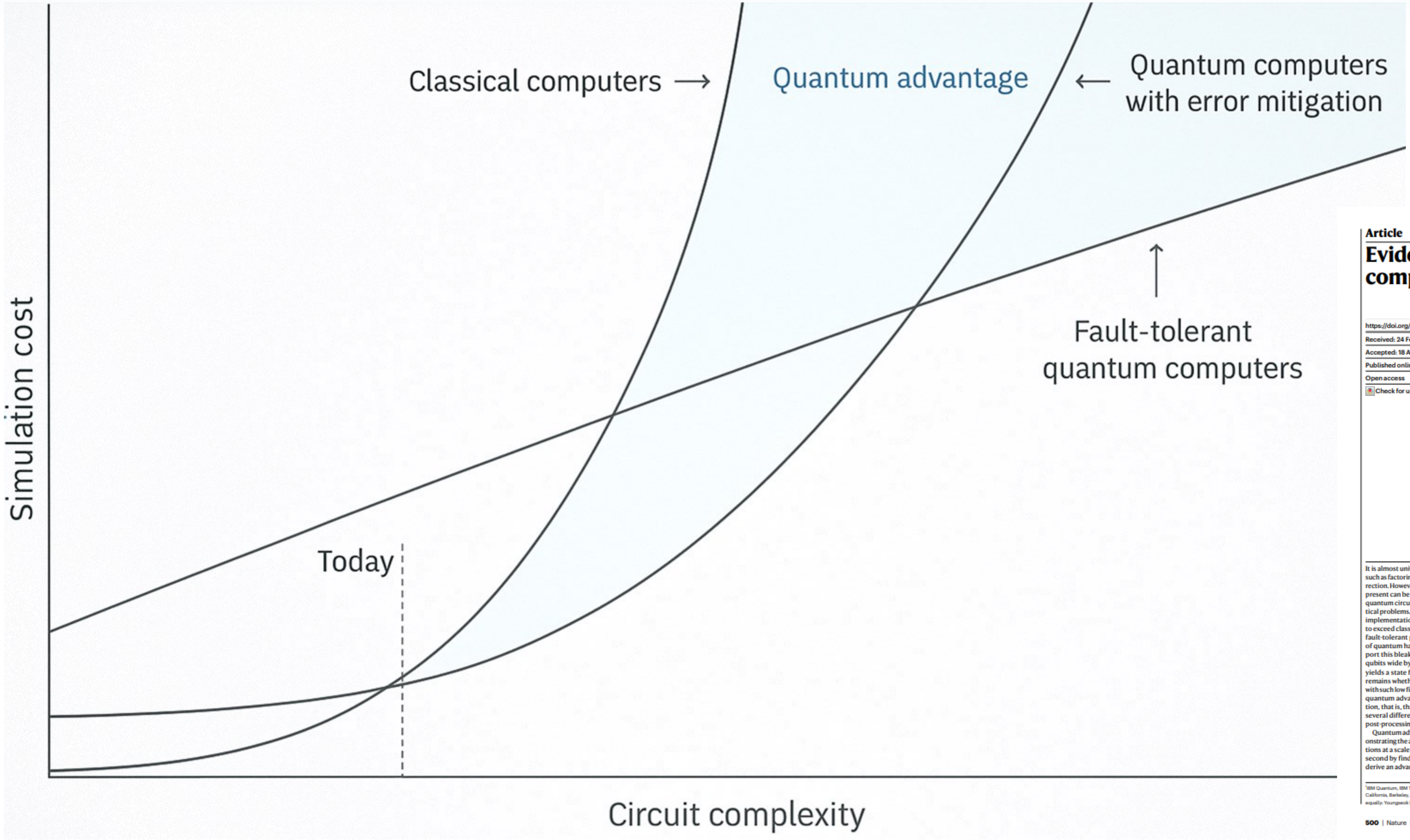




NISQ (Noisy Intermediate-Scale Quantum) vs. **FTQC** (Fault-Tolerant Quantum Computing)



Quantum Advantage (expected by 2026?)



Article

Evidence for the utility of quantum computing before fault tolerance

https://doi.org/10.1038/s41586-023-06096-3

Received: 24 February 2023

Accepted: 18 April 2023

Published online: 14 June 2023

Open access

Check for updates

Youngseok Kim[1,6✉], Andrew Eddins[2,6✉], Sajant Anand[3], Ken Xuan Wei[1], Ewout van den Berg[1], Sami Rosenblatt[1], Hasan Nayfeh[1], Yantao Wu[3,4], Michael Zaletel[3,5], Kristan Temme[1] & Abhinav Kandala[1✉]

Quantum computing promises to offer substantial speed-ups over its classical counterpart for certain problems. However, the greatest impediment to realizing its full potential is noise that is inherent to these systems. The widely accepted solution to this challenge is the implementation of fault-tolerant quantum circuits, which is out of reach for current processors. Here we report experiments on a noisy 127-qubit processor and demonstrate the measurement of accurate expectation values for circuit volumes at a scale beyond brute-force classical computation. We argue that this represents evidence for the utility of quantum computing in a pre-fault-tolerant era. These experimental results are enabled by advances in the coherence and calibration of a superconducting processor at this scale and the ability to characterize[1] and controllably manipulate noise across such a large device. We establish the accuracy of the measured expectation values by comparing them with the output of exactly verifiable circuits. In the regime of strong entanglement, the quantum computer provides correct results for which leading classical approximations such as pure-state-based 1D (matrix product states, MPS) and 2D (isometric tensor network states, isoTNS) tensor network methods[2,3] break down. These experiments demonstrate a foundational tool for the realization of near-term quantum applications[4,5].

It is almost universally accepted that advanced quantum algorithms such as factoring[6] or phase estimation[7] will require quantum error correction. However, it is acutely debated whether processors available at present can be made sufficiently reliable to run other, shorter-depth quantum circuits at a scale that could provide an advantage for practical problems. At this point, the conventional expectation is that the implementation of even simple quantum circuits with the potential to exceed classical capabilities will have to wait until more advanced, fault-tolerant processors arrive. Despite the tremendous progress of quantum hardware in recent years, simple fidelity bounds[8] support this bleak forecast; one estimates that a quantum circuit 100 qubits wide by 100 gate-layers deep executed with 0.1% gate error yields a state fidelity less than 5×10^{-4}. Nonetheless, the question remains whether properties of the ideal state can be accessed even with such low fidelities. The error-mitigation[9,10] approach to near-term quantum advantage on noisy devices exactly addresses this question, that is, that one can produce accurate expectation values from several different runs of the noisy quantum circuit using classical post-processing.

Quantum advantage can be approached in two steps: first, by demonstrating the ability of existing devices to perform accurate computations at a scale that lies beyond brute-force classical simulation, and second by finding problems with associated quantum circuits that derive an advantage from these devices. Here we focus on taking the first step and do not aim to implement quantum circuits for problems with proven speed-ups.

We use a superconducting quantum processor with 127 qubits to run quantum circuits with up to 60 layers of two-qubit gates, a total of 2,880 CNOT gates. General quantum circuits of this size lie beyond what is feasible with brute-force classical methods. We thus first focus on specific test cases of the circuits permitting exact classical verification of the measured expectation values. We then turn to circuit regimes and observables in which classical simulation becomes challenging and compare with results from state-of-the-art approximate classical methods.

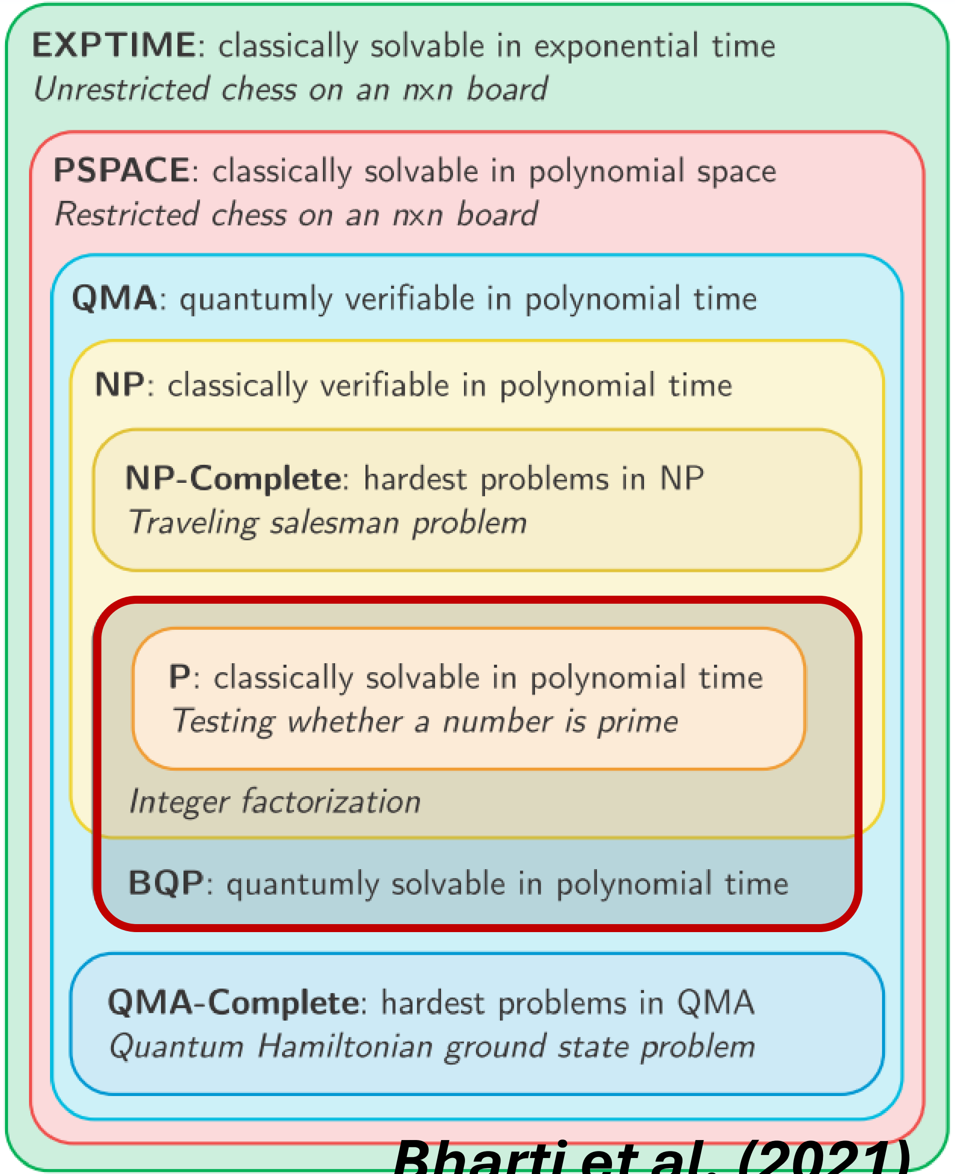
Our benchmark circuit is the Trotterized time evolution of a 2D transverse-field Ising model, sharing the topology of the qubit processor (Fig. 1a). The Ising model appears extensively across several areas in physics and has found creative extensions in recent simulations exploring quantum many-body phenomena, such as time crystals[11,12], quantum scars[13] and Majorana edge modes[14]. As a test of utility of quantum computation, however, the time evolution of the 2D transverse-field Ising model is most relevant in the limit of large entanglement growth in which scalable classical approximations struggle.

In particular, we consider time dynamics of the Hamiltonian,

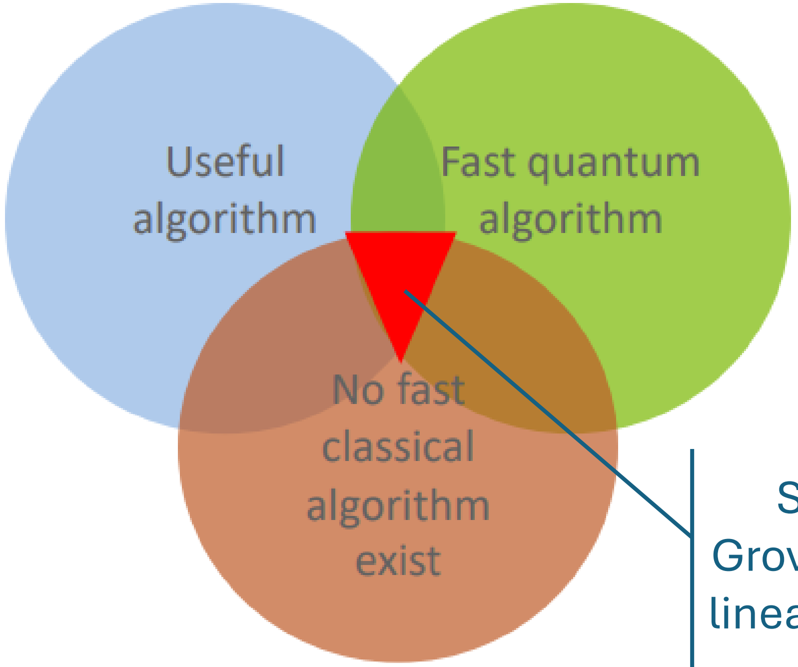
$$H = -J \sum_{\langle i,j \rangle} Z_i Z_j + h \sum_i X_i, \tag{1}$$

[1]IBM Quantum, IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA. [2]IBM Quantum, IBM Research - Cambridge, Cambridge, MA, USA. [3]Department of Physics, University of California, Berkeley, Berkeley, CA, USA. [4]RIKEN iTHEMS, Wako, Japan. [5]Materials Sciences Division, Lawrence Berkeley National Laboratory, Berkeley, CA, USA. [6]These authors contributed equally: Youngseok Kim, Andrew Eddins. ✉e-mail: youngseok.kim1@ibm.com; aeddins@ibm.com; akandala@us.ibm.com

Target Problems for QC

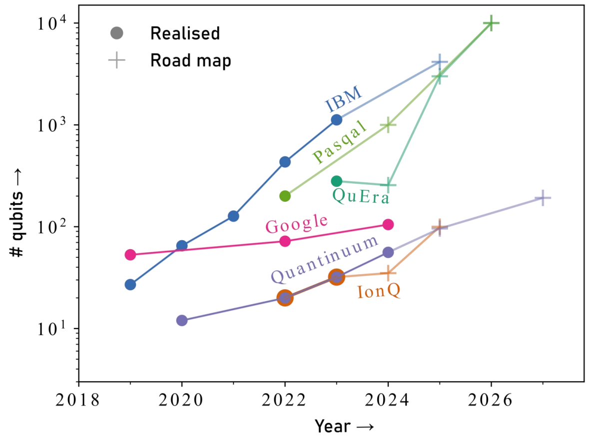


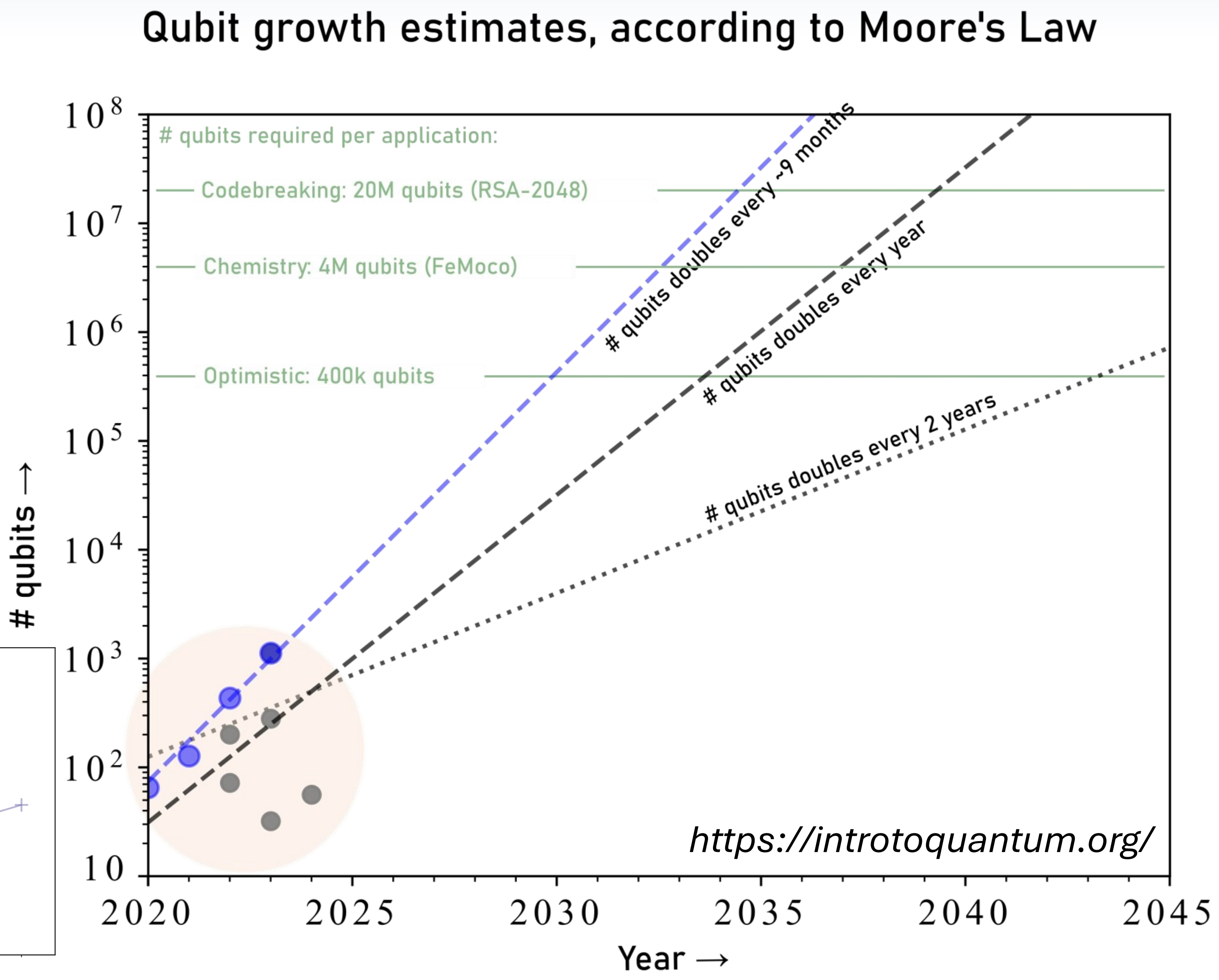
Bharti et al. (2021)



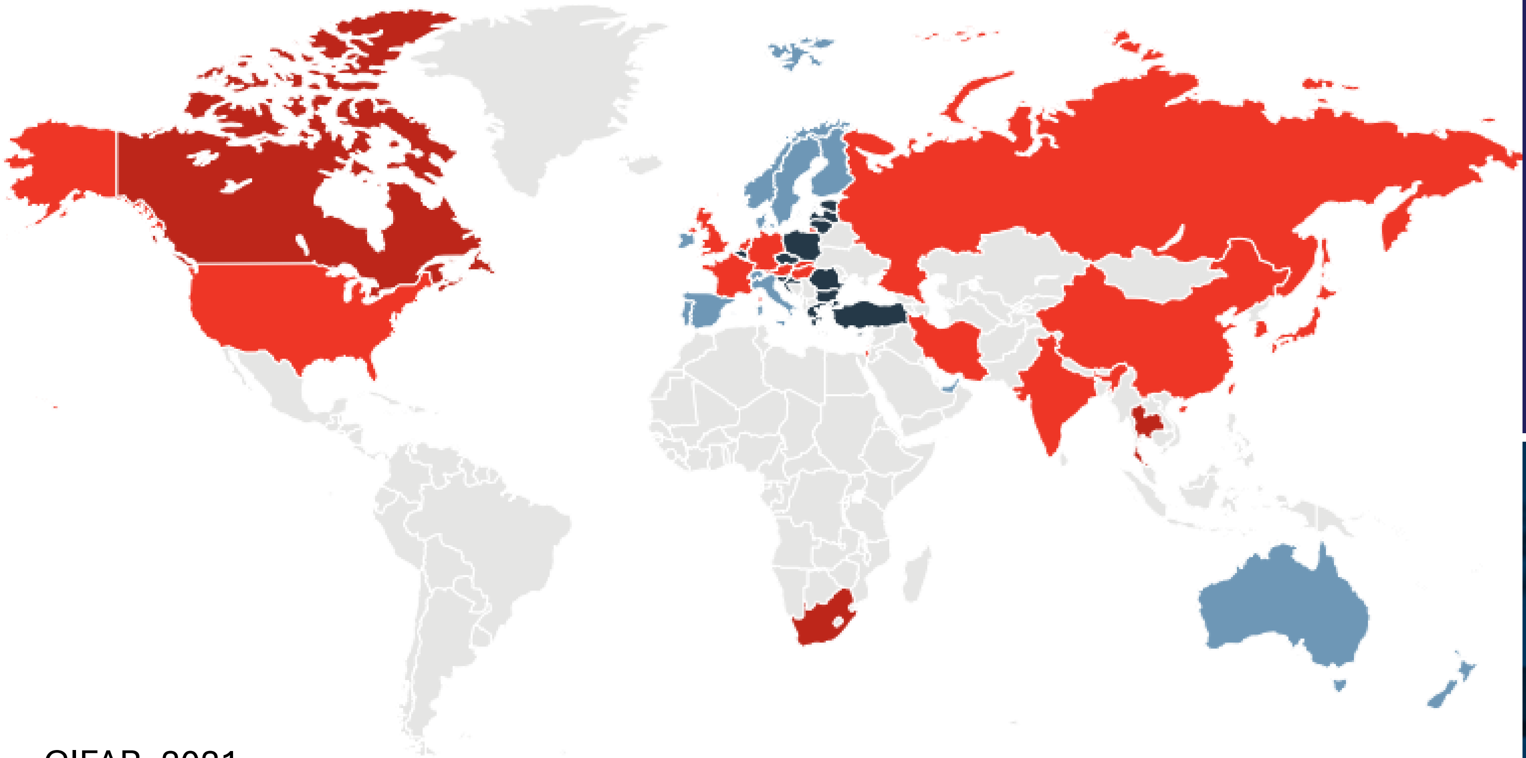
Shor's factoring, Grover's search, HHL's linear system solver,.... QML?

Predicted #Qubits for Quantum Advantage



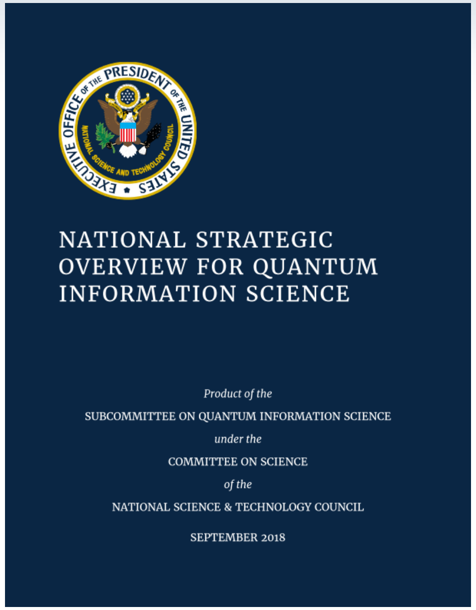


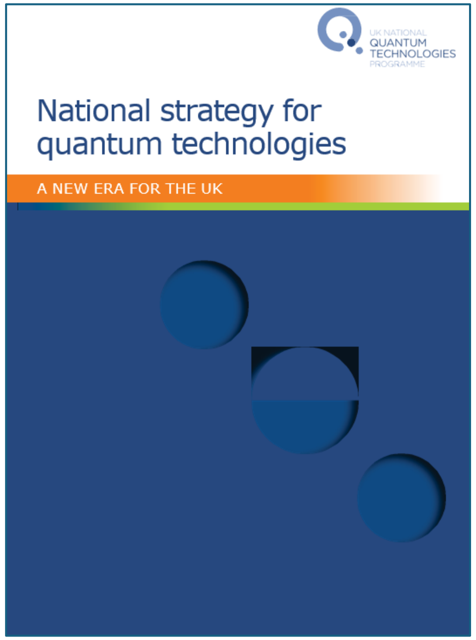
R&D Policies



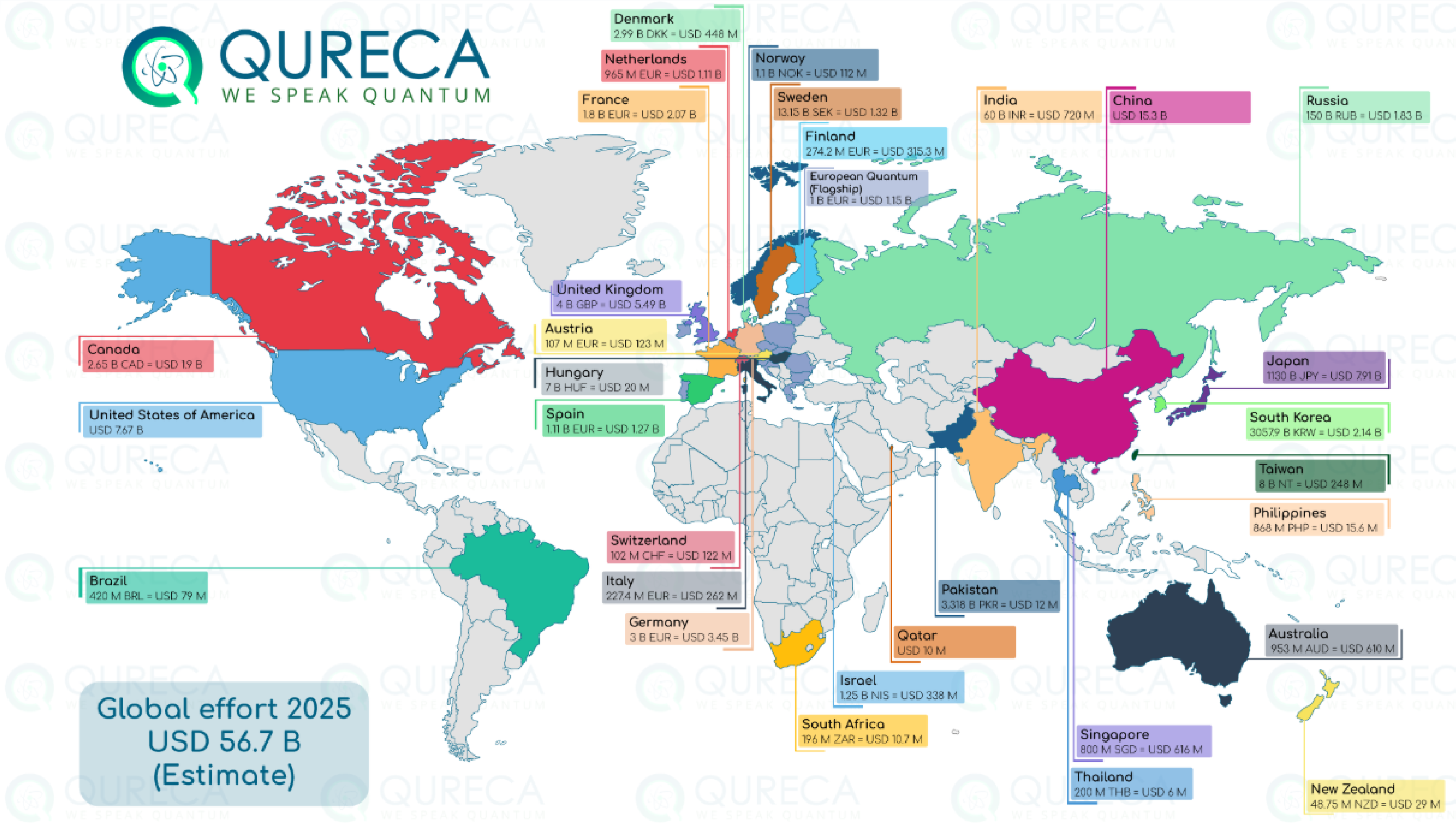
CIFAR, 2021

- Countries with coordinated national quantum strategy
- Quantum strategy in development
- Countries without national strategy, but with significant government or government-endorsed initiatives
- Countries without significant initiatives but are participants in international quantum partnerships





Public Investment



Agenda

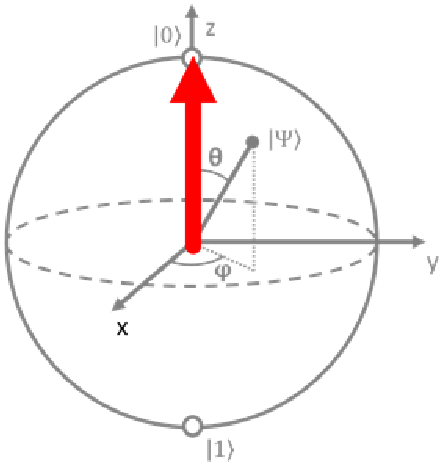
- Quantum Computing
- Quantum Computing Applications
- **Quantum Software & Quantum Software Engineering**
- Our Research Journey
- Conclusions

Scan for slides

Quantum Software works with Probability and Uncertainty

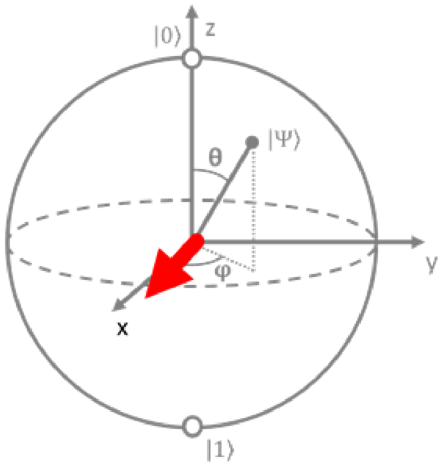
How to work with qubits?

initialization



$|0\rangle$

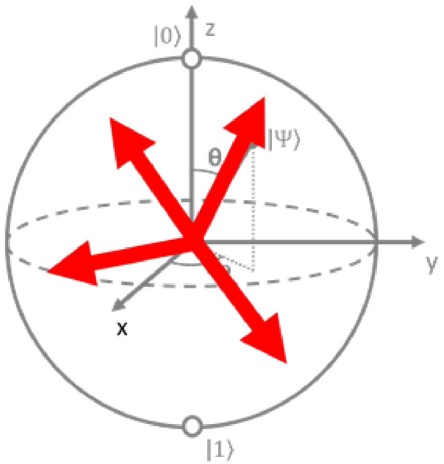
Hadamard gate



$$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

superposition of 0 and 1

other gates

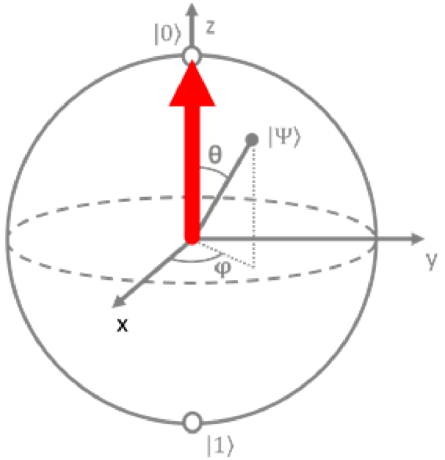


qubit vector turning around in Bloch sphere

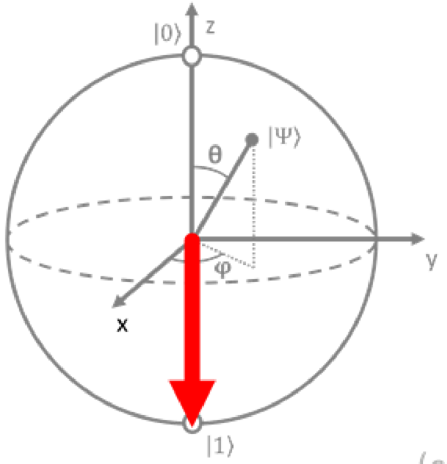
2 and 3 qubit gates will conditionally link qubits

during computing, the process is rather analog

measurement



measurement returns $|0\rangle$ with a probability α^2 depending on the qubit state, then qubit state becomes $|0\rangle$



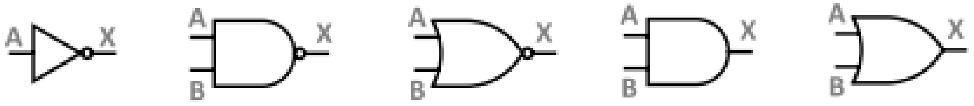
measurement returns $|1\rangle$ with a probability β^2 depending on the qubit state, then qubit state becomes $|1\rangle$

How to work with qubits?

classical logic gates

boolean algebra on 1 or 2 bits

NOT NAND NOR AND OR



truth table

NOT

A	X
0	1
1	0

NAND

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

NOR

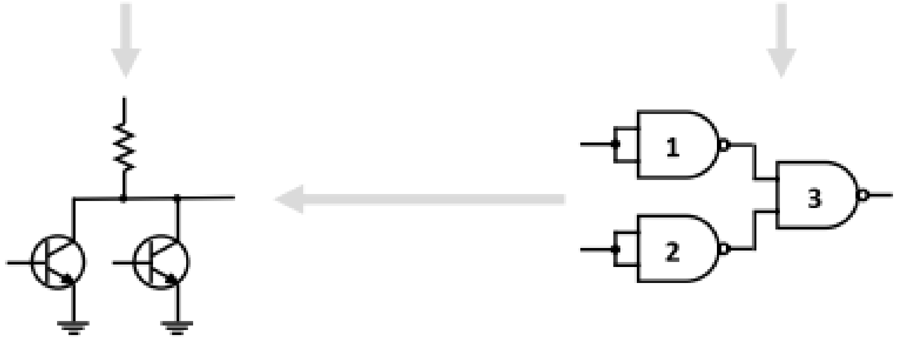
A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

AND

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1



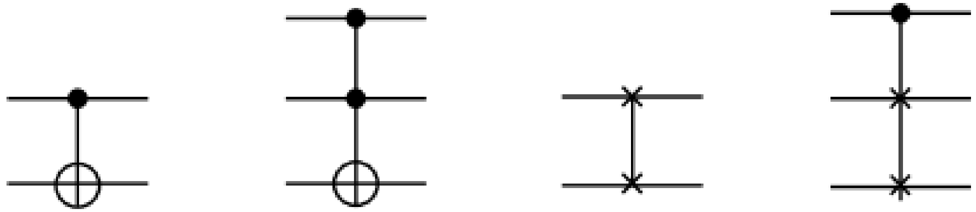
quantum gates

matrix based **unitary transformations**

rotation X NOT | **rotation Y** | **rotation Z** | **superposition** Hadamard

uitary matrixes

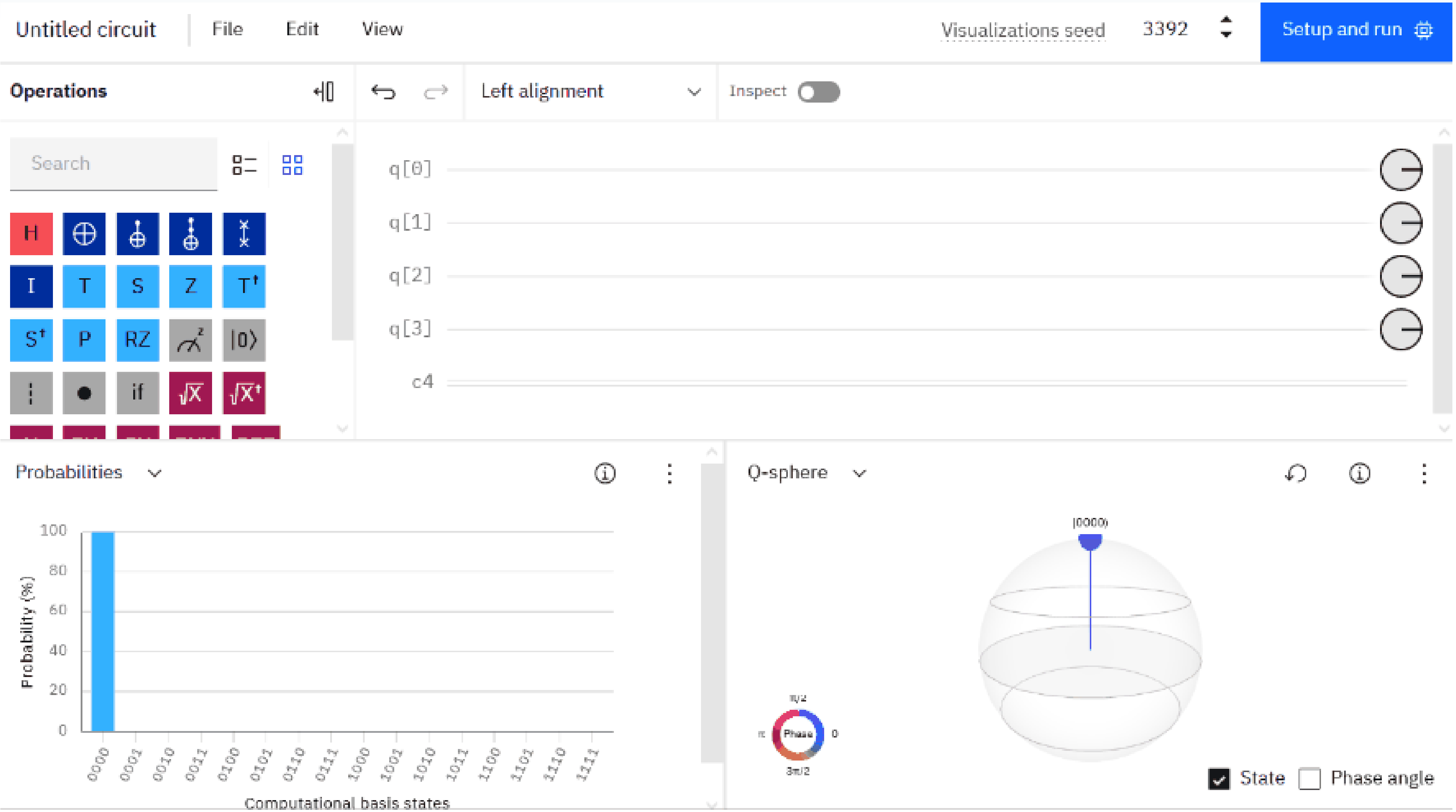
$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$



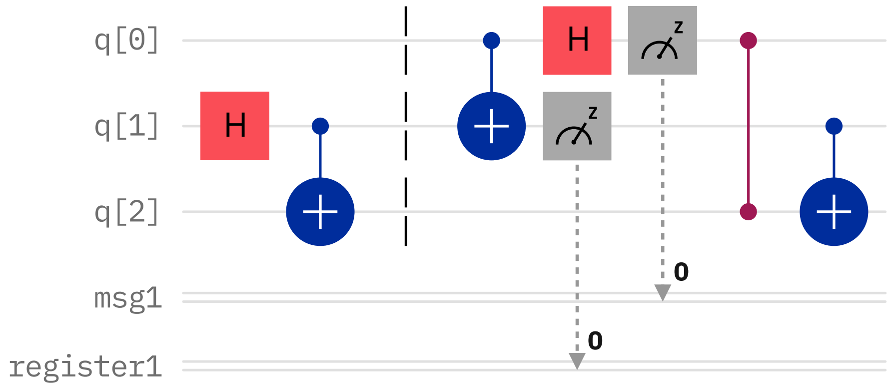
CNOT | **C2NOT** Toffoli | **SWAP** | **Fredkin** conditional SWAP

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

How to work with qubits?



Quantum Circuits & Quantum Programming Languages




```
from qiskit import QuantumRegister, ClassicalRegister,
                   QuantumCircuit
from numpy import pi

qreg_q = QuantumRegister(3, 'q')
creg_msg = ClassicalRegister(1, 'msg')
creg_register = ClassicalRegister(1, 'register')
circuit = QuantumCircuit(qreg_q, creg_msg, creg_register)

circuit.h(qreg_q[1])
circuit.cx(qreg_q[1], qreg_q[2])
circuit.barrier(qreg_q[1], qreg_q[0], qreg_q[2])
circuit.cx(qreg_q[0], qreg_q[1])
circuit.h(qreg_q[0])
circuit.measure(qreg_q[0], creg_msg[0])
circuit.cz(qreg_q[0], qreg_q[2])
circuit.measure(qreg_q[1], creg_register[0])
circuit.cx(qreg_q[1], qreg_q[2])
```

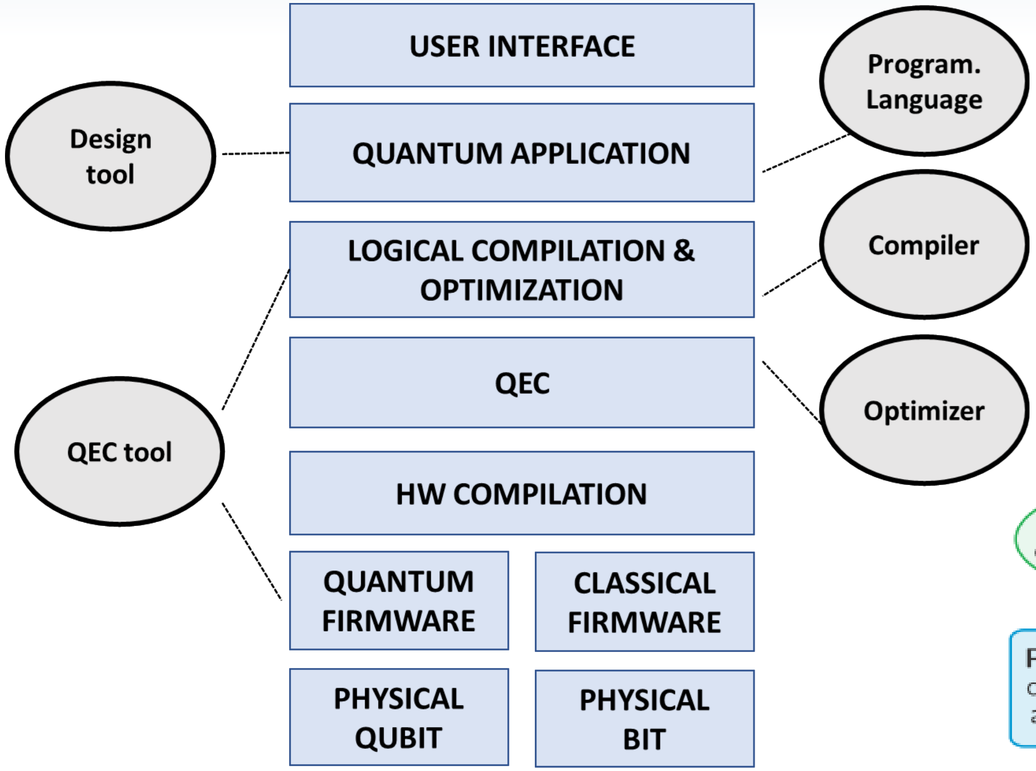

```
OPENQASM 2.0;
include "qelib1.inc";

qreg q[3];
creg msg[1];
creg register[1];

h q[1];
cx q[1],q[2];
barrier q[1],q[0],q[2];
cx q[0],q[1];
h q[0];
measure q[0] -> msg[0];
cz q[0],q[2];
measure q[1] -> register[0];
cx q[1],q[2];
```

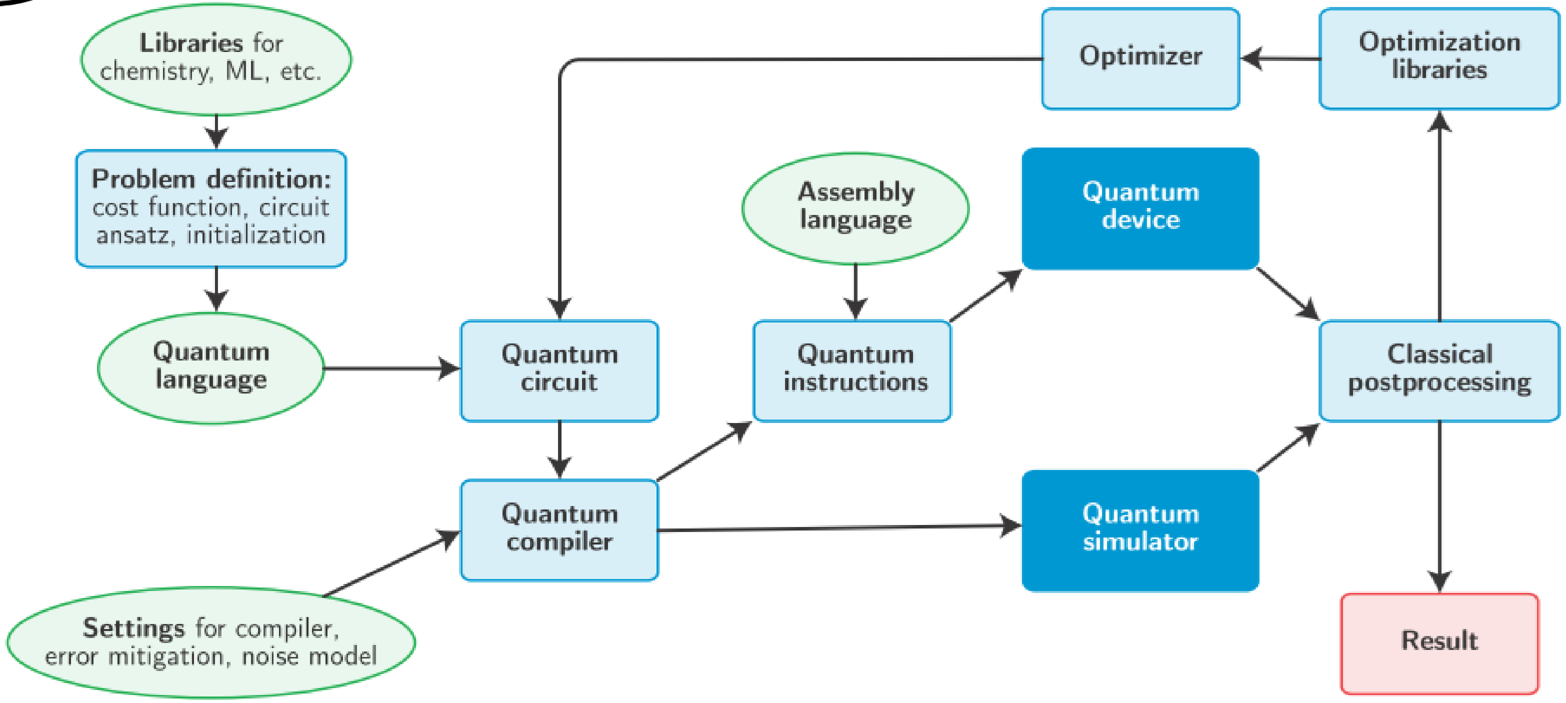
Quantum Programming Languages

Year	Programming Language	Base Language	Abstraction Level	Paradigm
1956	Quantum Lambda Calculi	Lambda calculus	High	Functional
1996	QCL	C	Low	Imperative
2000	qGCL	Pascal	High	Imperative
	A4	Lambda calculus	High	Functional
2003	Q language	C++	High	Imperative
2004	QEC (QPL)	Flowchart	High	Functional
	CQP	Process calculus	High	Other
2005	QML	Similar to Haskell	High	Functional
	qQPL	Denotational	High	Functional
	QPAlq	Process calculus	High	Other
2006	LanQ	C	Low	Imperative
2008	NDQJava	Java	High	Imperative
2009	Cove	C#	High	Imperative
2011	QuECT	Java	High	Other
2012	Scaffold	C (C++)	Low/High	Imperative
2013	Chisel-Q	Scala	High	Multiparadigm
	QuaFL	Haskell	High	Functional
	Quipper	Haskell	High	Functional
2014	LIQUi|>	F#	High	Functional
2015	Proto-Quipper	Haskell	High	Functional
2016	QASM	Assembly language	Low	Imperative
	FJQuantum	Feather-weight Java	High	Imperative
	ProjectQ	Python	High	Multiparadigm
	qPCF	Lambda calculus	High	Functional
	pyQuil (Quil)	Python	High	Functional
2017	OpenQASM	Assembly language	Low	Imperative
	GASM	Assembly language	Low	Imperative
	Python	Python	High	Multiparadigm
	FOREST	Haskell	High	Other
	OWtRE	Coq proof assistant	High	Other
2018	IQu	Idealized Algol	High	Imperative
	Strawberry Fields	Python	High	Multiparadigm
	Blackbird	Python	High	Multiparadigm
	QuantumOptics jl	Python	High	Imperative
	C#	C#	High	Multiparadigm
	Q|SI>	.Net language	High	Imperative
2020	Silq	Python	High	Imperative
	Jaquiil	Assembly language	High	Imperative

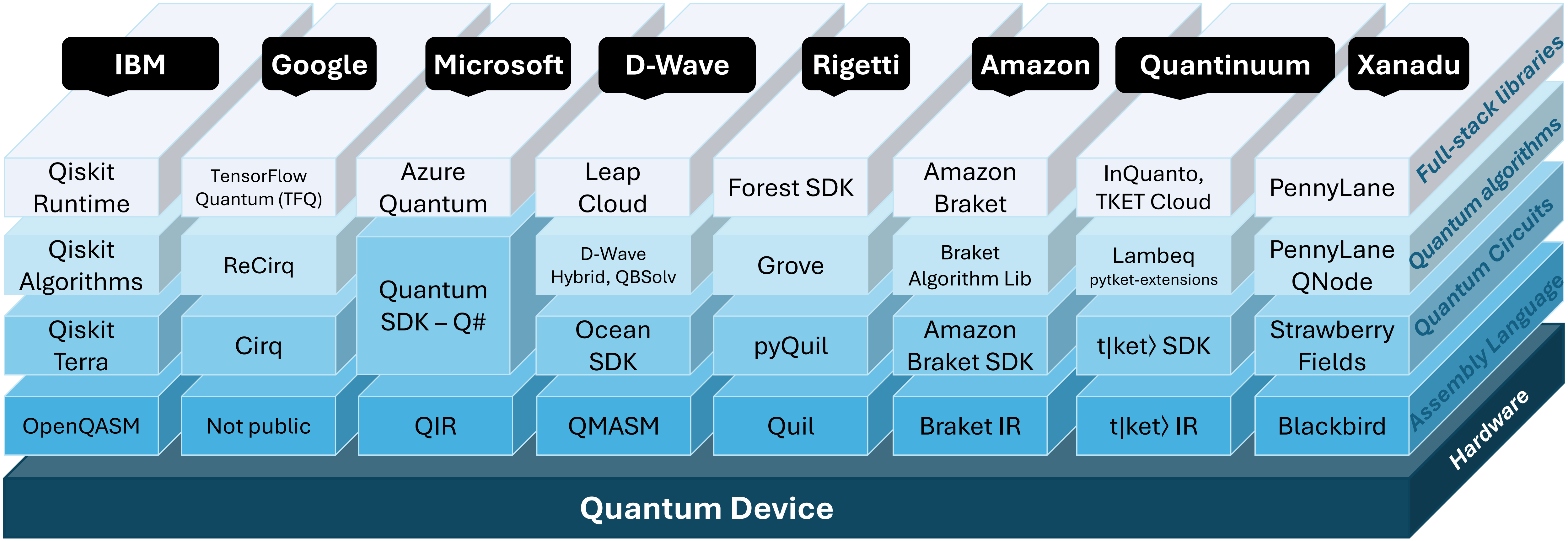
Manuel A. Serrano, José A. Cruz-Lemus, Ricardo Pérez-Castillo, Mario Piattini:
Quantum Software Components and Platforms: Overview and Quality Assessment. ACM Comput. Surv. 55(8): 164:1-164:31 (2023)

Quantum Software Development Stacks





Quantum Software Development Stacks



Quantum Software Engineering (QSE)

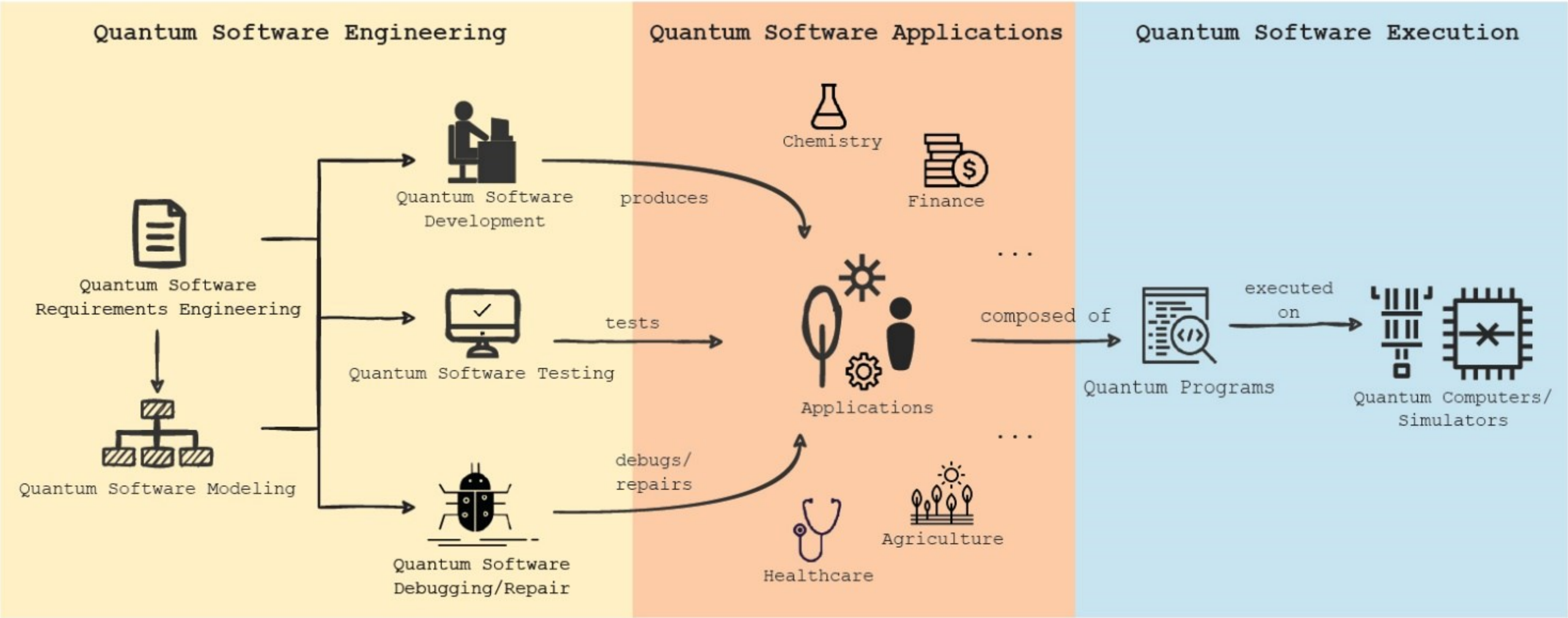
Quantum Software Engineering

Quantum Programming

Quantum Computer Science: Algorithms

Quantum Computers: Technology

Quantum Mechanics: Physics



- Understanding both low and high-level algorithms
- Know-how about the way these algorithms can be assembled and coupled with classical algorithms
- How to find the ways to translate “business problems” into these algorithms.

The Talavera Manifesto for Quantum Soft. Eng. and Programming (2020)

https://www.aquantum.es/manifesto/

The Talavera Manifesto for Quantum Soft. Eng. and Programming (2020)

Principles

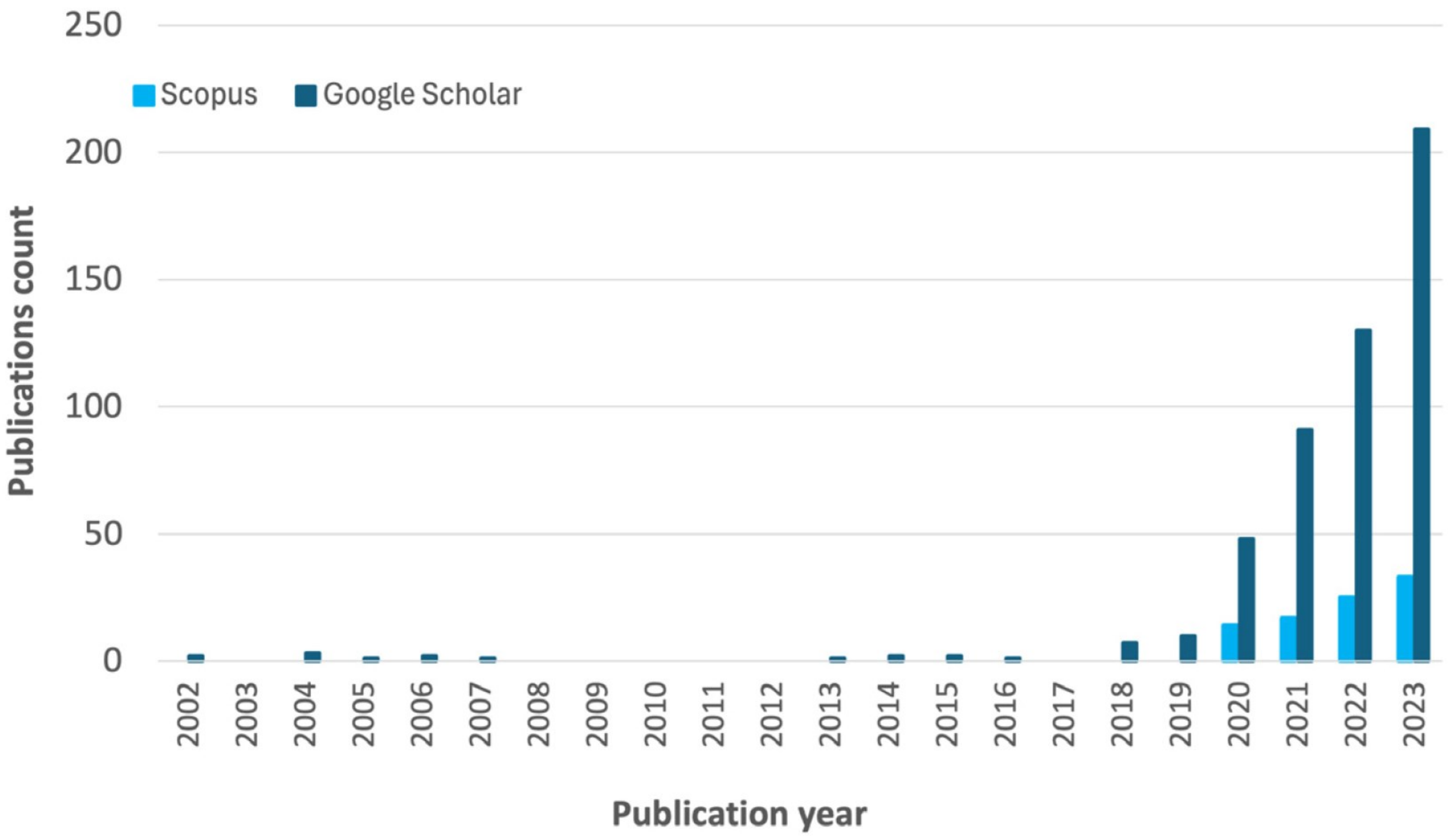
1. QSE is **agnostic** regarding quantum prog. languages and technologies
2. QSE embraces the **coexistence** of classical and quantum computing
3. QSE allows the management of quantum software development **projects**
4. QSE deals with the **evolution** of quantum software
5. QSE attempts to deliver quantum programs with desirable **zero defects**
6. QSE is concerned about the **quality** of quantum software
7. QSE is in favour of quantum software **reuse**
8. QSE is aware of the need for security and **privacy by design**
9. QSE covers the **governance and management** of software

QSE Roadmap and Challenges

Quantum Software Engineering: Roadmap and Challenges Ahead

JUAN MANUEL MURILLO, JOSE GARCIA-ALONSO, and ENRIQUE MOGUEL, University of Extremadura, Cáceres, Spain
JOHANNA BARZEN and FRANK LEYMANN, University of Stuttgart, Stuttgart, Germany
SHAUKAT ALI, Simula Research Laboratory, Oslo, Norway and Computer Science, Oslo Metropolitan University, Oslo, Norway
TAO YUE, Computer Science and Engineering, Beihang University, Beijing, China
PAOLO ARCAINI, National Institute of Informatics, Chiyoda-ku, Japan
RICARDO PÉREZ-CASTILLO, Faculty of Social Sciences and Information Technologies, University of Castilla-La Mancha, Talavera de la Reina, Spain
IGNACIO GARCÍA-RODRÍGUEZ DE GUZMÁN, Institute of Technologies and Information Systems, University of Castilla-La Mancha, Ciudad Real, Spain
MARIO PIATTINI, University of Castilla-La Mancha, Ciudad Real, Spain
ANTONIO RUIZ-CORTÉS, Computer Languages and Systems, University of Seville, Seville, Spain
ANTONIO BROGI, Department of Computer Science, University of Pisa, Pisa, Italy
JIANJUN ZHAO, Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka, Japan
ANDRIY MIRANSKYY, Toronto Metropolitan University, Toronto, Ontario, Canada
MANUEL WIMMER, Johannes Kepler University Linz, Linz, Austria

Research Interest on QSE



Murillo et al., 2025

Challenges on QSE

- **Quantum Software Testing**
  - Ch-ST-1. Efficient test oracles.
  - Ch-ST-2. Test scalability.
  - Ch-ST-3. From simulators to real quantum computers.
  - Ch-ST-4. (Quantum) AI and (quantum) software testing.
- **Quantum Service Oriented Computing**
  - Ch-SoC-1. Interoperability.
  - Ch-SoC-2. Platform independence.
  - Ch-SoC-3. Demand and capacity management.
  - Ch-SoC-4. Workforce training.
- **Quantum MDE**
  - Ch-MDE-1. Modeling quantum-specific constructs.
  - Ch-MDE-2. Development of high-level design methodologies.
  - Ch-MDE-3. Scalable quantum software maintenance and evolution.
  - Ch-MDE-4. Intelligent code generation and orchestration.
- **Quantum Programming Paradigms**
  - Ch-PP-1. Complexity of circuits.
  - Ch-PP-2. Composable and reusable quantum software.
  - Ch-PP-3. Abstractions for quantum software.
- **Quantum Software Architectures**
  - Ch-SA-1. Architectural decisions in quantum software.
  - Ch-SA-2. Design patterns for hybrid software systems.
  - Ch-SA-3. Empirical evidence for the application of design patterns.
  - Ch-SA-4. Evolution of hybrid software architectures.
- **Quantum Software Development Processes**
  - Ch-DP-1. Iterative development of hybrid software.
  - Ch-DP-2. Risk management.
  - Ch-DP-3. Project management.
- **Quantum Artificial Intelligence**
  - Ch-AI-1. Quantum circuit optimization.
  - Ch-AI-2. Developing hybrid AI-quantum workflows.
  - Ch-AI-3. Error mitigation and correction.
  - Ch-AI-4. Scalability of AI-assisted quantum software development.

Murillo et al., 2025

Challenges on QSE

- **Quantum Software Testing**
  - Ch-ST-1. Efficient test oracles.
  - Ch-ST-2. Test scalability.
  - Ch-ST-3. From simulators to real quantum computers.
  - Ch-ST-4. (Quantum) AI and (quantum) software testing.
- **Quantum Service Oriented Computing**
  - Ch-SoC-1. Interoperability.
  - Ch-SoC-2. Platform independence.
  - Ch-SoC-3. Demand and capacity management.
  - Ch-SoC-4. Workforce training.
- **Quantum MDE**
  - Ch-MDE-1. Modeling quantum-specific constructs.
  - Ch-MDE-2. Development of high-level design methodologies.
  - Ch-MDE-3. Scalable quantum software maintenance and evolution.
  - Ch-MDE-4. Intelligent code generation and orchestration.
- **Quantum Programming Paradigms**
  - Ch-PP-1. Complexity of circuits.
  - Ch-PP-2. Composable and reusable quantum software.
  - Ch-PP-3. Abstractions for quantum software.
- **Quantum Software Architectures**
  - Ch-SA-1. Architectural decisions in quantum software.
  - Ch-SA-2. Design patterns for hybrid software systems.
  - Ch-SA-3. Empirical evidence for the application of design patterns.
  - Ch-SA-4. Evolution of hybrid software architectures.
- **Quantum Software Development Processes**
  - Ch-DP-1. Iterative development of hybrid software.
  - Ch-DP-2. Risk management.
  - Ch-DP-3. Project management.
- **Quantum Artificial Intelligence**
  - Ch-AI-1. Quantum circuit optimization.
  - Ch-AI-2. Developing hybrid AI-quantum workflows.
  - Ch-AI-3. Error mitigation and correction.
  - Ch-AI-4. Scalability of AI-assisted quantum software development.

Murillo et al., 2025

Agenda

- Quantum Computing
- Quantum Computing Applications
- Quantum Software & Quantum Software Engineering
- **Our Research Journey**
- Conclusions

Scan for slides

Our Research Journey

- Reverse Engineering of Quantum Annealing programs to KDM

2020 | 2021 | 2022 | 2023 | 2024 | 2025 | …

- Talavera Manifesto
- Quantum Software Modernization Framework
- Reverse Engineering of Quantum Software to KDM

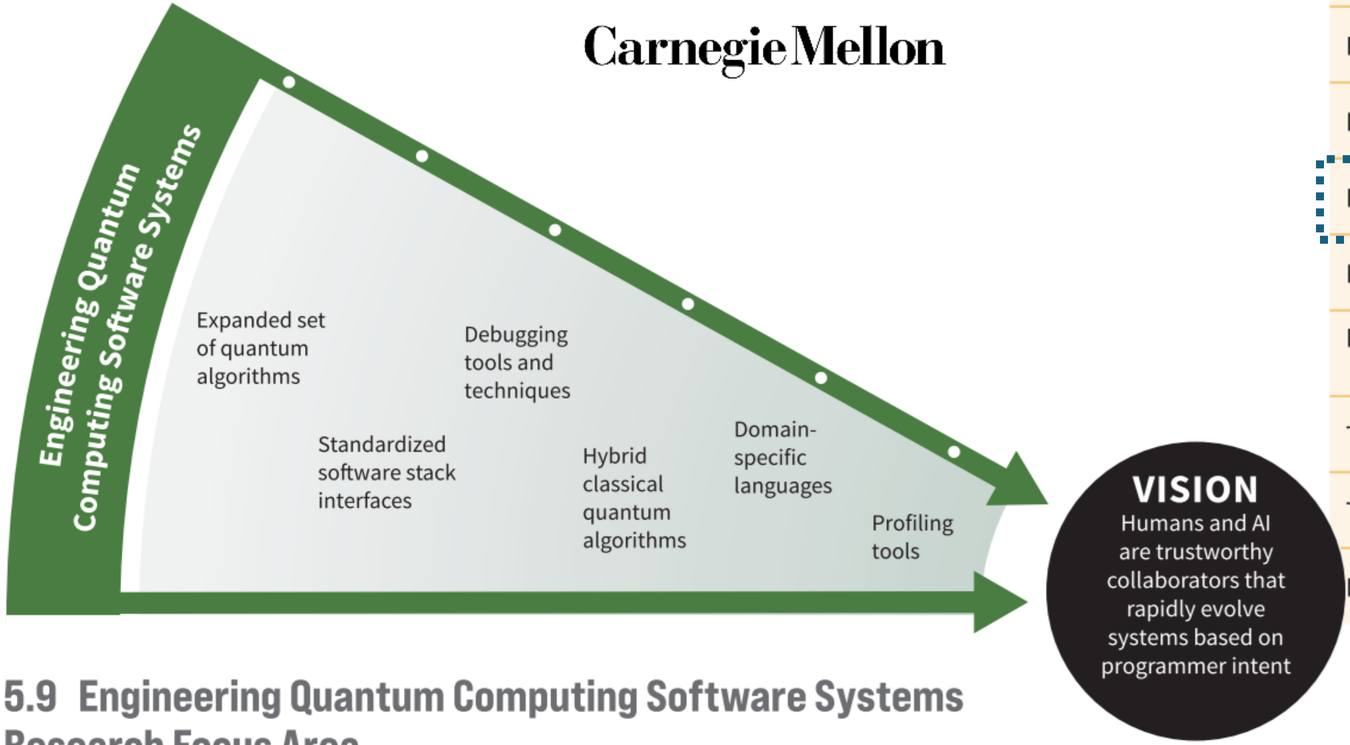
Classical vs Quantum Software Systems

- Classic Information Systems **are still useful** for companies
  - Those who have embedded a lot of mission-critical knowledge
- ***Quantumfy*** every business operation, ~~perhaps~~ **for sure**, **does not make sense**.
  - High cost and no computational gain for simple operations
  - Both computing paradigms must will operate together.

Quantum Software by SEI

Software Engineering Institute

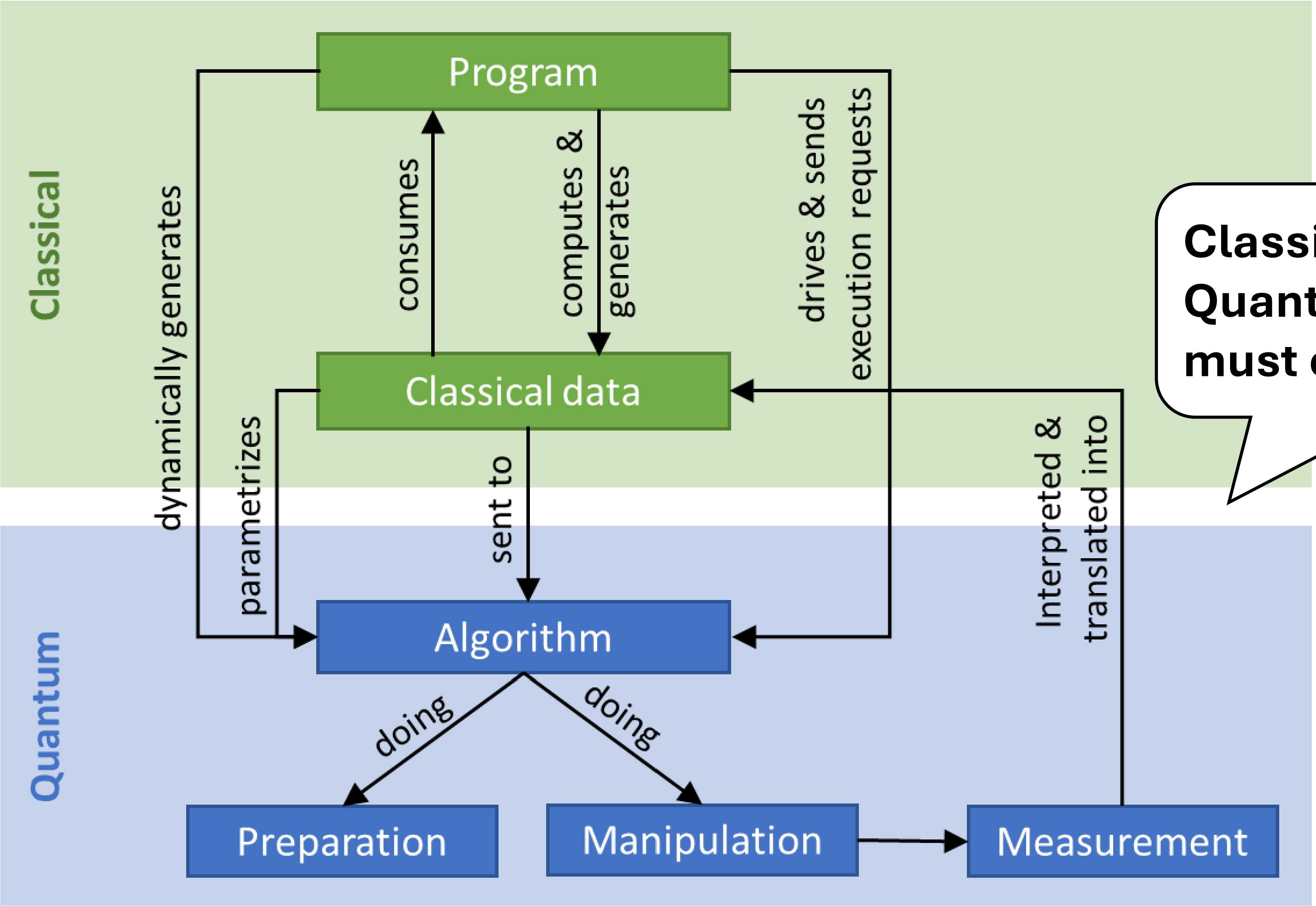
Carnegie Mellon



5.9 Engineering Quantum Computing Software Systems Research Focus Area

Research Direction	Short Term	Mid-term	Long-term
Benchmarks for quantum computing	X		
Large-scale simulation techniques	X		
Expanded set of quantum algorithms	X	X	
Quantum algorithm insights leveraged for classical algorithms	X	X	
Standardized software stack interfaces	X	X	
Intermediate comparisons: QC and simulators	X	X	
Debugging tools and techniques		X	X
Hybrid classical-quantum algorithms		X	X
Proven correct libraries		X	X
Domain-specific languages with greater abstraction		X	X
Target architecture mapping techniques		X	X
Tools for continuous integration		X	X
Profiling tools		X	X

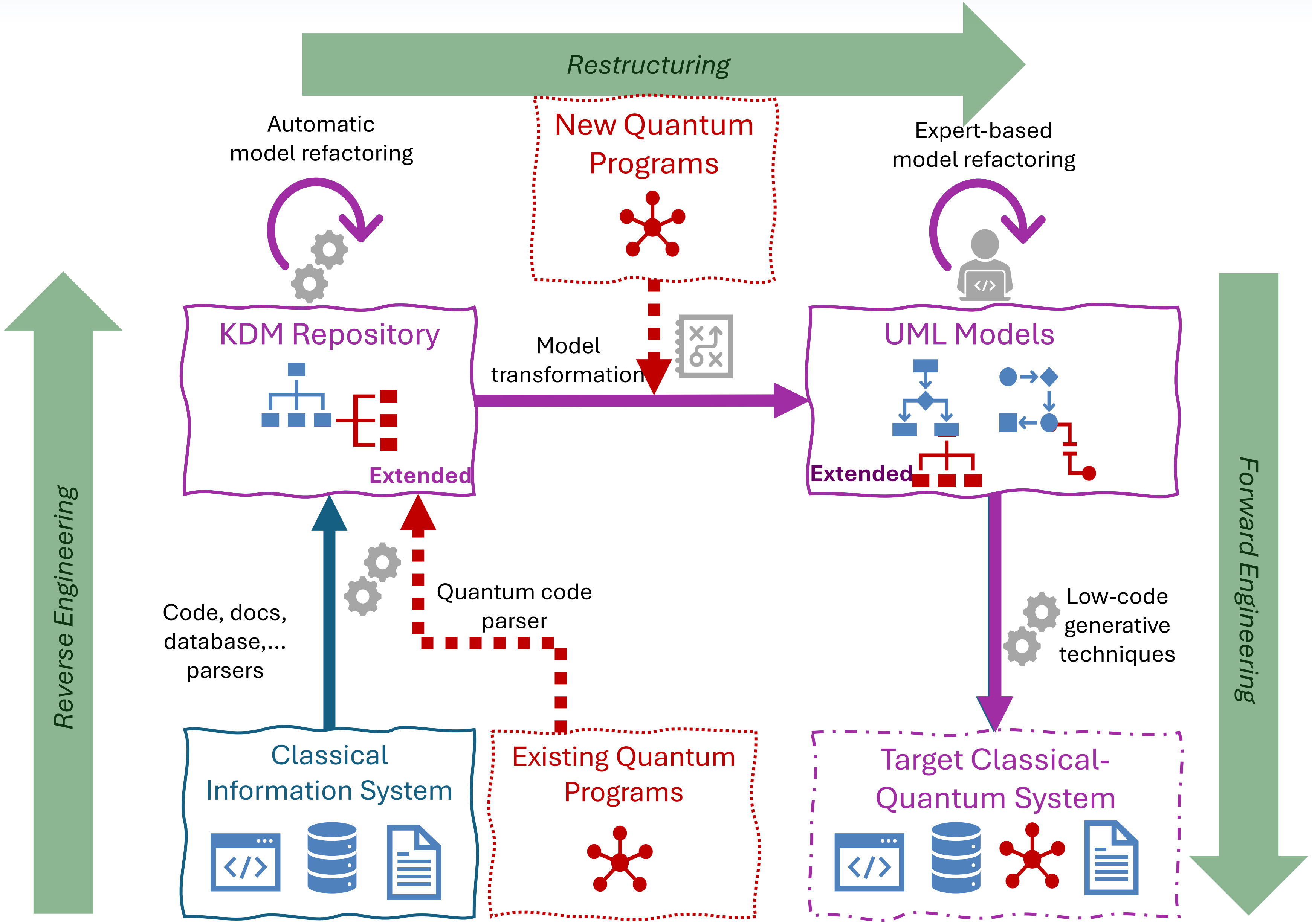
Integration in Hybrid Software Systems



Classical and Quantum Software must operate together

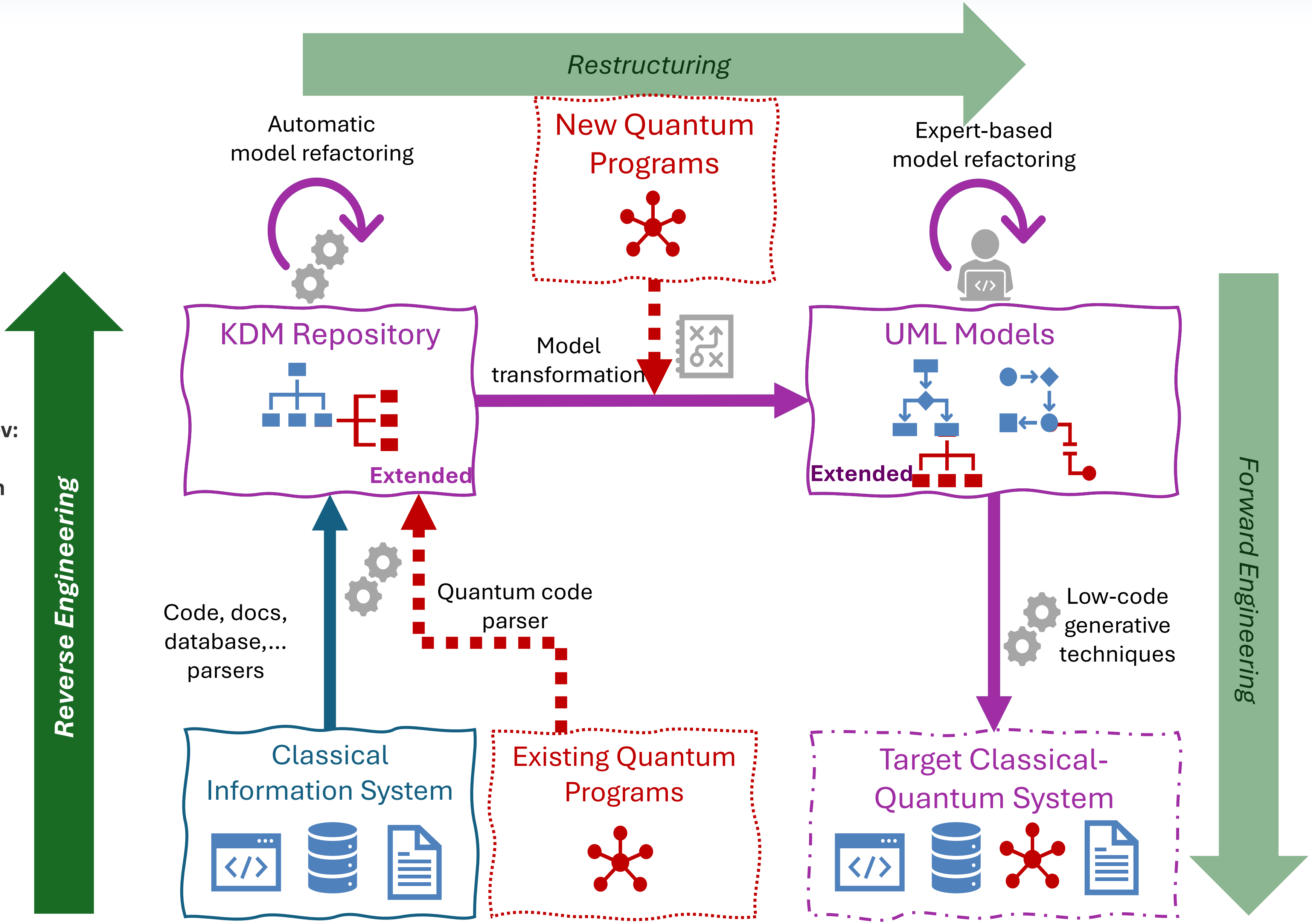
Software Modernization from/to Hybrid Software Systems

Pérez-Castillo, R., M.A. Serrano, and M. Piattini, ***Software modernization to embrace quantum technology***. *Advances in Engineering Software, 2021. 151: p. 102933.*

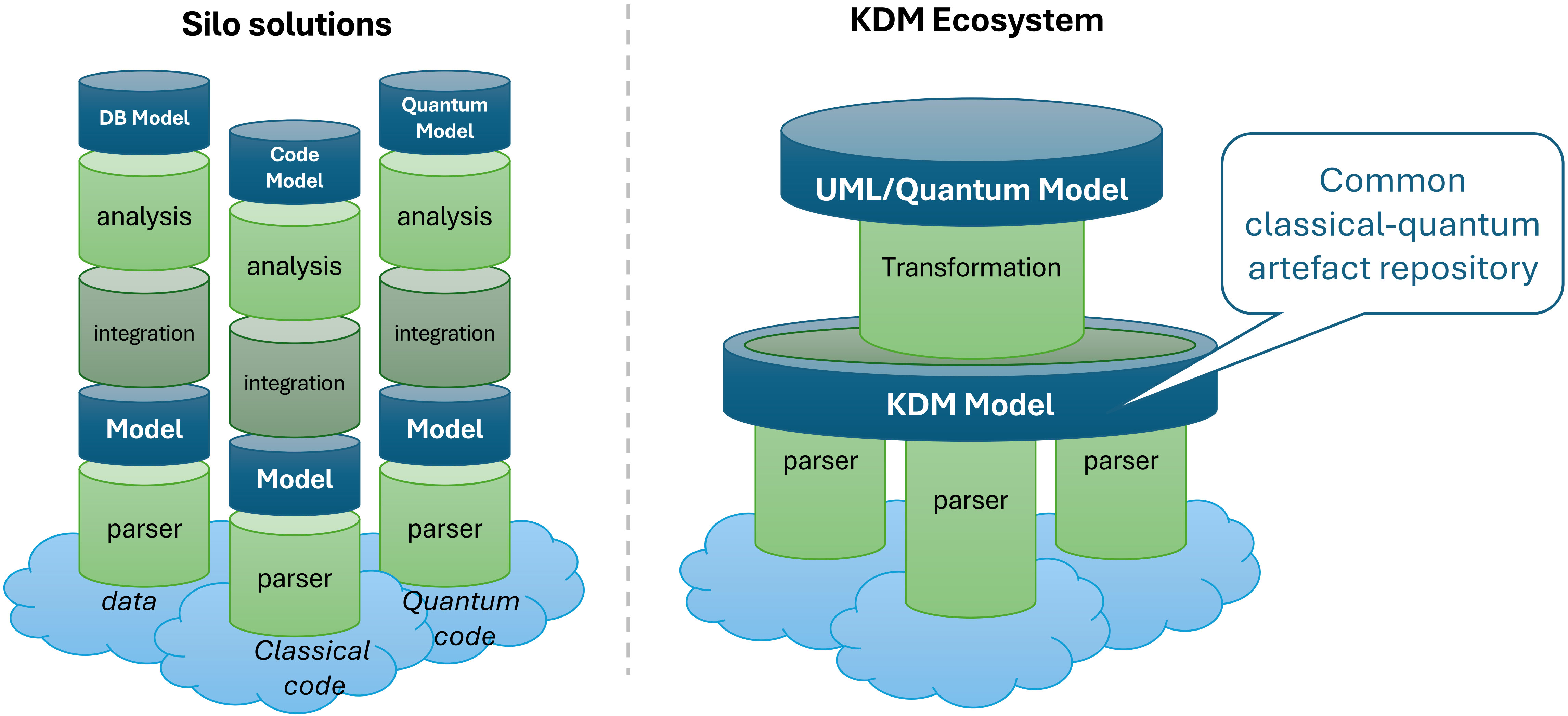


Software Modernization from/to Hybrid Software Systems

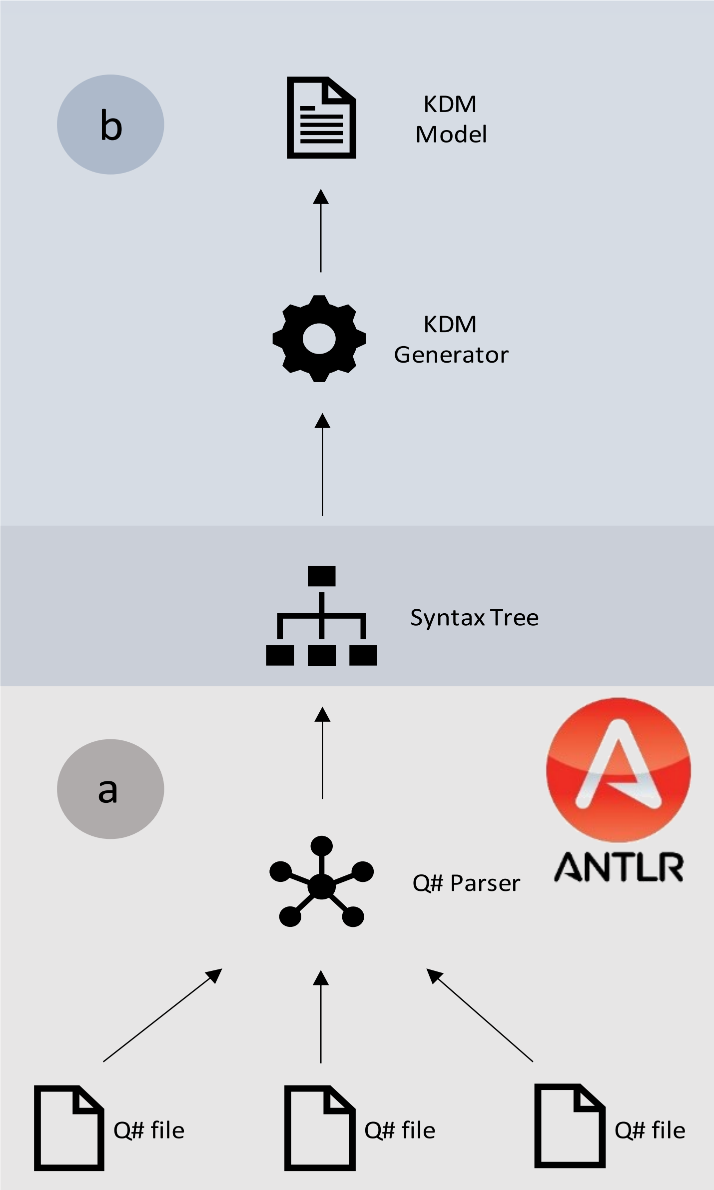
Pérez-Castillo, R., Jiménez-Navajas, L. & Piattini, M. **QRev: migrating quantum code towards hybrid information systems**. *Software Qual J* (2021)

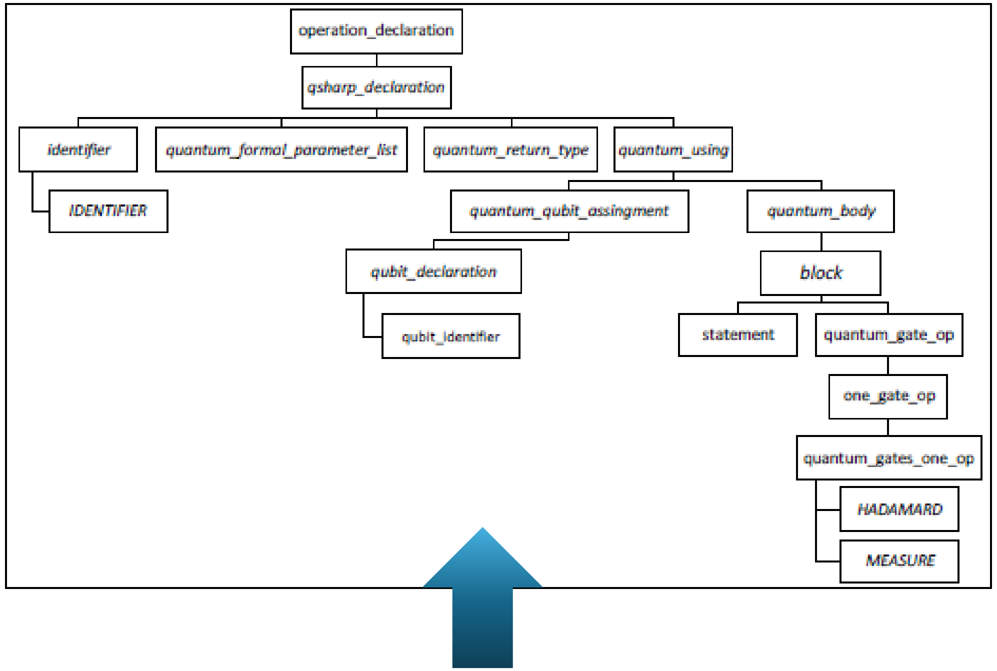


Advantage of KDM Ecosystem



Reverse Engineering: Q# to KDM





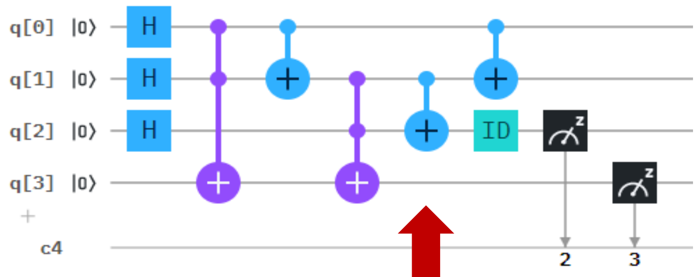

```
namespace Quantum.QSharpApplication1 {
  open Microsoft.Quantum.Canon;
  open Microsoft.Quantum.Intrinsic;

  operation HelloQ () : (Result) {
    using (var qubit = Qubit()){ // Allocate a qubit.
      H(qubit);
      let r = M(qubit);
      Message("Welcome to the quantum world!");
      return r;
    }
  }
}
```

Reverse Engineering: Q# to KDM

```xml
&lt;kdm:Segment xmlns:xmi="http://www.omg.org/XMI" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:action="http://www.omg.org/spec/KDM/20160201/action" xmlns:code="http://www.omg.org/spec/KDM/20160201/code"
 xmlns:kdm="http://www.omg.org/spec/KDM/20160201/kdm" xmi:version="2.0" name="Stereotype Example">
  <extensionFamily name="Example extensions">
    <stereotype name="quantum programming language"/>
    <stereotype name="quantum program">
      <tag tag="implemented in" type=""/>
    </stereotype>
    <stereotype name="quantum gate" type=""/>
    <stereotype name="qubit"/>
    <stereotype name="qubit measure" type=""/>
  </extensionFamily>
```

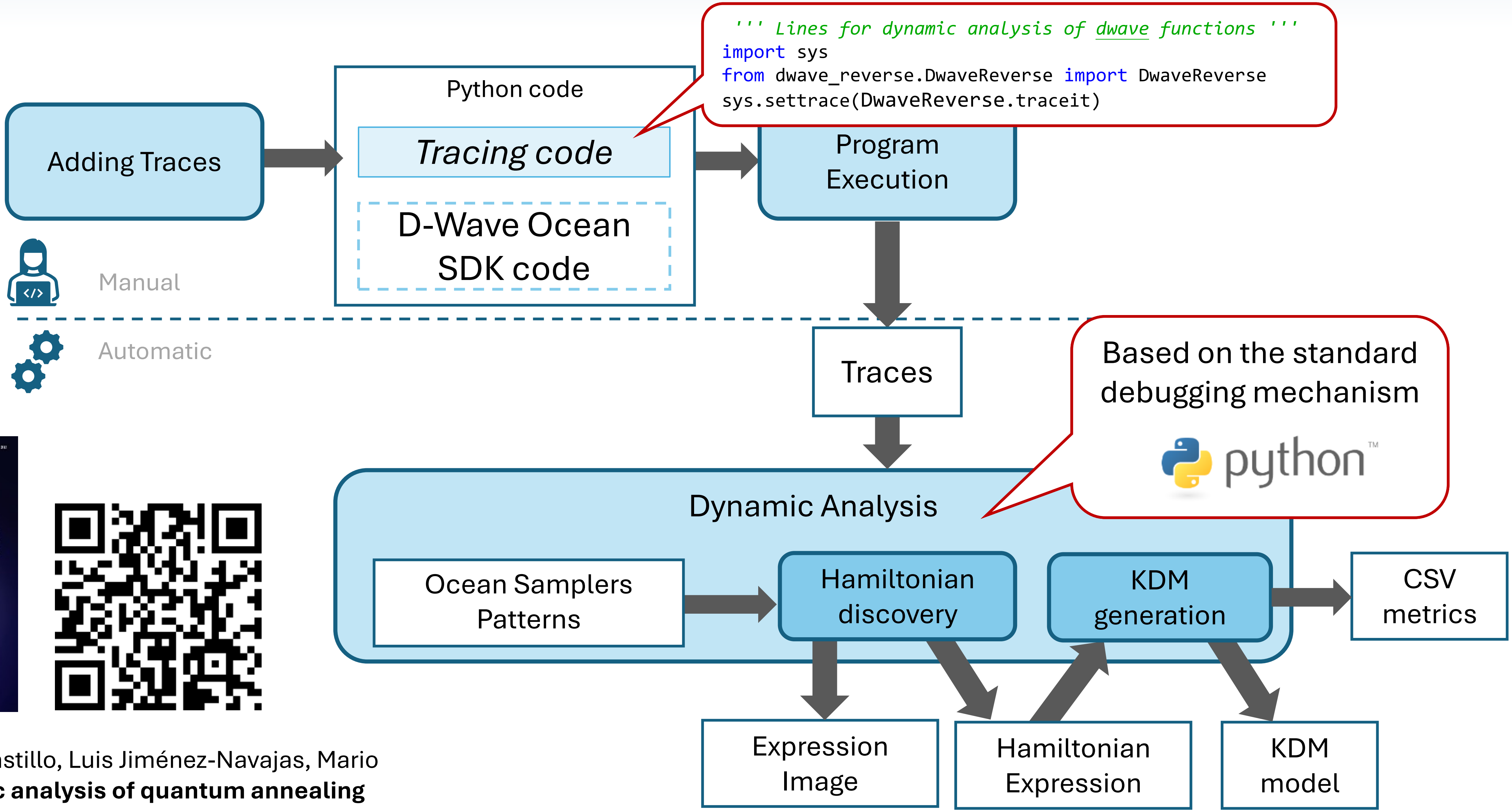
KDM Quantum Extension Family

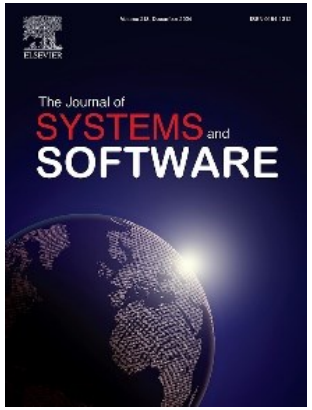
```xml
  <model xsi:type="code:CodeModel" name="Quantum Full Adder">
    <codeElement xsi:type="code:LanguageUnit" stereotype="//@extensionFamily.0/@stereotype.0" name="QASM">
      <codeElement xsi:type="code:DefinedType" stereotype="//@extensionFamily.0/@stereotype.3" name="qubit"/>
      <codeElement xsi:type="code:DefinedType" stereotype="//@extensionFamily.0/@stereotype.3 //@extensionFamily.0/@stereotype.4" name="measured qubit"/>
    </codeElement>
    <codeElement xsi:type="code:CompilationUnit" stereotype="//@extensionFamily.0/@stereotype.1" name="Full Adder">
      <annotation text="This program implements a quantum full adder"/>
      <taggedValue xsi:type="kdm:TaggedValue" tag="//@extensionFamily.0/@stereotype.1/@tag.0" value="QASM"/>
      <entryFlow to="//@model.0/@codeElement.1/@codeElement.4" from="//@model.0/@codeElement.1"/>
      <codeElement xsi:type="code:StorableUnit" stereotype="//@extensionFamily.0/@stereotype.3" name="q[0]" type="//@model.0/@codeElement.0/@codeElement.0" ext="" isStat
      <codeElement xsi:type="code:StorableUnit" stereotype="//@extensionFamily.0/@stereotype.3" name="q[1]" type="//@model.0/@codeElement.0/@codeElement.0" ext="" isStat
      <codeElement xsi:type="code:StorableUnit" stereotype="//@extensionFamily.0/@stereotype.3" name="q[2]" type="//@model.0/@codeElement.0/@codeElement.0" ext="" isStat
      <codeElement xsi:type="code:StorableUnit" stereotype="//@extensionFamily.0/@stereotype.3" name="q[3]" type="//@model.0/@codeElement.0/@codeElement.0" ext="" isStat
      <codeElement xsi:type="action:ActionElement" stereotype="//@extensionFamily.0/@stereotype.2" name="hadamard" kind="">
        <source language="QASM" snippet="h q[0];"/>
        <actionRelation xsi:type="action:Addresses" to="//@model.0/@codeElement.1/@codeElement.0" from="//@model.0/@codeElement.1/@codeElement.4"/>
        <actionRelation xsi:type="action:Flow" to="//@model.0/@codeElement.1/@codeElement.5" from="//@model.0/@codeElement.1/@codeElement.4"/>
      </codeElement>
      <codeElement xsi:type="action:ActionElement" stereotype="//@extensionFamily.0/@stereotype.2" name="hadamard" kind="">
        <source language="QASM" snippet="h q[1];"/>
        <actionRelation xsi:type="action:Addresses" to="//@model.0/@codeElement.1/@codeElement.1" from="//@model.0/@codeElement.1/@codeElement.5"/>
        <actionRelation xsi:type="action:Flow" to="//@model.0/@codeElement.1/@codeElement.6" from="//@model.0/@codeElement.1/@codeElement.5"/>
      </codeElement>
      <codeElement xsi:type="action:ActionElement" stereotype="//@extensionFamily.0/@stereotype.2" name="hadamard" kind="">
        <source language="QASM" snippet="h q[2];"/>
        <actionRelation xsi:type="action:Addresses" to="//@model.0/@codeElement.1/@codeElement.2" from="//@model.0/@codeElement.1/@codeElement.6"/>
        <actionRelation xsi:type="action:Flow" to="//@model.0/@codeElement.1/@codeElement.7" from="//@model.0/@codeElement.1/@codeElement.6"/>
      </codeElement>
      <codeElement xsi:type="action:ActionElement" stereotype="//@extensionFamily.0/@stereotype.2" name="toffoli" kind="">
        <source language="QASM" snippet="ccx q[0],q[1],q[3];"/>
        <actionRelation xsi:type="action:Reads" to="//@model.0/@codeElement.1/@codeElement.0" from="//@model.0/@codeElement.1/@codeElement.7"/>
        <actionRelation xsi:type="action:Reads" to="//@model.0/@codeElement.1/@codeElement.1" from="//@model.0/@codeElement.1/@codeElement.7"/>
        <actionRelation xsi:type="action:Addresses" to="//@model.0/@codeElement.1/@codeElement.3" from="//@model.0/@codeElement.1/@codeElement.7"/>
        <actionRelation xsi:type="action:Flow" to="//@model.0/@codeElement.1/@codeElement.8" from="//@model.0/@codeElement.1/@codeElement.7"/>
      </codeElement>
      <codeElement xsi:type="action:ActionElement" stereotype="//@extensionFamily.0/@stereotype.2" name="cnot" kind="">
        <source language="QASM" snippet="cx q[0],q[1];"/>
        <actionRelation xsi:type="action:Reads" to="//@model.0/@codeElement.1/@codeElement.0" from="//@model.0/@codeElement.1/@codeElement.8"/>
        <actionRelation xsi:type="action:Addresses" to="//@model.0/@codeElement.1/@codeElement.1" from="//@model.0/@codeElement.1/@codeElement.8"/>
        <actionRelation xsi:type="action:Flow" to="//@model.0/@codeElement.1/@codeElement.9" from="//@model.0/@codeElement.1/@codeElement.8"/>
      </codeElement>

      ...
  </model>
&lt;/kdm:Segment&gt;
```




```
OPENQASM 2.0;
include "qelib1.inc";
gate nG0 ( param ) q  {
  h q;
}
qreg q[4];
creg c[4];
h q[0];
h q[1];
h q[2];
ccx q[0],q[1],q[3];
cx q[0],q[1];
ccx q[1],q[2],q[3];
cx q[1],q[2];
cx q[0],q[1];
id q[2];
measure q[2] -> c[2];
measure q[3] -> c[3];
```

Dynamic Analysis for reversing Quantum Annealing Programs





Ricardo Pérez-Castillo, Luis Jiménez-Navajas, Mario Piattini: **Dynamic analysis of quantum annealing programs.** J. Syst. Softw. 201: 111683 (2023)

Our Research Journey

2020

- Talavera Manifesto
- Quantum Software Modernization Framework
- Reverse Engineering of Quantum Software to KDM

2021

- Reverse Engineering of Quantum Annealing programs to KDM

2022

- UML extension for Quantum Software
- KDM-to-UML transformation

2023

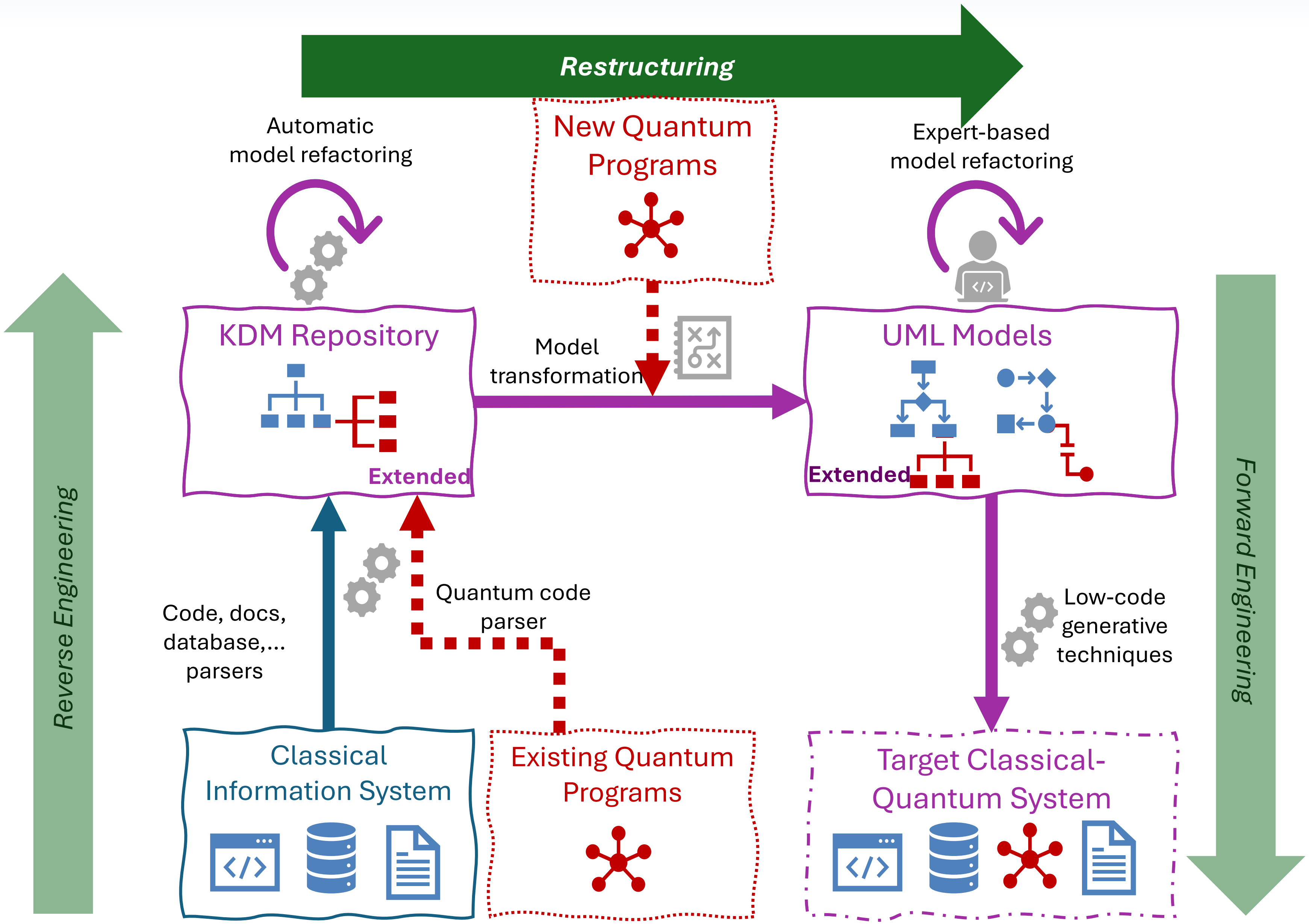
- Survey on Quantum Software Components and Platforms
- Reverse Engineering of OpenQASM3 Quantum Programs

2024

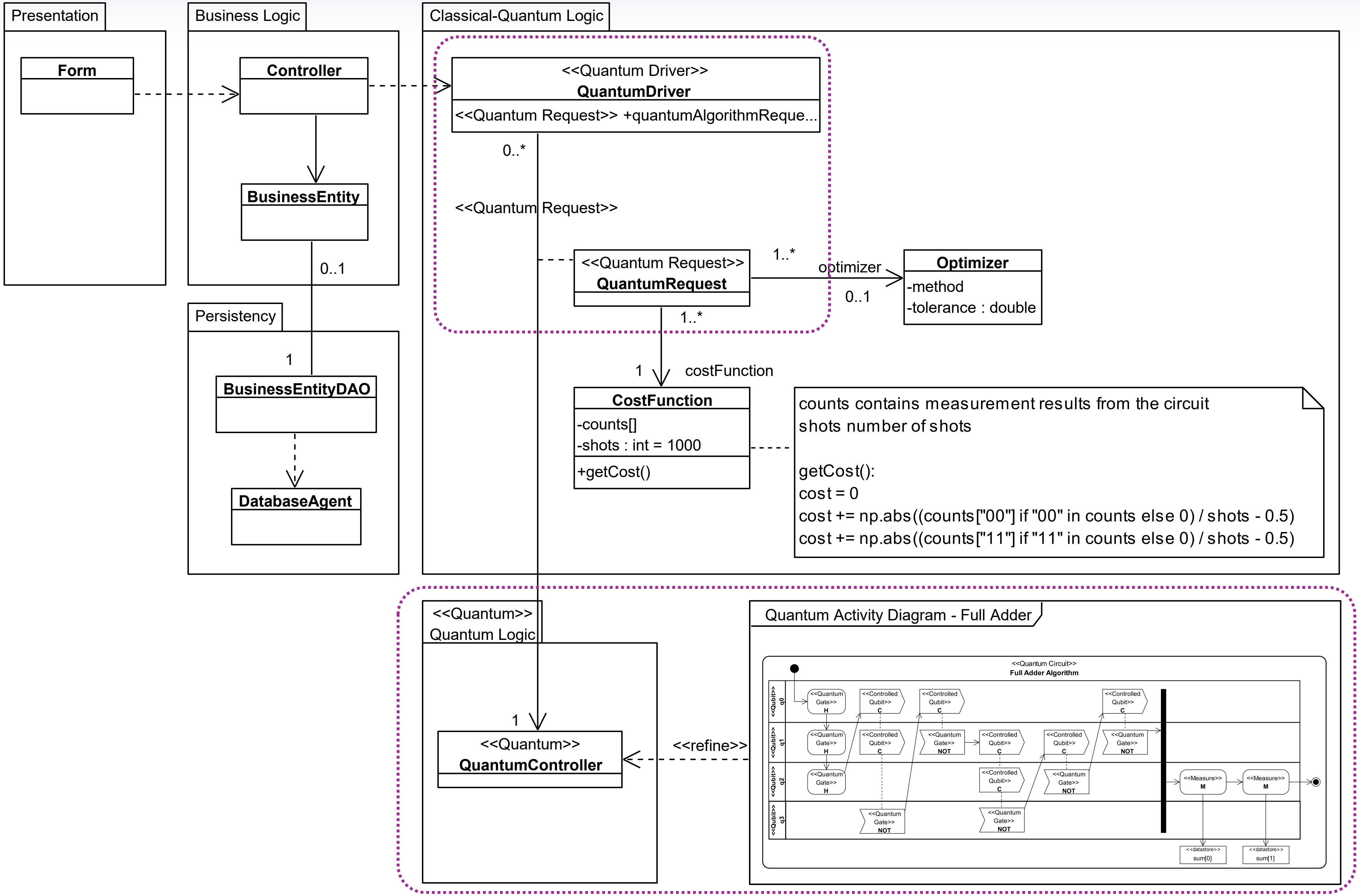
2025

…

Software Modernization from/to Hybrid Software Systems

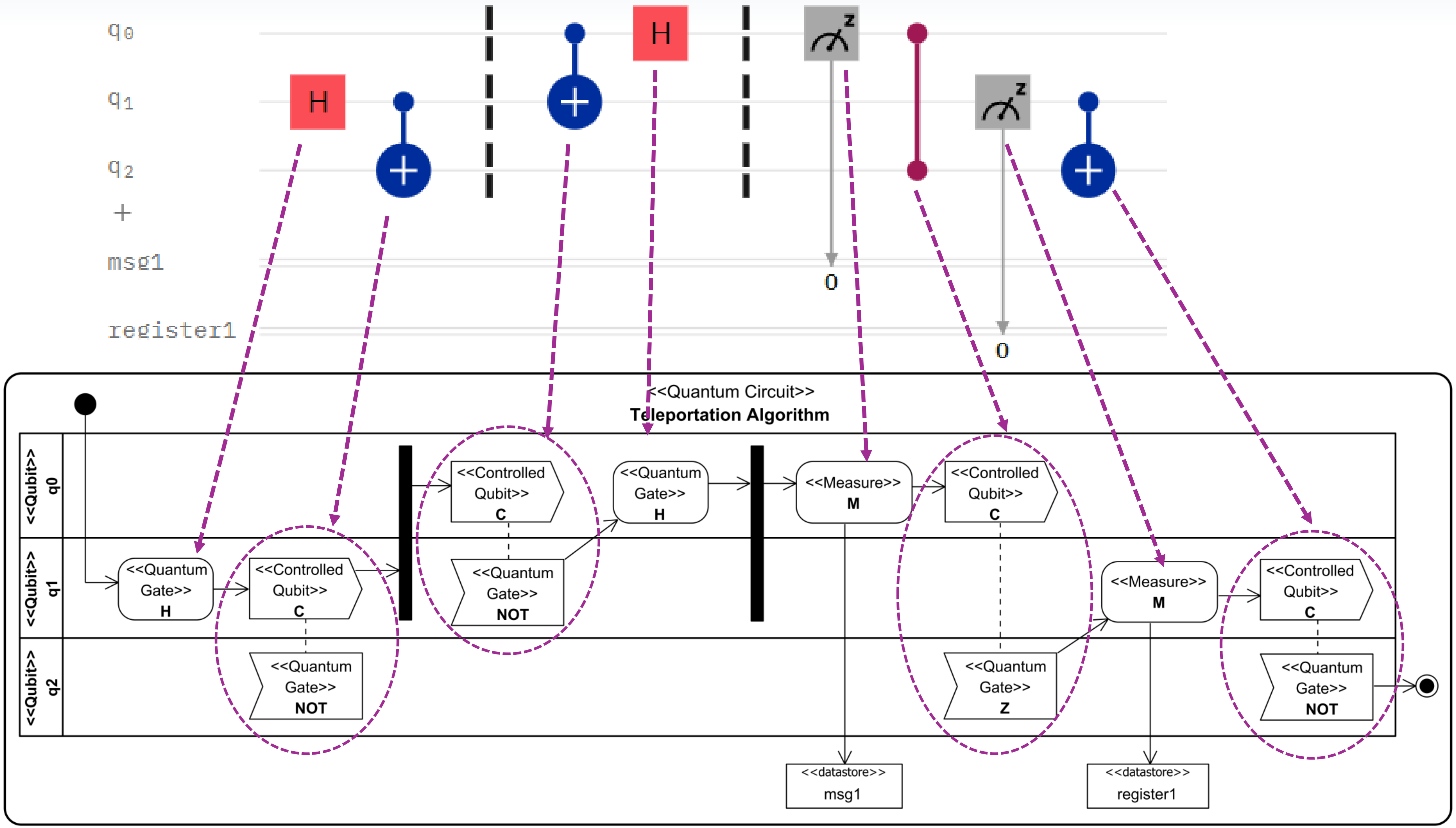


Quantum UML Profile

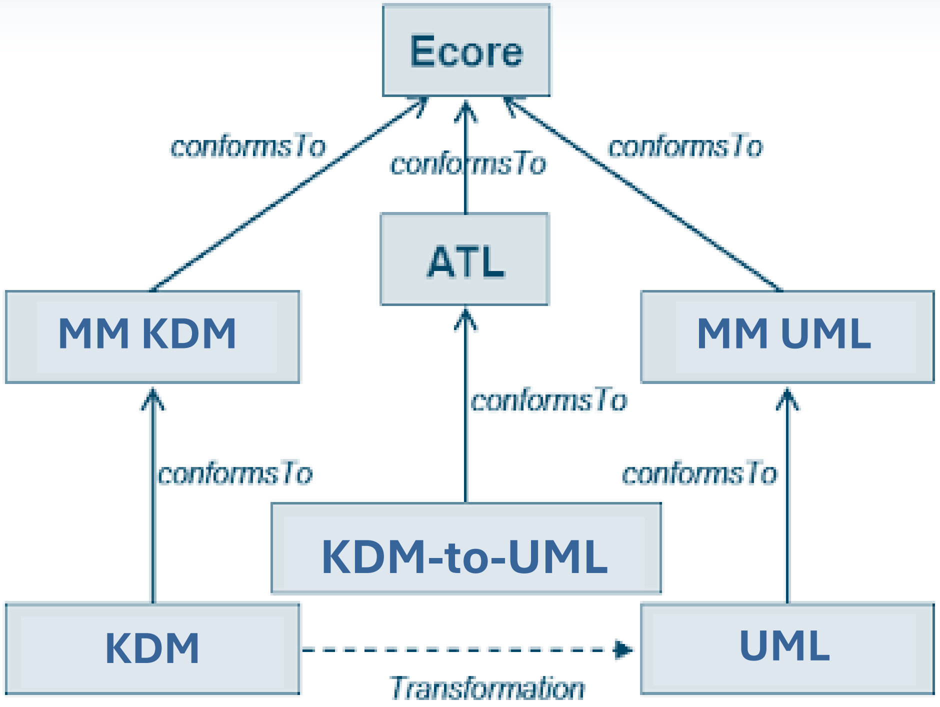


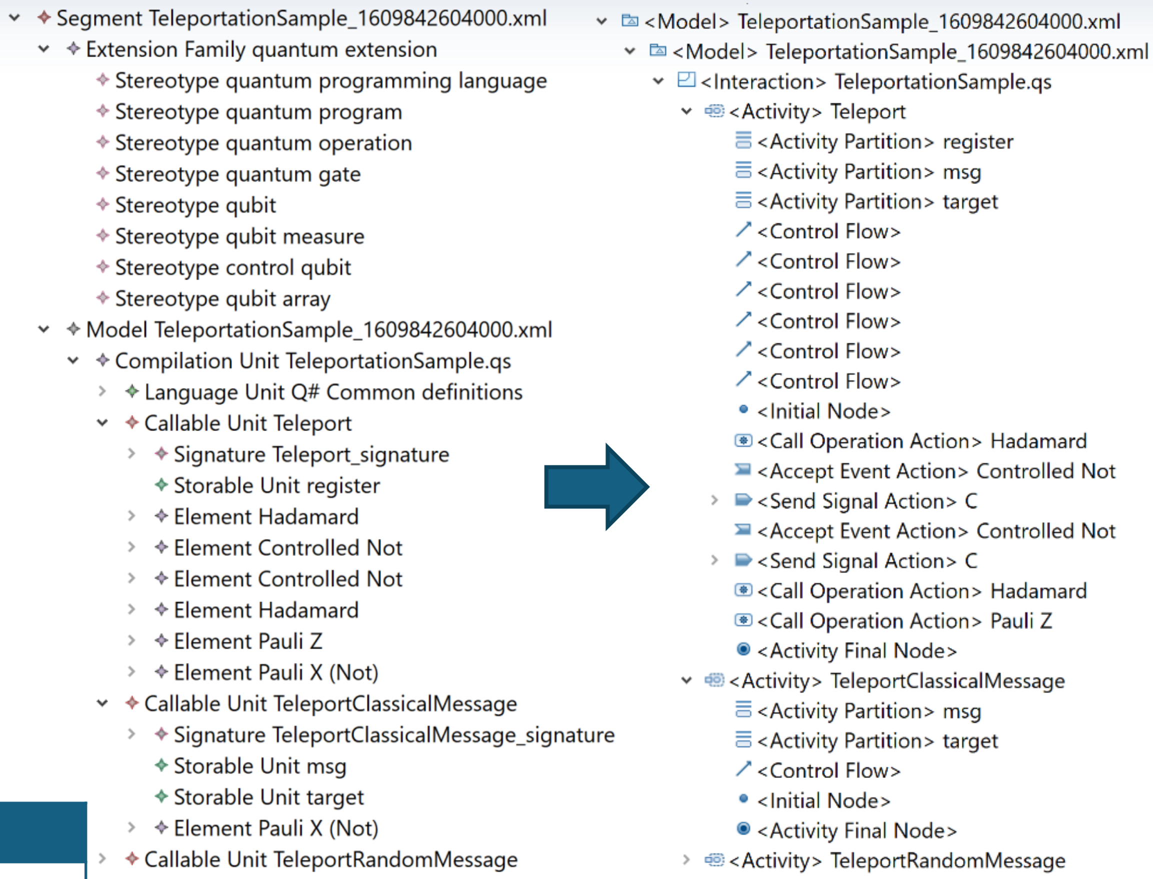
Ricardo Pérez-Castillo, Mario Piattini: ***Design of classical-quantum systems with UML****. Computing 104(11): 2375-2403 (2022)*

Restructuring: Quantum UML profile

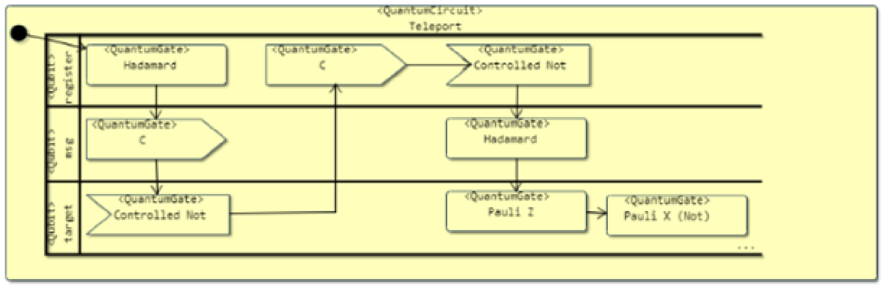


Restructuring: Transformación KDM a UML





Q# element	Input KDM	Output UML
Quantum program	CompilationUnit	Interaction
Quantum operation	CallableUnit	Activity
Qubit declaration	StorableUnit	ActivityPartition
Quantum gate	ActionElement	CallOperationAction/ AcceptEventAction/ SendSignalAction
Data flow between gates	Flow	ControlFlow



Our Research Journey

2020
- Talavera Manifesto
- Quantum Software Modernization Framework
- Reverse Engineering of Quantum Software to KDM

2021
- Reverse Engineering of Quantum Annealing programs to KDM

2022
- UML extension for Quantum Software
- KDM-to-UML transformation

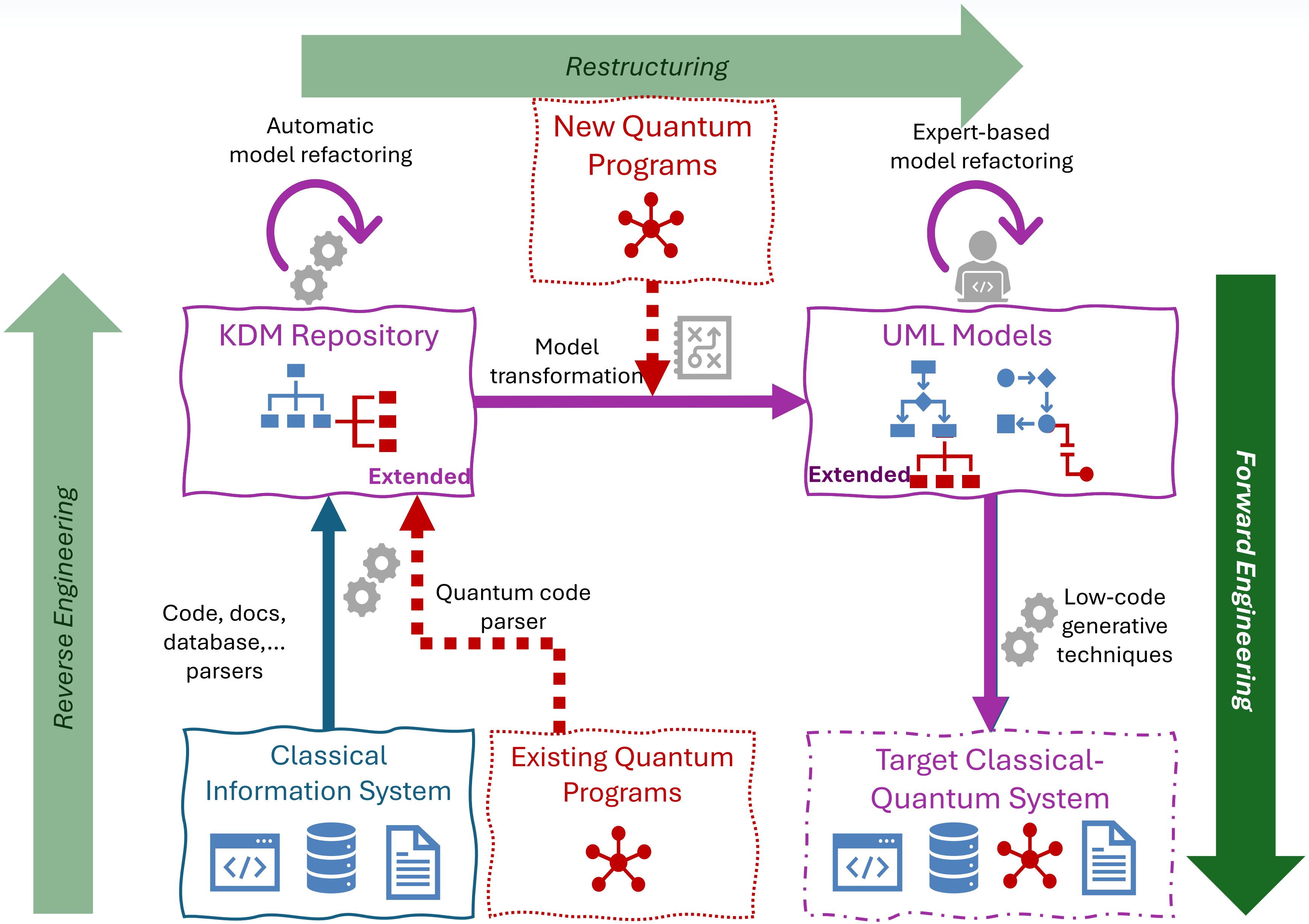
2023
- Survey on Quantum Software Components and Platforms
- Reverse Engineering of OpenQASM3 Quantum Programs

2024
- Hybrid code generation from UML models
- ICSM lifecycle for developing hybrid software systems

2025
- Quantum Software Design Patterns
- Quantum Software Product Lines

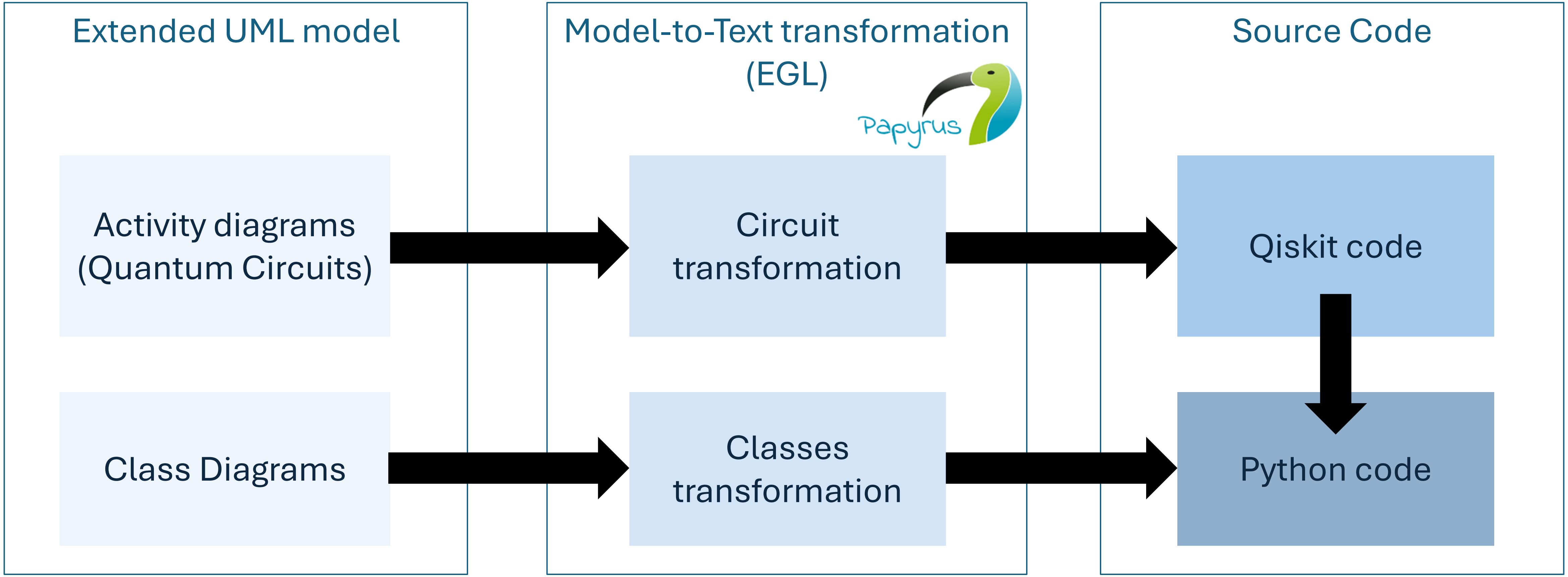
…

Software Modernization from/to Hybrid Software Systems



Hybrid code generation from UML

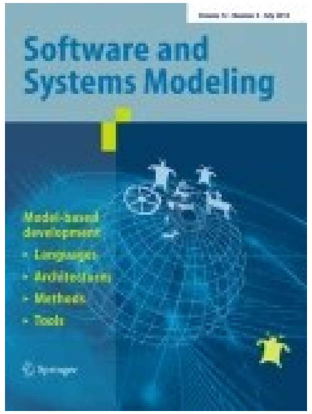
- Model-to-Text transformation with EGL (Eclipse Generation Language)



Hybrid code generation from UML

- Excerpt of EGL transformation for quantum gates generation in Qiskit

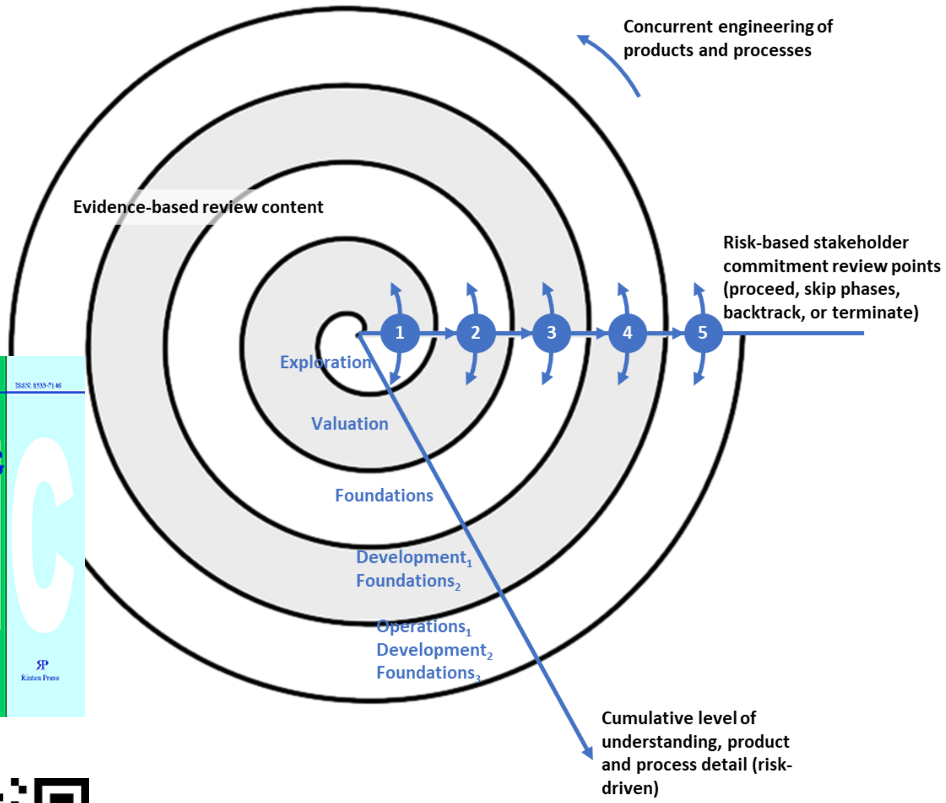
```
[*QUANTUM GATES*]
    [% var initial = getInitialNode();
       if(initial == null){
           "There is no InitialNode".println();
       } else{
           var actual = initial.outgoings.target.get(0);
           var type = actual.toString().split(' ').get(0).split('@').get(0);
           while(not(type == "org.eclipse.uml2.uml.internal.impl.ActivityFinalNodeImpl")){
               var gate = checkGateType(actual);
               if (not (gate == "pass")){%]
                   [%=circuitName%].[%=gate%]
               [%}
               actual = nextNode(actual);
               type = actual.toString().split(' ').get(0).split('@').get(0);
           }
       }%]
```

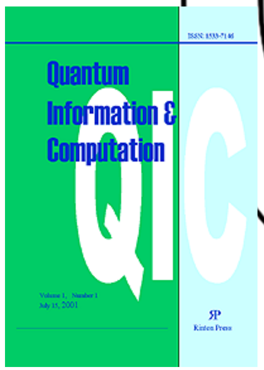


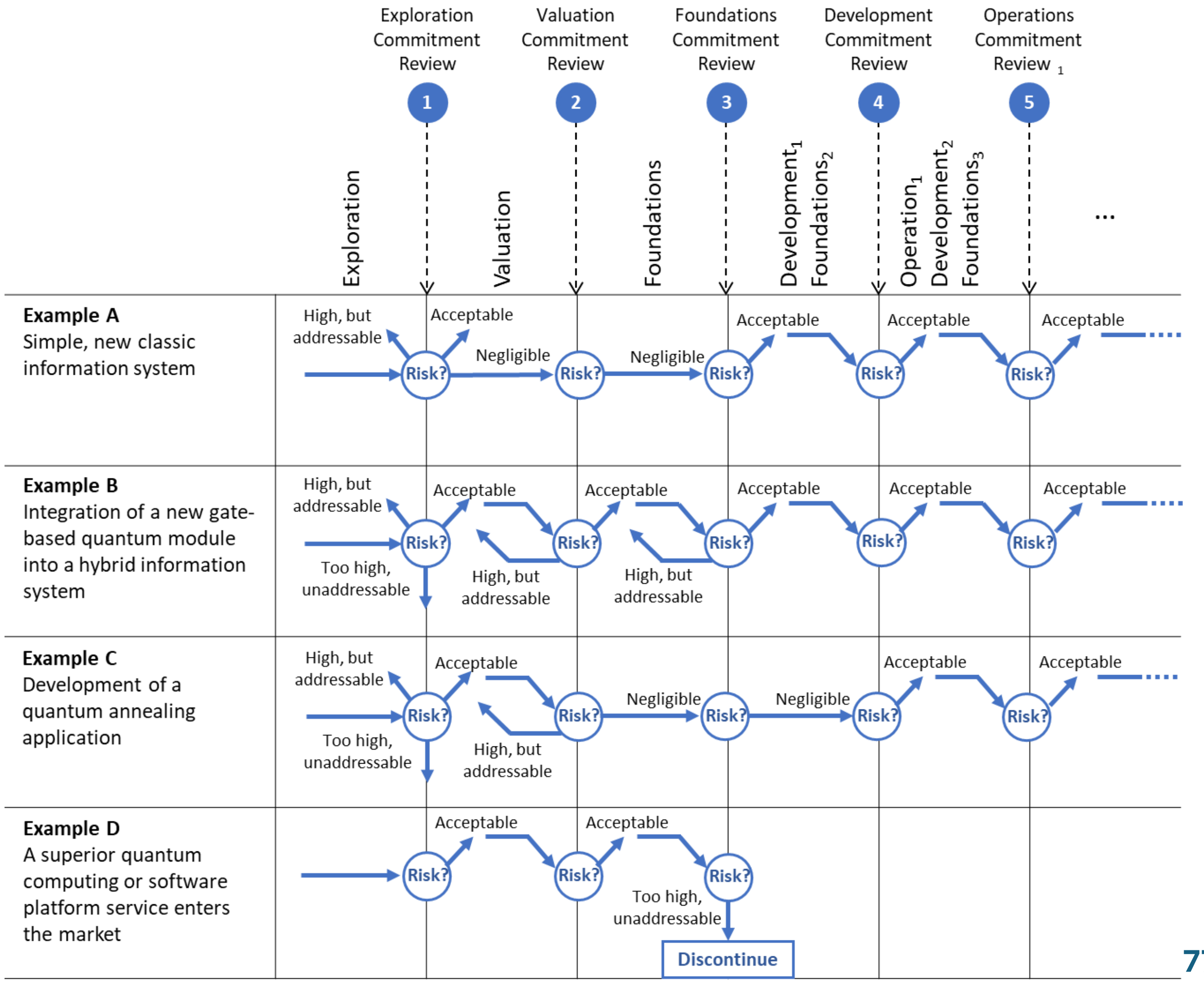
Luis Jiménez-Navajas, Ricardo Pérez-Castillo, Mario Piattini. ***Code generation for classical-quantum software systems modeled in UML.*** *Softw. Syst. Model. 24(3): 795-821 (2025)*

ICSM Life Cycle for Hybrid Software

- ICSM (Incremental Commitment Spiral Model) as a Lifecycle

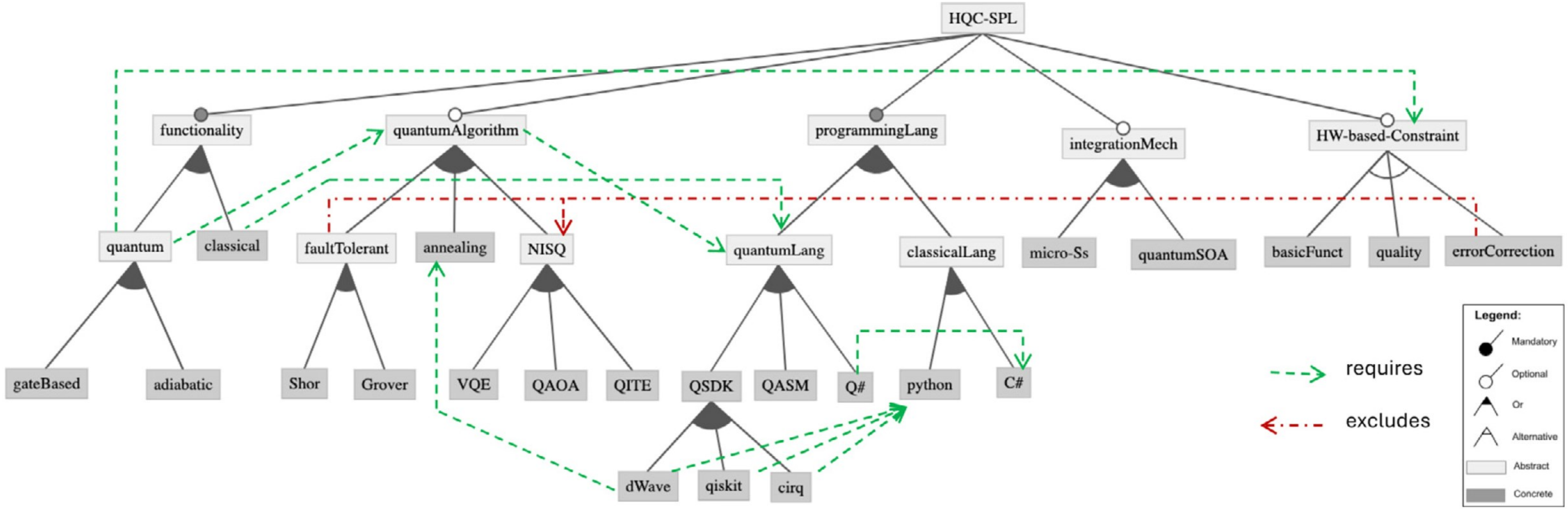


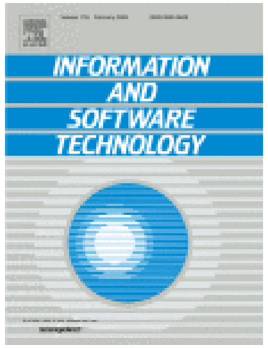




Ricardo Pérez-Castillo, Manuel Ángel Serrano, José A. Cruz-Lemus, Mario Piattini: **Guidelines to use the ICSM for developing quantum-classical systems**. Quantum Inf. Comput. 24(1&2): 71-88 (2024)

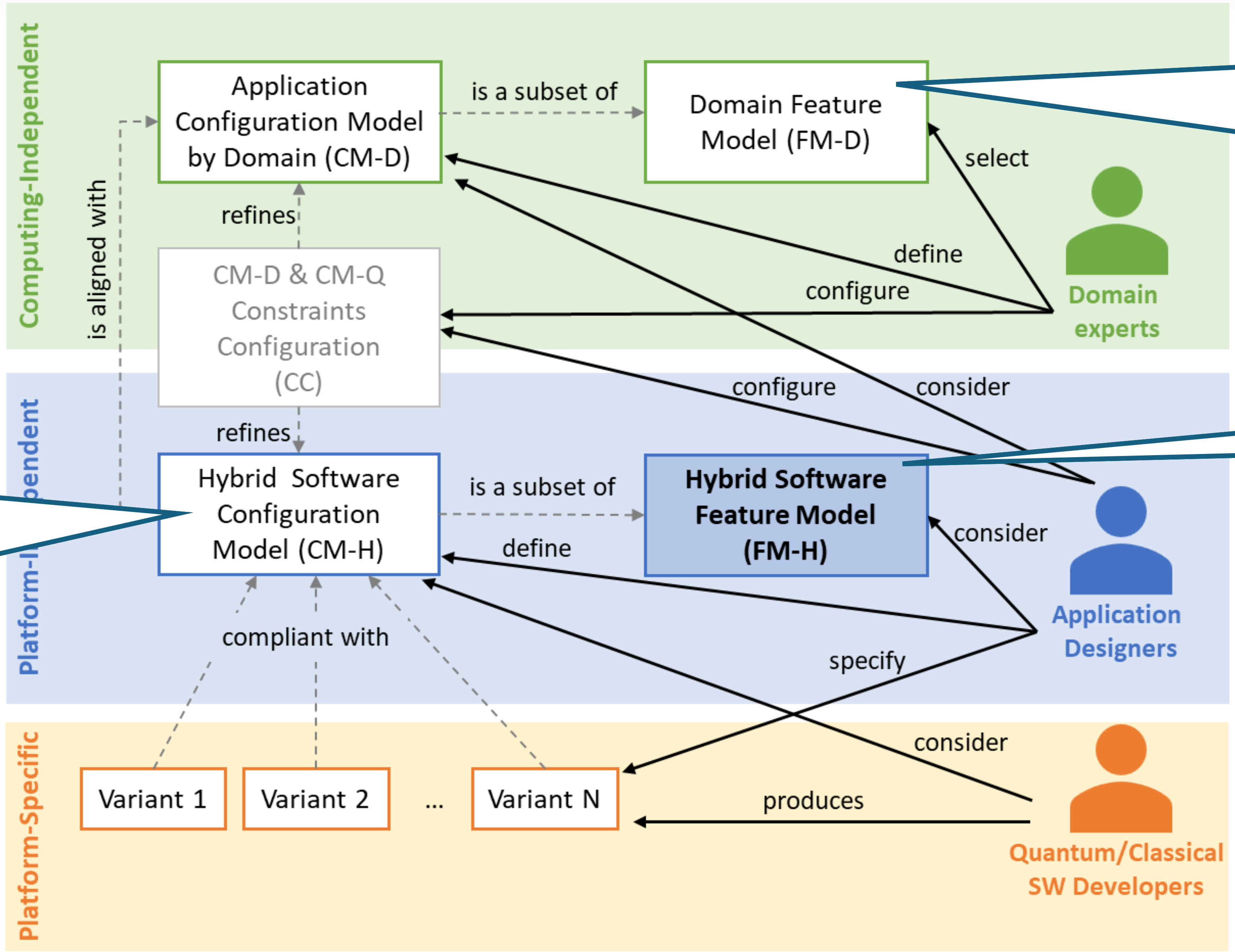
Software Product Lines for Hybrid Software





Samuel Sepúlveda, Ricardo Pérez-Castillo, Mario Piattini. **A software product line approach for developing hybrid software systems.** Inf. Softw. Technol. 178: 107625 (2025)

Software Product Lines for Hybrid Software

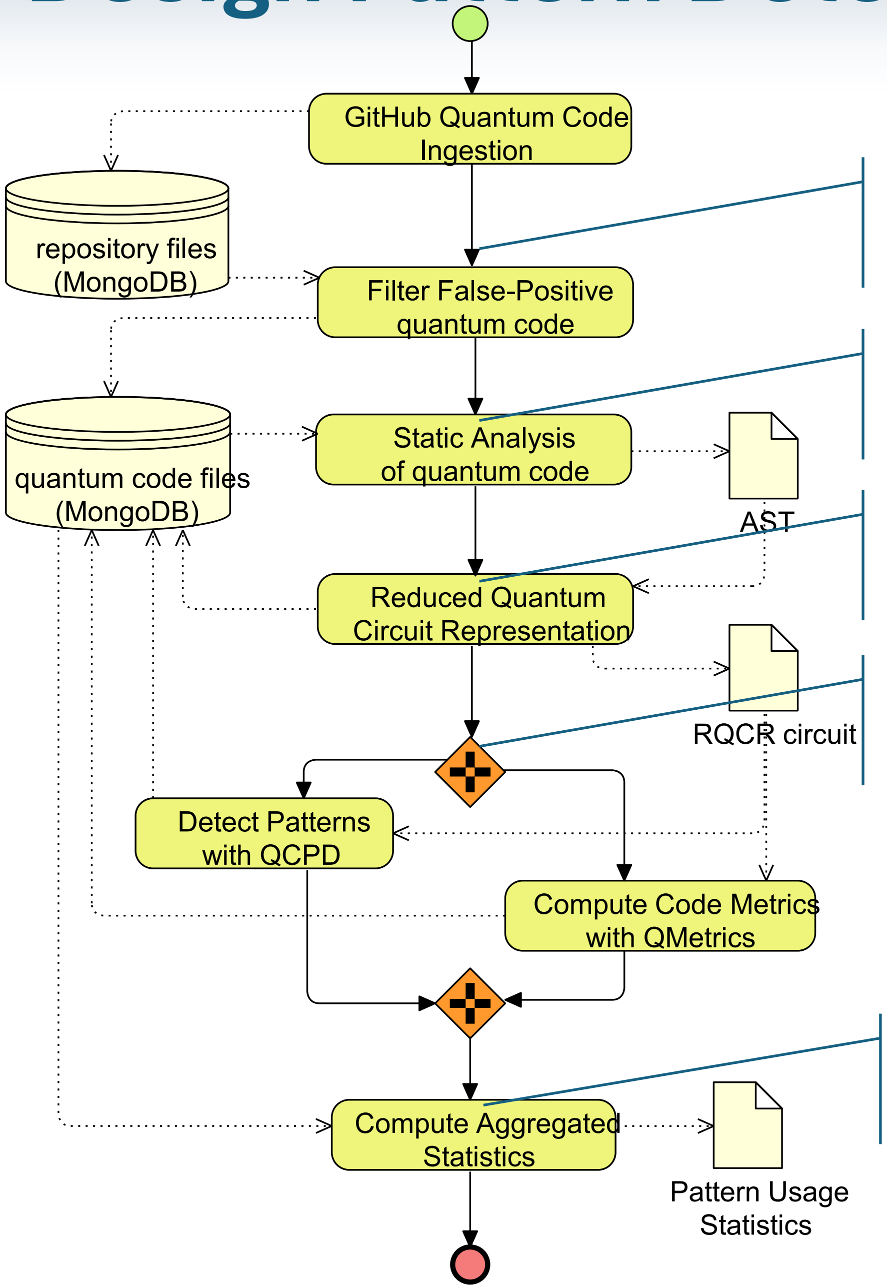


There are already many existing FM-Ds. Domain experts can use existing FM-Ds or define their own ones.

Our proposal

The proposed FM for hybrid software systems (FM-H) can be applied to different domains through the FM-D.

Design Pattern Detection on Quantum Programs

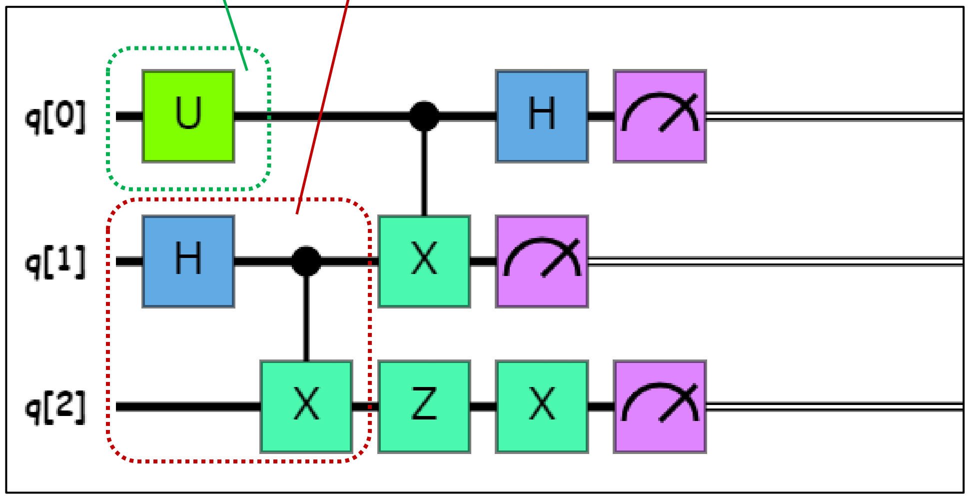


Qiskit	QASM
13462	75800
5004	15320
1616	101
806	101
30	50

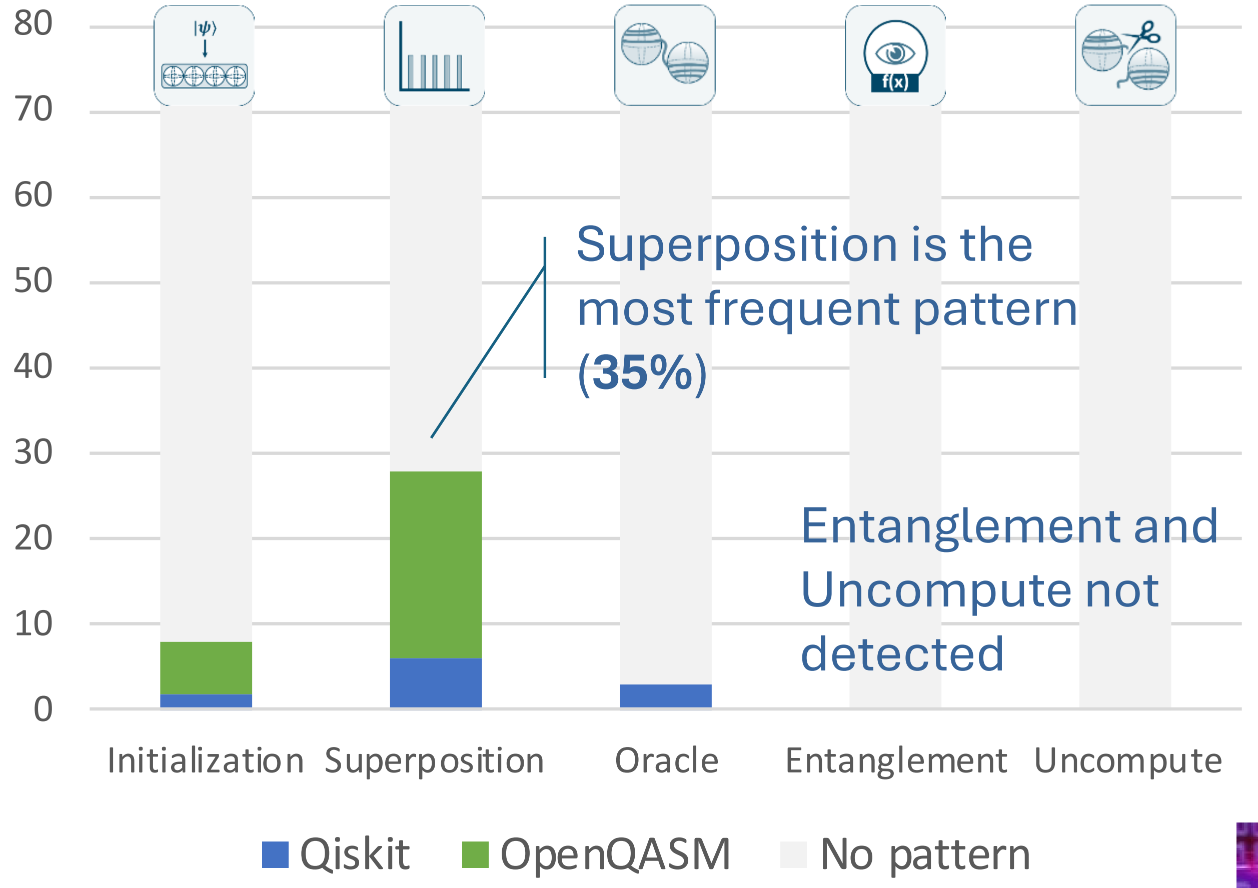
GitHub Qiskit

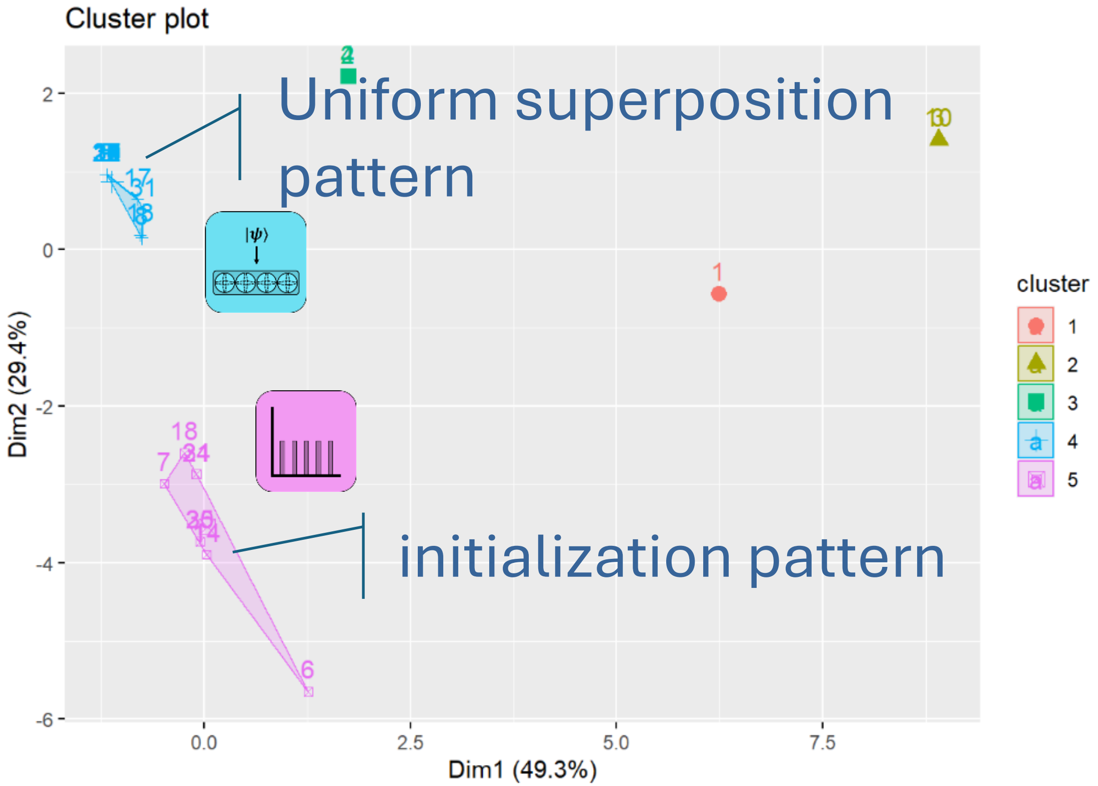
Oracle pattern detected

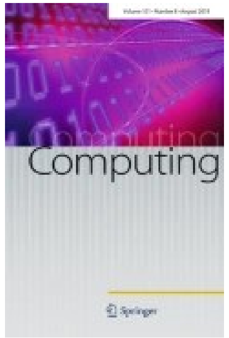
Entanglement pattern not detected



Design Pattern Detection on Quantum Programs







Miriam Fernández-Osuna, Ricardo Pérez-Castillo, José A. Cruz-Lemus, Michal Baczyk, Mario Piattini. **Exploring design patterns in quantum software: a case study.** Computing 107(5): 111 (2025)

Our Research Journey

2020
- Talavera Manifesto
- Quantum Software Modernization Framework
- Reverse Engineering of Quantum Software to KDM

2021
- Reverse Engineering of Quantum Annealing programs to KDM

2022
- UML extension for Quantum Software
- KDM-to-UML transformation

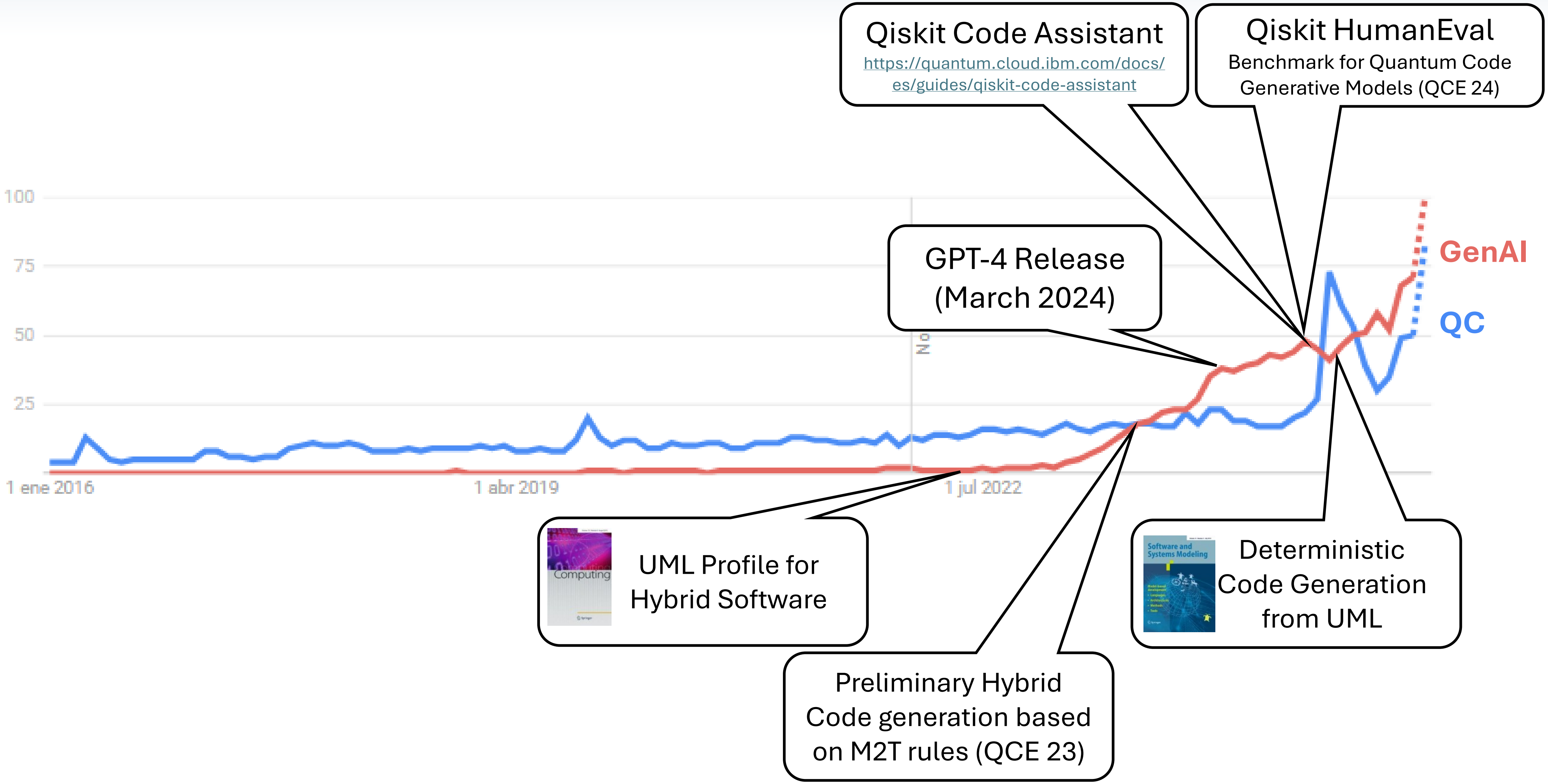
2023
- Survey on Quantum Software Components and Platforms
- Reverse Engineering of OpenQASM3 Quantum Programs

2024
- Hybrid code generation from UML models
- ICSM lifecycle for developing hybrid software systems

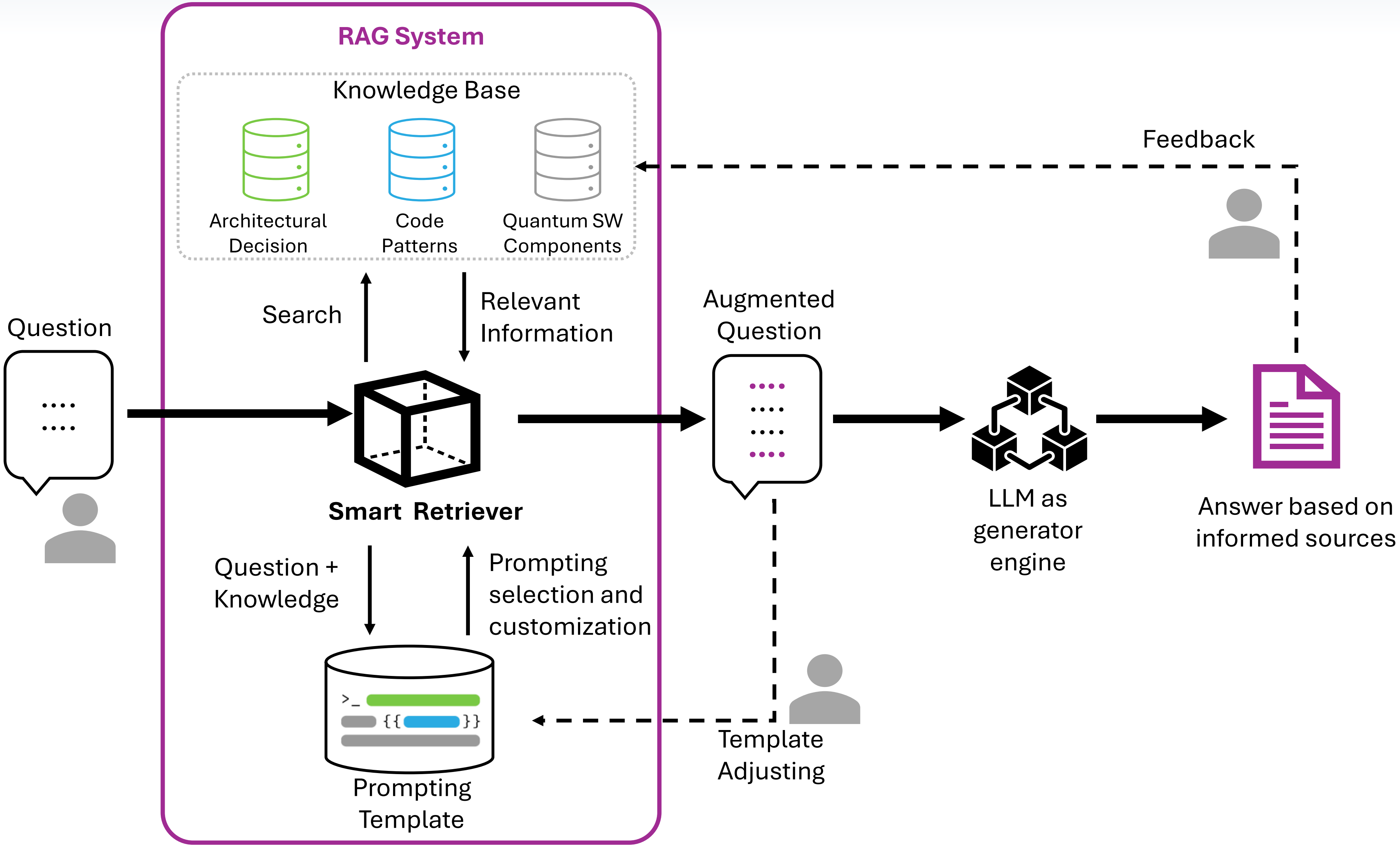
2025
- Quantum Software Design Patterns
- Quantum Software Product Lines

...
- AI for Hybrid Software Architectures
- DSPL for self-adaptive hybrid systems

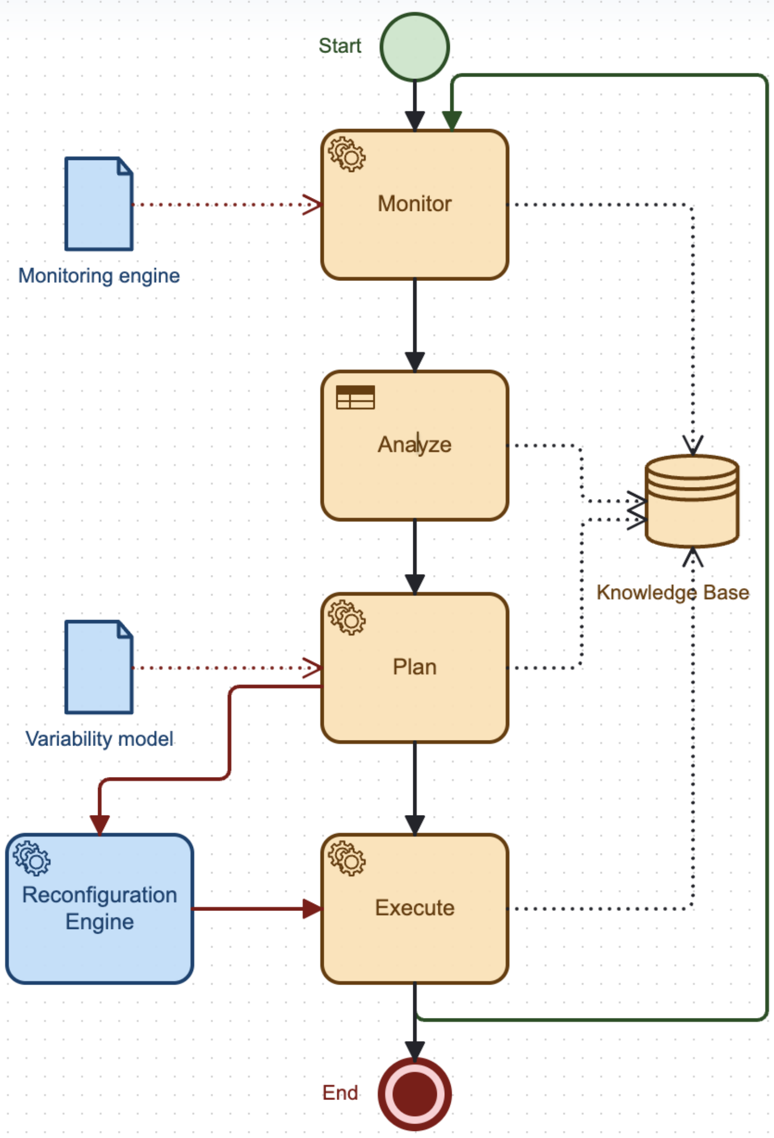
GenAI is a game changer?



RAG for hybrid software architecting



Dynamic SPL for Self-Adaptation



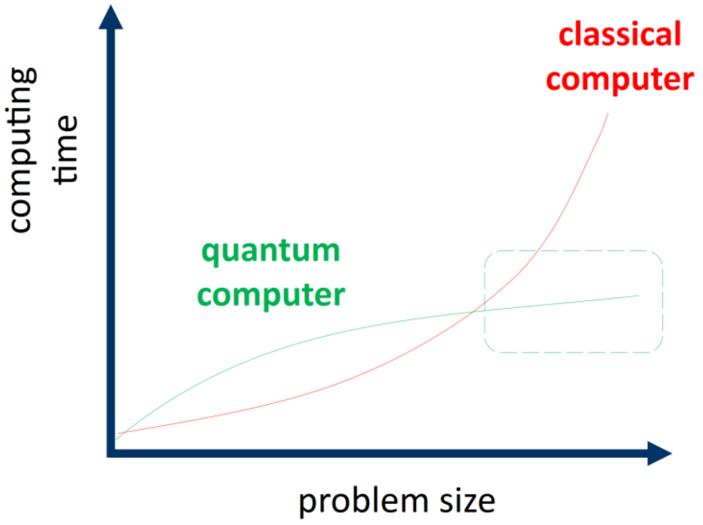
- **DSPL Integration with MAPE-K**
  - Integration of Dynamic Software Product Lines (DSPL) with the MAPE-K feedback loop to support self-adaptation.
- **Run-time Variability Model**
  - Explicit representation of variability at run-time to enable context-aware reconfiguration.
- **Monitoring Engine**
  - Collects data from the running system and its environment to detect changes and trigger adaptations.
- **Reconfiguration Engine**
  - Applies adaptation decisions by enacting the selected system configuration.

Agenda

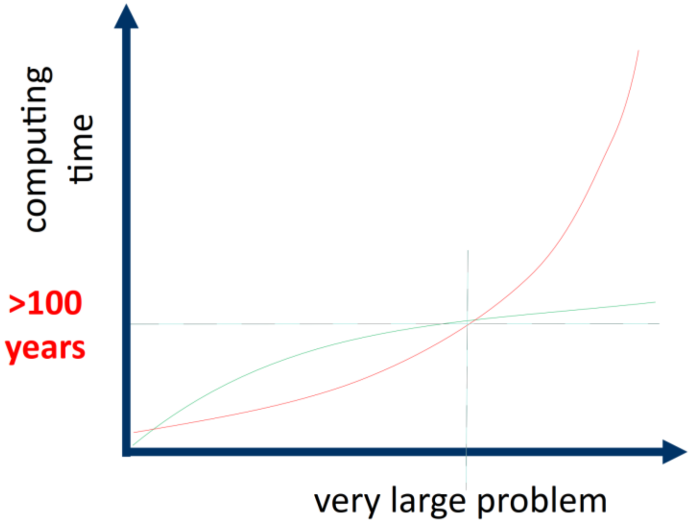
- Quantum Computing
- Quantum Computing Applications
- Quantum Software & Quantum Software Engineering
- Our Research Journey
- **Conclusions**

Scan for slides

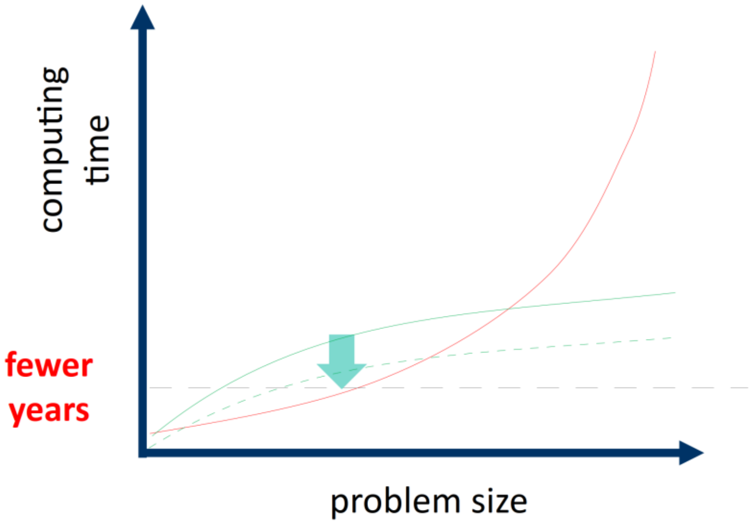
The ‘funny’ discussion



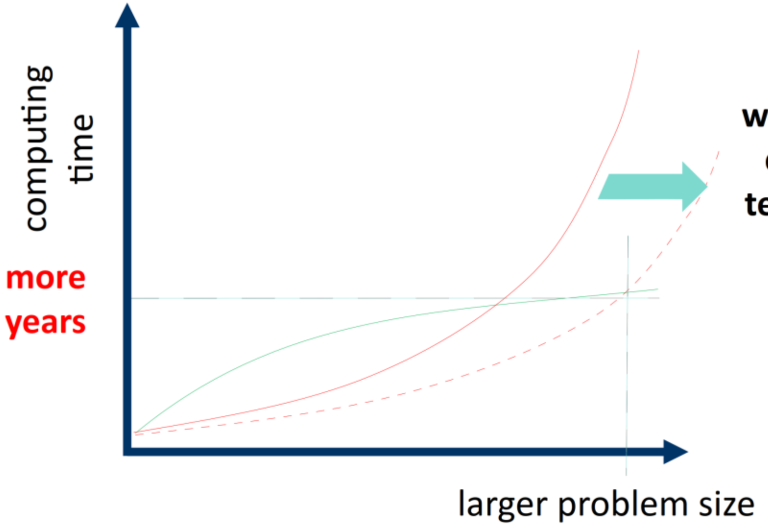
what the quantum computing theoretician sees



what the user sees



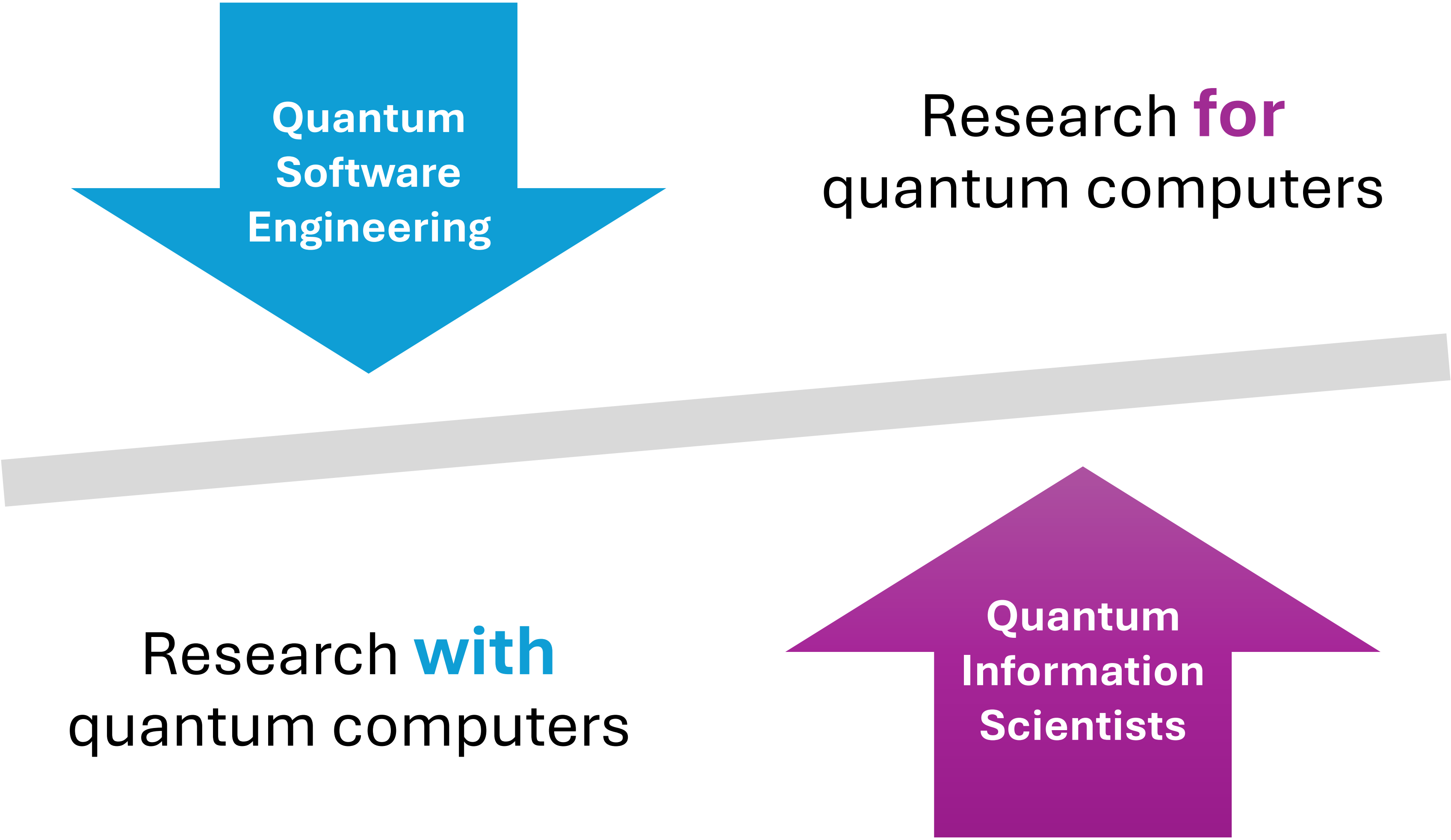
what quantum computing technologists are trying to do



what classical computing technologists are doing

Olivier Ezratty, September 2025.

Quantum Software Engineering Matters



The Road Ahead for QSE?

Service-Oriented Computing (SOC)

- **Challenges:** Platform independence and interoperability, demand/capacity management.
- **Future Directions:** Standards and APIs, and tools for quantum and classical services integration

Model-Driven Engineering (MDE)

- **Challenges:** Design of high-level methodologies for hybrid systems, scalability in quantum software maintenance, and code generation.
- **Future Directions:** Improvement of code generation and orchestration of hybrid systems.

Testing and Debugging

- **Challenges:** Test scalability, optimization of test oracles, and transition from simulators to real.
- **Future Directions:** Noise reduction techniques, efficient oracles, specialized debugging tools, and mutation and metamorphic testing?

The Road Ahead for QSE?

Programming Paradigms

- **Challenges:** Complexity of quantum circuits and reuse of quantum software.
- **Future Directions:** Abstractions and composition of quantum programs.

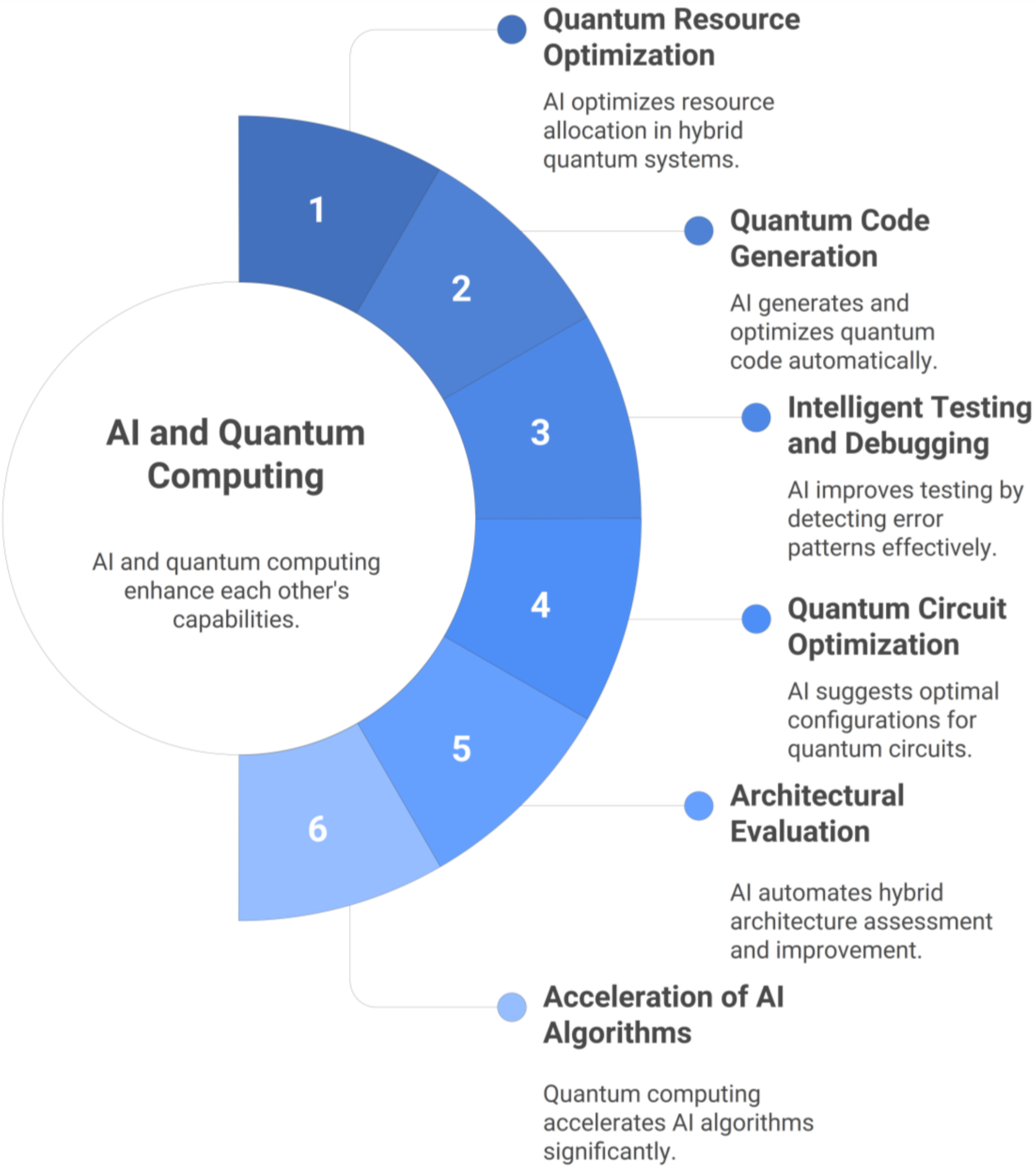
Software Architecture

- **Challenges:** Architectural decisions for integrating quantum and classical systems.
- **Future Directions:** Empirical studies on design patterns and architectural evolutions.

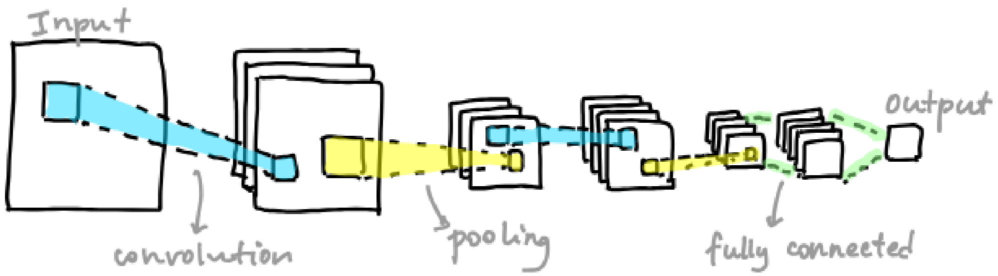
Software Development Processes

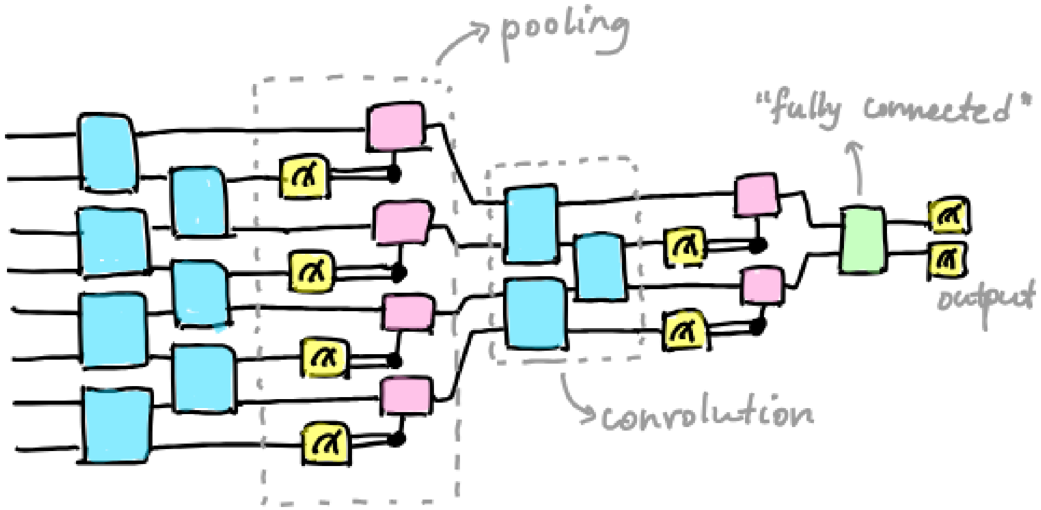
- **Challenges:** Iterative development models and risk management in hybrid software.
- **Future Directions:** Agile methodologies and DevOps tools to quantum environments.

Unveiling the synergy of AI and QC



Quantum Convolutional Neural Network





Takeaway #1

Quantum Computing is here and now

This is a today technology more than a tomorrow one

Still, some technical milestones are expected for boosting its adoption

Skepticism is a normal and healthy attitude in science

Takeaway #2

Quantum Software is crucial to get the quantum advantage

Takeaway #3

The business information systems will

combine classical and quantum software

Software modernization is needed

Takeaway #4

The Quantum Workforce is key

Be ready

Get the elephant out of the room!

Please, ask me

THE
AGE
OF
QUANTUM
COMPUTING
HAS
NOW
ARRIVED

Universidad de
Castilla-La Mancha

Ricardo Pérez del Castillo

Faculty of Social Sciences &
Information Technologies
Av. Real Fábrica de Seda, s/n
45600 Talavera de la Reina – Spain

ricardo.pdelcastillo@uclm.es
www.linkedin.com/in/ricardo-perez-castillo

Alarcos Research Group

Facultad de
Ciencias Sociales y
Tecnologías de la Información
Talavera de la Reina. UCLM

23rd International Conference on
Software Engineering and Formal Methods

SEFM 2025

Quantum Software in Action
Challenges and Opportunities in Software Engineering

Ricardo Pérez del Castillo

UCLM Universidad de Castilla~La Mancha

10-14 November 2025, Toledo, Spain