Computer science

An operating system for quantum computers

Claudio Cicconetti

Quantum computers have gained the powerful abstractions that allow programmers of classical computers to design and integrate new apps and hardware, and connect devices into networks with ease. **See p.321**

Abstractions are the keystone of the dynamic ecosystem of information and communication technology. You can think of them as allowing you to drive a car without understanding how the engine or transmission system works. For most drivers, a car is just a box with a steering wheel, pedals and a gearstick. The details of whether it is a full-electric vehicle or a model with a conventional engine, say, are largely irrelevant.

In computing, abstractions in the design of operating systems and interfaces mean that a company or research group does not need to reconfigure transistors in a computer's processor, or design an entirely new way to interface with the Internet, to make an innovative idea happen. Instead, developers can just plug their innovation into pre-existing, standardized schemes.

But what's true for conventional, classical computers has not, up until now, applied to potentially one of the most disruptive innovations in information technology – quantum computers. That changes with the work of Delle Donne *et al.*[1] (see page 321). The authors introduce a set of abstractions and standardized interfaces for running quantum applications distributed across devices in a quantum network. This is a major advance on previous test-bed implementations of quantum networks[2], which required custom hardware and software tailored for each specific set-up. As of now, the wheel does not have to be reinvented every time.

To understand a little more about the importance of abstractions in computing, let's take a step back. The end user of a classical computer – be it a smartphone, a laptop, a data-centre server or a game console – will generally run applications such as messaging or word-processing apps, or web servers or games. But these applications don't interact directly with the hardware. Instead, they rely on an operating system to provide a simplified, uniform interface to resources such as the central processing unit, working memory and peripheral gadgets such as hard drives or webcams.

The idea of process abstraction is central to the design of any operating system (Fig. 1a). The operating system does not care about the internal details of the applications it is running, or what they are doing: they are just 'black box' processes that interface to it in a uniform way. Similarly, each application runs without 'awareness' of any other application, as if it has exclusive control over the hardware. Process abstraction is how modern operating systems enable multitasking, running multiple applications as simultaneous processes without them interfering with one another.

Even the operating system itself does not interact directly with hardware, but does so through specialized software modules called drivers. These drivers represent another abstraction: they offer a standardized interface that allows operating-system developers to support various hardware technologies – whether an old magnetic hard disk or a modern solid-state one – without rewriting core software, simply by adding a new driver module. This is the principle of 'implementation isolation', which emerged in the 1960s as mainframe computers, which were originally designed for single tasks, began evolving into modern multitasking machines. The transformation was driven by foundational concepts introduced by Dennis Ritchie and Ken Thompson at Bell Labs with their UNIX operating system, which continue to shape computing today[3].

Applications that need to communicate over a network use another key abstraction: network sockets. Just as an electrical socket connects devices to the power grid, so network sockets allow applications to send and receive data across the Internet, independent of the

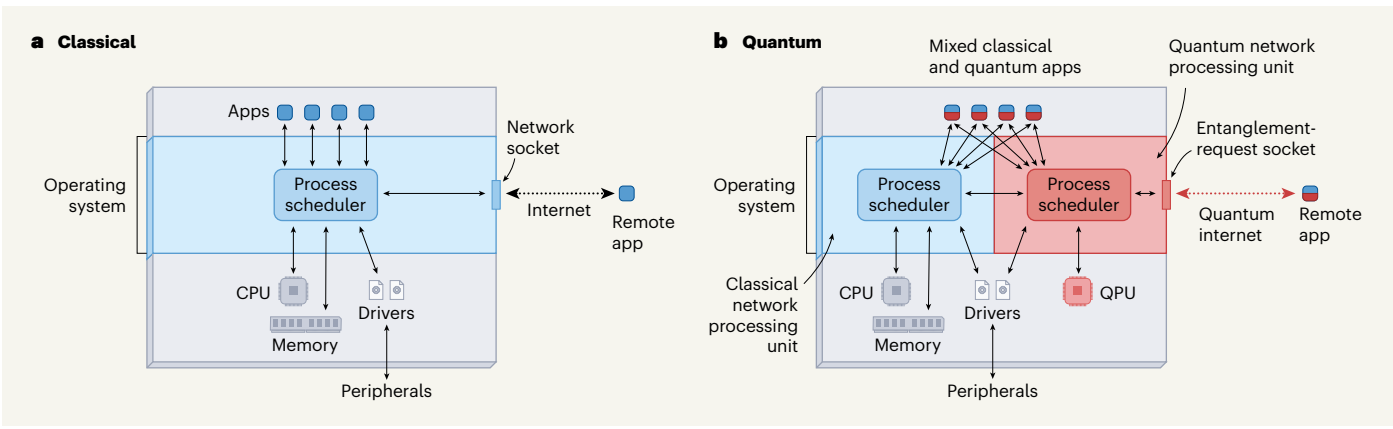

Figure 1 | Classical and quantum computing paradigms compared. **a**, High-level view of a classical computer. An operating system allows multiple applications (apps) to be run simultaneously, using a scheduler to regulate demands on working memory and the central processing unit (CPU). Applications access peripheral hardware devices such as hard drives or webcams through drivers, which allow different technologies to be used with a uniform interface. A network socket allows an application to exchange bits through the Internet with a remote peer. **b**, Simplified blueprint of QNodeOS, a similar scheme for quantum computing[1]. Applications consist of classical (blue) and quantum (red) blocks. Multitasking is enabled by a classical network processing unit, whereas a quantum network processing unit oversees quantum bit (qubit) operations and access to a quantum processing unit (QPU). Networking operations require the use of quantum entanglement. An entanglement-request socket abstraction provides a uniform way to establish entanglement between a qubit in a local unit and another one in a second remote unit through a quantum Internet.

underlying network transmission technology, whether it involve fibre optics, copper cables or wireless links.

Quantum computing and quantum communications have recently made significant progress, raising expectations that an Internet of networked quantum devices might be imminent (or perhaps not). But quantum technologies operate under fundamentally different principles from those that gave shape to modern computing systems and network protocols. Unlike classical bits, quantum bits (qubits) cannot be copied or shared freely, owing to a feature of quantum mechanics known as the no-cloning theorem. Moreover, remote interactions between quantum computers depend on entanglement. This is a phenomenon with no direct classical counterpart, in which operations on one qubit instantaneously affect another, even at a distance[4].

These differences necessitate new abstractions that are still as intuitive and flexible as those developed for classical computing. The QNodeOS operating system developed by Delle Donne and colleagues[1] is a theoretical and experimental framework that provides these for networked quantum computers (Fig. 1b). In QNodeOS, an application consists of intertwined quantum and classical procedures, or blocks. This architecture enables developers to use quantum computing resources only when needed, while offloading routine computations to classical systems.

Within a quantum block, developers can perform local operations on a quantum processing unit, or establish entanglement with remote qubits through a quantum network processing unit (QNPU). Much like the Internet Protocol stack, which regulates access to the Internet in classical networking, the QNPU abstracts the complexity of quantum communication by offering developers quantum sockets, with a uniform set of operations independent from the underlying network technologies. Furthermore, the principle of implementation isolation is maintained: the QNPU does not access quantum network hardware directly, but instead interacts through quantum devices. Each of these has its own driver — itself abstracted — allowing for distinct underlying technologies.

To validate their approach, the authors demonstrated a simple, yet representative, network application: delegated quantum computation, in which a task is split between two quantum computers. This involved generating a pair of entangled qubits spread across two quantum network nodes, a precursor step to quantum teleportation — transferring a qubit state between two nodes by destroying it at one and recreating it at the other. Entanglement generation was made possible by introducing quantum sockets, an abstraction enabling seamless quantum communication. QNodeOS can also manage concurrent execution of multiple quantum applications on the same quantum computer through a process scheduler. The hardware independence of these abstractions was demonstrated by implementing QNodeOS on two different quantum processors: one using nitrogen-vacancy centres in diamonds[5], the other a trapped-ion system based on a single calcium atom[6].

Nobody can say for certain whether QNodeOS will become 'the new UNIX'. The success of abstractions and interfaces depends largely on how well they align with the needs of mainstream applications — something that is highly uncertain at this early stage of quantum computing. However, QNodeOS is a step in the right direction and can lead quantum computing and networking into a new phase of maturity. By establishing standardized abstractions, researchers and engineers can accelerate progress, replicating the transformative impact that classical computing and networking abstractions had on technology, industry and society.

Claudio Cicconetti is at the Institute of Informatics and Telematics of the National Research Council, Pisa, Italy.
e-mail: claudio.cicconetti@iit.cnr.it

1. Delle Donne, C. *et al. Nature* **639**, 321–328 (2025).
2. Main, D. *et al. Nature* **638**, 383–388 (2025).
3. Ritchie, D. M. *Bell Labs Tech J.* **63**, 1577–1593 (1984).
4. Couteau, C. *et al. Nature Rev. Phys.* **5**, 326–338 (2023).
5. Doherty, M. W. *et al. Phys. Rep.* **528**, 1–45 (2013).
6. Teller, M. *et al. AVS Quantum Sci.* **5**, 012001 (2023).

The author declares no competing interests.

Cancer

Abnormal RNA processing suggests therapy targets

Al Charest

Tumour cells often have problems processing messenger RNA. The finding that these splicing errors result in commonly expressed peptides that are recognized by immune cells offers a target for cancer treatments. **See p.463**

Cancer cells are recognized by the immune system as being distinct from normal tissues because they present immune-stimulatory peptides, known as neoantigens, that are derived from proteins encoded by mutated genes. A common mechanism of resistance to immunotherapies is the low abundance and lack of widespread expression of these neoantigens in a tumour. An ideal tumour neoantigen would be uniformly expressed across all tumour cells, efficiently processed and presented on the surface of the cell for recognition and activation by immune-system T cells, and be a target found in many people who have cancer. On page 463, Kwok *et al.*[1] present an extensive survey of RNA-sequencing results for tumour cells reported in databases. The authors highlight the potential of a largely underexplored class of neoantigens derived from abnormal processing of messenger RNA in tumour cells across multiple cancer types. This discovery reveals new avenues for targeting these neoantigens in cancer immunotherapies.

Cancer cells harbour gene mutations that are absent in normal cells, allowing the immune system to recognize and eliminate them under normal circumstances. Leveraging this natural immune function forms the foundation of many immunotherapies, which aim to harness the immune system to specifically target and attack cancer cells while sparing normal tissues. However, the effectiveness of these approaches is often limited in cancers with a low number of mutations and high mutation variation (heterogeneity) in an individual tumour or in tumours from different people, as is the case for brain cancers called gliomas, for example. Ideally, neoantigens that are expressed consistently across most, if not all, tumour cells, are shared among a wide range of cancers and found in many people would serve as highly attractive targets for the development of 'off the shelf' immunotherapies.

Kwok and colleagues' work focuses on a newly recognized form of cancer mutation that arises from abnormal mRNA splicing. Splicing is a processing event in which non-coding segments called introns are removed from immature mRNA and the coding segments (exons) are joined together to form a mature mRNA (Fig. 1). In this abnormal splicing, the protein-coding segments aren't all joined up in the usual way, and a join at an abnormal location in the sequence generates a junction, called a neojunction, in the mRNA sequence.

It is well established that such aberrant splicing of RNA can generate unusual and at