QUANTUM NETWORK **ABSTRACTION**

April 2023

Contents

Abstract 2
1 Introduction 2
 1.1 Defining abstraction 2
 1.2 Literature review 4
2 Standards bodies 6
 2.1 Internet Engineering Task Force (IETF) 6
 2.2 European Telecommunications Standards Institute (ETSI) 8
 2.3 International Telecommunication Union 10
 2.4 Institute of Electrical and Electronic Engineers (IEEE) 11
 2.5 Global System for Mobile Communications (GSMA) 13
 2.6 Quantum Internet Alliance 16
3 Quantum network simulation packages 16
 3.1 SimulaQron 16
 3.2 Quantum NETwork SIMulator (QuNetSim) 17
 3.3 Simulator for Quantum Networks and CHannels (SQUANCH) 17
 3.4 Quantum Internet Simulator Package (QuISP) 18
 3.5 Simulator of Quantum Network Communication (SeQUeNCE) 19
 3.6 The NETwork Simulator for QUantum Information with Discrete events (NetSquid) 20
 3.7 Quantum Key Distribution Network Simulation Module (QKDNetSim) 20
 3.8 Quantum/classical 20
 3.9 QuNET 21
 3.10 NetQASM 22
4 Conclusion 22

About the Quantum Economic Development Consortium

The Quantum Economic Development Consortium (QED-C®) is an industry-driven consortium managed by SRI International. With a diverse membership representing industry, academia, government, and other stakeholders, the consortium seeks to enable and grow the quantum industry and associated supply chain. For more about QED-C, visit our website at quantumconsortium.org.

Acknowledgments

QED-C thanks the QED-C Standards and Performance Metrics Committee and, particularly, GE Research and Stephen F. Bush for material contributions to this report.

Abstract

This report provides findings on quantum network abstractions developed by defining high-level components, input/output, and controls based upon examination of proposed abstractions from standards bodies and quantum network simulation packages. This is the first in a series of two QED-C reports on quantum networks; the subsequent report presents findings on quantum network parameters.[1]

1 Introduction

Abstraction represents the simplifications that all human minds must make when addressing the complexities of real problems. The good thing is that abstraction can make the problem tractable; the bad thing is that they may "throw out the baby with the bath water" and thereby hide the real issue needing solution.

Humans need to interact with most networks, so the graphical user interfaces (GUIs) and input methods needed for that are important. They are classical, of course, since humans' senses read classical information only. Therefore, the interfaces for this human interaction with quantum systems may be somewhat different than with classical networks. For example, the probabilistic characterization of the outputs may need to be presented to the human, as may the entanglement status inside the machine (if it can be determined). Here we consider what else a human might need to know about a quantum network in operation that is not present in a classical network.

1.1 Defining abstraction

Abstraction[2] is the purposeful suppression, or hiding, of some details of a process or artifact to more clearly highlight other aspects, details, or structure.[3] Abstraction defines how a problem is viewed: which aspects are emphasized, and which are hidden or ignored.

The notion of layering, as defined by the Open Systems Interconnection (OSI) model,[4] is one of the quintessential examples of abstraction.[5] In this hierarchy of abstraction, details that would render communication networking intractable are abstracted into layers with well-defined functionality, where focusing on each layer in isolation reduces the challenge of communication into an easier, more modular problem. Good abstractions are insights that later appear to be simple, common-sense notions once understood, but require deep insight to discover. Layering is now taken for granted in almost all aspects of communication and computation.[6] Another example is software-defined networking (SDN), which distinguishes the control plane and data plane of a communication network. SDN uses software-based controllers or application programming interfaces (APIs) to communicate with underlying hardware infrastructure (control plane) and direct traffic on a network (data plane). This abstraction differs from

[1] QED-C, Quantum Network Parameters (2023).

[2] Martin Casado, Nate Foster, and Arjun Guha, "Abstractions for Software-Defined Networks," *Communications of the ACM* 57, no. 10 (October 23, 2014): pp. 86-95, doi:10.1145/2661061.2661063.

[3] See Oregon State University, Chapter 2, https://web.engr.oregonstate.edu/~budd/Books/oopintro3e/info/chap02.pdf.

[4] ISO/IEC 7498-1:1994 Information Technology — Open Systems Interconnection — Basic Reference Model: The Basic Model, (June 1999). Retrieved 26 August 2022.

[5] For example, see ISO-OSI model Diagram in https://benchpartner.com/computer-fundamentals-and-applications/communication-protocol-and-osi-reference-model/89/77/content.

[6] For example, see "Layers of Abstraction" illustration in https://www.secplicity.org/2018/09/19/understanding-the-layers-of-a-computer-system/.

that of traditional networks, which use dedicated hardware devices, such as routers and switches, to control network traffic. SDN can create and control a virtual network—or control traditional hardware—via software. While network virtualization allows organizations to segment different virtual networks within a single physical network, or to connect devices on different physical networks to create a single virtual network, software-defined networking enables a new way of controlling the routing of data packets through a centralized server. This notion may seem like common sense, but that only happened after its discovery and socialization. Abstraction also impacts the effectiveness of research and development by emphasizing the attributes of a problem to be solved. Working with a poor abstraction that emphasizes the wrong attributes of a problem will lead to poor research and development outcomes.

If quantum technology is fundamentally unique and provides any practical benefit over classical networking technology that can be exploited, then that benefit should be reflected in its abstraction. Which quantum effects (if any) will be exposed, and which hidden by abstraction? If no quantum effects will be exposed, then what is the benefit of quantum networking? A proper abstraction is critical to successful adoption of new technology. It will also shed light on the high-level components, input/output, and controls required.

In the world of classical computing, the Turing Machine[7] has been the preferred abstraction.[8] It attempts to remove intractable details of computation and focus on only the essential elements of computation, for example, output produced by state changes driven by a program and its input. The quantum circuit diagram[9] abstraction similarly focuses on output produced by input and state changes driven by a program comprised of fundamental quantum gate operations. However, it ignores details such as error and timing, which may be a partial cause for the slow development of a means to adequately address these challenges. In addition, the quantum circuit diagram has never officially been standardized, which leads to misunderstandings when non-obvious or ambiguous annotations are used.

Quantum network simulators follow the common abstraction of a communication network as a graph with vertices representing processors and edges representing communication links. Again, this abstraction is so common that it is never questioned. Thus, all quantum network simulators have been designed as partitioned quantum circuit diagrams following a communication network as a graph abstraction to interconnect the partitions. Most quantum network simulator abstractions tend to hide details related to error, timing, and synchronization, which are the hard problems that have stymied progress in quantum technology. This is not to say that these aspects are completely missing, but they must be developed and pre-configured by the user in an ad hoc manner. In other words, the user must have a full understanding of the problem and be able to characterize it before simulation, which is antithetical to the notion of abstraction. Many quantum network simulators also assume a software-defined network architecture with a controller and software-defined nodes. Again, this abstraction comes from classical communication networking and there is no direct quantum abstraction.

This report assumes these nearly universal abstractions are already incorporated into standards and simulators, unless otherwise indicated.

[7] AM Turing, "Computing Machinery and Intelligence," *Mind* 59(236), pp. 433–460, (1950).
[8] For example, see Figure 2.1 in Mueller, M., "Quantum Kolmogorov Complexity and the Quantum Turing Machine" (2007), arXiv:0712.4377.
[9] Richard P Feynman, "Quantum Mechanical Computers," Foundations of Physics, Springer Science and Business Media LLC, 16 (6): 507–531, (1986), doi:10.1007/bf01886518. ISSN 0015-9018. S2CID 122076550.

1.2 Literature review

This section reviews selected literature that appears relevant to quantum network abstraction.

"Tools for quantum network design"[10] provides a summary of quantum network simulation tools and their design approaches. This includes modeling of a quantum network and the mathematical abstraction of different components in a quantum network.

It states the well-known differences between quantum and classical networks, where the main difference between simulating quantum and classical networks is that the state of the network can be a quantum state. This implies the intractability of describing a quantum state as an aggregate of the local description of the states at each network node where the state grows exponentially with the number of quantum bits (qubits) in the network. Techniques are used to reduce the exponential growth, such as limiting to only Clifford gates (a set of mathematical transformations that normalize the n-qubit Pauli group, i.e., map tensor products of Pauli matrices to tensor products of Pauli matrices through conjugation) and Pauli noise (noise on n-qubit Pauli channels, which are quantum channels that randomly apply an n-qubit Pauli operator to the input state), or limiting to many small, entangled states that dynamically interact, shrink via measurement, or are fused but never grow larger than a few qubits.

The abstraction emerges as a pair of entangled qubits shared by spatially separated nodes being defined as a "link." Entanglement generation refers to the sharing of a Bell state between two nodes in the network, which are directly connected through a communication channel, such as an optical fiber. The generated entanglement is referred to as an elementary link.

The distance limit arising due to decoherence can be overcome by adding nodes in between, so-called quantum repeaters, which perform entanglement swaps to connect two short-distance links into a single long-distance one. Typically, entanglement swaps are probabilistic, with a fixed success probability that is normally independent of the states swapped but depends on the physical implementation. Performing an entanglement swap on two imperfect Bell states yields long-distance entanglement of lower fidelity than the original two states individually had, and its quality is further decreased by imperfections in the quantum operations. In principle, entanglement distillation can be used to improve the fidelity by probabilistically converting multiple low-quality entangled pairs of qubits into a single one of high quality.

A more general model is a probabilistic mixture of the four Bell states. This state is always achieved by applying local Pauli gates randomly to an arbitrary state of a pair of qubits, and it is not less entangled than a Werner state (bipartite quantum state that is invariant under all unitary operators). Imperfections of the quantum devices, for example, operational noise and detector inefficiencies, are commonly modeled by depolarizing, dephasing, or amplitude damping channels. Particularly relevant in the context of entanglement generation using probabilistic components is the noise caused by time-dependent memory decoherence; in case multiple links are needed, the earliest link is usually generated before the others are ready and, thus, must be stored in a quantum memory. For simplicity, the network nodes can be modeled by a fully connected quantum information processing device capable of generating entanglement in parallel with its neighbors.

[10] Koji Azuma, Stefan Bäuml, Tim Coopmans, David Elkouss, and Boxi Li, "Tools for Quantum Network Design," AVS Quantum Sci. 3, 014101 (2021), https://doi.org/10.1116/5.0024062.

“Experimental demonstration of entanglement delivery using a quantum network stack”[11] suggests several abstractions. The abstraction of tasks and services offered by the quantum network should enable platform-independent applications to be executed without the knowledge of the underlying physical implementation. They experimentally demonstrate, using remote solid-state quantum network nodes, a link layer, and a physical layer protocol for entanglement-based quantum networks. The link layer abstracts the physical-layer entanglement attempts into a robust, platform-independent entanglement delivery service.

The link layer abstracts the generation of entangled states between two physically separated solid-state qubits into a robust and platform-independent service. An application can request entangled states from the link layer and then, in addition, apply local quantum operations on the entangled qubits in real-time. Using the link layer, full state tomography of the generated states is performed to calibrate the system to achieve remote state preparation—a building block for blind quantum computation—as well as to measure the latency of the entanglement generation service.

A software stack comprises a hardware abstraction layer (HAL) to interface with the physical layer's device controller. An instruction processor dispatches instructions either directly to the physical layer, or to the link layer protocol in case a remote entangled state is requested by the application.

At the application layer, a simple platform-independent routine is sent to the network controller. The network controller implements the platform-independent stack—in this work, only the link layer protocol—and a HAL to interface with the physical layer's device controller. The link layer schedules entanglement requests and synchronizes with the remote node (on a local area network, LAN) using a time-division multiple access (TDMA) schedule computed by a centralized scheduler (external). At the physical layer, the device controller fetches commands from—and replies with outcomes to—the network controller. Driven by a clock shared with the neighboring node, it performs hard real-time synchronization for entanglement generation using a digital input/output (DIO) interface. By controlling the optical and electronic components (e.g., an arbitrary waveform generator), the device controller can perform universal quantum control of the communication-qubit in real time, as well as attempt long-distance entanglement generation with the neighboring node.

While not offering abstractions per se, a lower-level simulator, “the virtual quantum optics laboratory”[12] is worth mentioning as it attempts to provide the look and feel of real quantum optics laboratory equipment in a simulation. It operates in time segments of one microsecond. The actual runtime of an experiment will depend on the local machine and the complexity of the experiment: a one-millisecond experiment typically runs in about 15 seconds of real time. Setting the time to exactly one microsecond gives something that looks like a single photon, though this is not how the Virtual Quantum Optics Laboratory models quantum light. In effect, something like this tool could provide details that are often missing from current abstractions.

[11] Pompili, M., “Experimental Demonstration of Entanglement Delivery Using a Quantum Network Stack”, *Quantum Information* 8 (2022), doi:10.1038/s41534-022-00631-2.
[12] BR La Cour, M Maynard, P Shroff, G Ko, and E Ellis, “The Virtual Quantum Optics Laboratory” (2021), arXiv:2105.07300.

2 Standards bodies

This section examines standards development organizations and their attempts to abstract a quantum network. Table 1 in the Conclusion section presents a summary of abstractions across the following standards efforts.

2.1 Internet Engineering Task Force (IETF)

The Quantum Internet Research Group (QIRG)[13] has been working on identifying use cases and a framework of principles for creating a quantum internet. It identifies three basic schemes for heralded entanglement[14] generation on a link through coordinated action of the two nodes at the two ends of the link:

- "At mid-point" where an entangled photon pair source is positioned midway between the two nodes with matter qubits sending an entangled photon through a quantum channel to each of the nodes. There, transducers are invoked to transfer the entanglement from the flying qubits to the matter qubits. In this scheme, the transducers know if the transfers succeeded and are able to herald successful entanglement generation via a message exchange over the classical channel.
- "At source" where one of the two nodes sends a flying qubit that is entangled with one of its matter qubits. A transducer at the other end of the link will transfer the entanglement from the flying qubit to one of its matter qubits. Just like in the previous scheme, the transducer knows if the transfer succeeded and can herald successful entanglement generation with a classical message sent to the other node.
- "At both end-points" where both nodes send a flying qubit that is entangled with one of their matter qubits. A detector somewhere between the nodes performs a joint measurement on the two qubits, which stochastically projects the remote matter qubits into an entangled quantum state. The detector knows if the entanglement succeeded and is able to herald successful entanglement generation by sending a message to each node over the classical channel.

The "mid-point source" scheme is more robust to photon loss, but in the other schemes the nodes retain greater control over the entangled pair generation. Note that while photons travel in a particular direction through the quantum channel, the resulting entangled pair of qubits does not have a direction associated with it. Physically, there is no upstream or downstream end of the pair.

The standard technique of entanglement swapping (involving the transfer or teleportation of entanglement to two photons that were produced independently and never previously interacted) is used to extend the length of an entangled state distribution channel. Here, one may make the obvious abstraction of hop-by-hop layer for individual single hops and longer entangled composite hops as a higher layer.

With respect to the software defined abstraction, the data plane separation is much more distinct and there will be two data planes: a classical data plane that processes and forwards classical packets, and a

[13] Datatracker, "Architectural Principles for a Quantum Internet" (2023), https://datatracker.ietf.org/doc/draft-irtf-qirg-principles/
[14] Heralded entanglement is when spontaneous parametric down-conversion converts a single incoming photon to two outgoing photons where one measures one photon that indicates there must be a second photon as well.

quantum data plane that processes and swaps entangled pairs. Third generation quantum networks may also forward qubits in addition to swapping Bell pairs.

In addition to control plane messages, there will be control information messages that operate at the granularity of individual entangled pairs, such as heralding messages used for elementary link generation. In terms of functionality, these messages are closer to classical packet headers than control plane messages and thus they are considered to be part of the quantum data plane. Therefore, a quantum data plane also includes the exchange of classical control information at the granularity of individual qubits and entangled pairs.

Quantum repeaters (QRs) are identified as the core building block of a quantum network. However, a quantum repeater will have to do more than just entanglement swapping in a functional quantum network. Its key responsibilities can include:

- Creating link-local entanglement between neighboring nodes
- Extending entanglement from link-local pairs to long-range pairs through entanglement swapping
- Performing distillation (the transformation of N copies of an arbitrary entangled state into some number of approximately pure Bell pairs) to manage the fidelity of the produced pairs
- Participating in the management of the network (including routing)

Not all QRs in the network will be the same; they are further broken down into:

- Controllable quantum nodes (quantum routers) - A quantum repeater with a control plane that participates in the management of the network and will make decisions about which qubits to swap to generate the requested end-to-end pairs.
- Automated quantum nodes - A data-plane-only quantum repeater that does not participate in the network control plane. Since the no-cloning theorem precludes the use of amplification, long-range links will be established by chaining multiple such automated nodes together.
- End-nodes - In a quantum network these must be able to receive and handle an entangled pair, but they do not need to be able to perform an entanglement swap (and are not necessarily QRs). End-nodes are also not required to have any quantum memory as certain quantum applications can be realized by having the end-node measure its qubit as soon as it is received.
- Non-quantum nodes - Not all nodes in a quantum network need to have a quantum data plane. A non-quantum node is any device that can handle classical network traffic. Additionally, two kinds of links that will be used in a quantum network need to be identified :
    - Quantum links (QL) - A link that can be used to generate an entangled pair between two directly connected QRs. This may include additional mid-point elements. It may also include a dedicated classical channel that is to be used solely for the purpose of coordinating the entanglement generation on this quantum link.
    - Classical links - A link between any node in the network that is capable of carrying classical network traffic.

Note that passive elements, such as optical switches, may feasibly transmit quantum states. Therefore, it is possible to connect multiple quantum nodes with each other over an optical network and perform optical switching rather than routing via entanglement swapping at quantum routers. This requires coordination with the elementary link entanglement generation process, and it still requires repeaters to overcome the short-distance limitations. However, this is a potentially feasible architecture for local area networks.

An application (App) running on two end-nodes attached to a network will at some point need the network to generate entangled pairs for its use. This may require negotiation between the end nodes (possibly ahead of time) because they must both open a communication endpoint that the network can use to identify the two ends of the connection. The two end-nodes use a classical channel available in the network to achieve this goal. When the network receives a request to generate end-to-end entangled pairs it uses the classical communication links to coordinate and claim the resources necessary to fulfill this request. This may be some combination of prior control information (e.g., routing tables) and signaling protocols, but the details of how this is achieved are an active research question.

During or after the distribution of control information, the network performs the necessary quantum operations such as generating entanglement over individual QLs, performing entanglement swaps at QRs, and further signaling to transmit the swap outcomes and other control information.

Since Bell pairs do not carry any user data, some of these operations can be performed before the request is received in anticipation of the demand. Note that here, “signaling” is used in a very broad sense and covers many different types of messaging necessary for entanglement generation control. For example, heralded entanglement generation requires very precise timing synchronization between the neighboring nodes, and thus the triggering of entanglement generation and heralding may happen over its own.

Higher level signaling with less stringent timing requirements (e.g., control plane signaling) may then happen over its own communication links. The entangled pair is delivered to the application once it is ready together with the relevant pair identifier. However, being ready does not necessarily mean that all link pairs and entanglement swaps are complete, as some applications can start executing on an incomplete pair. In this case the remaining entanglement swaps will propagate the actions across the network to the other end, sometimes necessitating fixup operations at the end node.

2.2 European Telecommunications Standards Institute (ETSI)

The Industry Specification Group (ISG) on Quantum Key Distribution (QKD)[15] has performed groundbreaking standard development work in laying a foundation for specifying quantum channels, in particular, their terminology and characteristics, which the National Institute of Standards and Technology (NIST) has recently leveraged. However, the abstractions developed within the ETSI QKD specifications are primarily oriented toward quantum key generation and key management, as it should be.

The following specifications have been released in the ETSI QKD series:

- Orchestration interface for software defined networks
- Control interface for software defined networks
- Application interface
- Device and communication channel parameters for QKD deployment
- Protocol and data format of rest-based key delivery API
- Vocabulary

[15] ETSI, “Industry Specification Group (ISG) on Quantum Key Distribution (QKD)” (2023), https://www.etsi.org/committee/1430-qkd.

- Components and internal interfaces
- Component characterization: characterizing optical components for QKD systems
- Security proofs
- QKD module security specification
- Application interface
- Use cases

As described in the introductory clause, QKD interfaces are an abstraction of the QKD systems, which are contained within a secure location as part of a software defined (SD) QKD node. This abstraction allows an SDN controller to identify all the QKD systems within a location and aggregate them as a single network element with multiple interfaces (e.g., as a switch or a router, with very different capabilities). Due to interoperability issues, the current version of the model shall specify the QKD technology implemented by the device and the vendor and model, as mismatching different QKD modules in the current state of development will lead to inoperative links with no key generation.

A QKD Key Association Link is a logical key association between two remote SD-QKD nodes. These associations can be of two different types: (1) direct (also called physical), if there is a direct quantum channel through which keys are generated, i.e., a physical QKD link connecting the pair of QKD modules, or (2) virtual, if keys are forwarded (key relay) through several SD-QKD trusted nodes to form an end-to-end key association (i.e., there is no direct quantum channel connecting the endpoints, and a set of them have to be concatenated such that each secret key is produced and then used to relay a key from the initial to the endpoint in a multi-hop way). Any new key association link created in an SD-QKD node has to be tracked, labeled, and isolated from other links. Virtual links are also registered as internal applications, as they make use of QKD-derived keys from other QKD key association links for the key transport. The information exported to the SDN controller should be kept minimal but sufficient to allow analysis and optimization of the deployed links and applications. In contrast, other crucial information (e.g., the QKD-derived keys) shall be kept within the SD-QKD node security perimeter.

The purpose of the information model presented in the present document, regardless of the protocol chosen to implement the control channel, is to simplify the management of the QKD resources by implementing an abstraction layer described in YANG (a data modeling language for the definition of data sent over network management protocols). This will allow the user to optimize the creation and usage of the QKD-derived keys by introducing a central element through the SDN controller. This is a standard component of SDN networks that has a global view of the network. This abstraction layer will enable the SDN controller to simultaneously manage both the classical and quantum parts of the network. The integration has the added benefit of using well-known mechanisms and tools in the classical community, which will facilitate its adoption and deployment by the telecommunications world.

An SD-QKD node is an aggregation of one or more QKD modules that interface with an SDN controller using standard protocols (i.e., it is SDN-enabled). The connection between node and controller allows information to be retrieved from the QKD domain and dynamically and remotely configures the behavior of the QKD systems to create, remove, or update key associations (either through a quantum channel or via multi-hop) between remote secure locations. An SD-QKD node shall be compliant with some specific requirements:

- An SD-QKD node shall comprise at least one QKD module, exposed to the controller as a QKD interface.
- It may comprise multiple QKD modules, creating an abstraction of a single node with multiple interfaces.

2.3 International Telecommunication Union

The International Telecommunication Union (ITU) has several sets of QKD specifications including:

- ITU-T Focus Group on Quantum Information Technology for Networks (FG QIT4N)[16]
- Y.3800-Y.3999 Quantum key distribution networks[17]

The ITU-T Focus Group on Quantum Information Technology for Networks (FG QIT4N) contains:

- D1.1: Quantum information technology for networks terminology: Network aspects of quantum information technologies
- D1.2: Quantum information technology for networks use cases: Network aspects of quantum information technologies
- D1.4: Standardization outlook and technology maturity: Network aspects of quantum information technologies
- D2.1: Quantum information technology for networks terminology: quantum key distribution network (QKDN)
- D2.2: Quantum information technology for networks use cases: QKDN
- D2.3: Quantum key distribution network protocols: Quantum layer
- D2.3: Quantum key distribution network protocols: Key management layer, QKDN control layer and QKDN management layer
- D2.4: Quantum key distribution network transport technologies
- D2.5: Standardization outlook and technology maturity: Quantum key distribution network

A breakdown of concepts from their terminology appears in Figure 1 in D2.1: Quantum information technology for networks terminology: QKDN.

A typical QKDN consists of several layers: the quantum layer, key management layer, QKDN control layer, QKDN management layer and the service layer. Compared to traditional communication networks, the quantum layer is unique to QKDN. In the quantum layer, QKD protocols are implemented in QKD modules and symmetric keys are established through point-to-point QKD links.

In the quantum layer of a QKDN, the main protocols involved are QKD protocols, which establish symmetric keys between two trusted nodes. Other protocols and interactions may also be involved to assist the QKD process and QKDN operations such as synchronization protocols. As specified in the functional architecture model of QKDN the quantum layer mainly consists of QKD modules and QKD links, along with several interfaces and other layers.[18]

The ITU Y.3800-series specifications on quantum key distribution networks contain:

[16] ITU, "ITU-T Focus Group on Quantum Information Technology for Networks (FG-QIT4N)" (2023), https://www.itu.int/en/ITU-T/focusgroups/qit4n/Pages/default.aspx.
[17] ITU "Y.3800: Overview on Networks Supporting Quantum Key Distribution" (2023), https://www.itu.int/rec/T-REC-Y.3800/en.
[18] For an example, see Figure 1 in D2.3: Quantum key distribution network protocols: Quantum layer.

- Y.3800: Overview on networks supporting quantum key distribution
- Y.3801: Functional requirements for quantum key distribution networks
- Y.3802: Quantum key distribution networks – Functional architecture
- Y.3803: Quantum key distribution networks – Key management
- Y.3804: Quantum key distribution networks – Control and management
- Y.3805: Quantum key distribution networks – Software-defined networking control
- Y.3806: Quantum key distribution networks – Requirements for quality-of-service assurance
- Y.3807: Quantum key distribution networks – Quality of service parameters
- Y.3808: Framework for integration of quantum key distribution network and secure storage network
- Y.3809: A role-based model in quantum key distribution networks deployment
- Y.3810: Quantum key distribution network interworking – Framework
- Y.3811: Quantum key distribution networks – Functional architecture for quality-of-service assurance

A modular breakdown of the SDN controller for a quantum key distribution network reflects a hierarchical structure that represents control of the network.[19]

Although ITU-T Y.3810 considers the interworking of QKD with different technologies, there is no interworking at the quantum layer.[20] The integration is done at higher layers.

The ITU-T Y.3800 series also considers an abstraction for machine learning (ML) applied to QKD networks.[21] This is shown as a generic approach for integrating ML with QKD such that the QKD system is simply a source of data for learning and a sink for the learned control information. This enables ideation with respect to integration of ML into QKD systems and the benefits it might enable. This would of course include quantum machine learning. It should be noted that machine learning near the quantum layer would likely have to handle faster changing signals than those at the higher layers. For example, optical tuning parameters would likely change at a faster rate than user and key management parameters.

2.4 Institute of Electrical and Electronic Engineers (IEEE)

The IEEE P1913 YANG Model for Software-Defined Quantum Communication project has been operating under the COM/NetSoft-SC/QuantumComm sponsorship of IEEE Standards.[22] This standard defines a YANG model that enables configuration of quantum endpoints in a communication network to dynamically create, modify, or remove quantum protocols or applications. The protocol design facilitates future integration with Software-Defined Networking (Figure 1). The standard defines a set of quantum device configuration commands that control the transmission, reception, and operation of quantum states over a quantum network. These device commands contain parameters that describe quantum state preparation, measurement, and readout. The purpose of the YANG model is to define a classical interface to quantum communication devices that permits these devices to be reconfigured to implement a variety of protocols and measurements. There are numerous quantum communication products and protocols under development, but there is no consistent way that they are controlled or interact. Implementing,

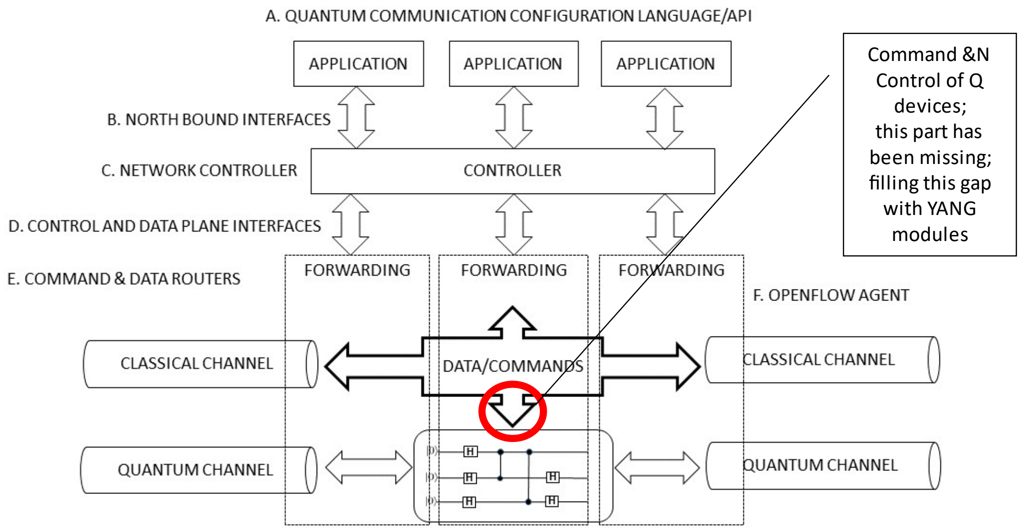
[19] For an example, see Figure 2 in Y.3805: Quantum key distribution networks – Software-defined networking control.

[20] For an example, see Figure 5 in Y.3810: Quantum key distribution network interworking – Framework

[21] For an example, see Figure 6-1 in ITU-T Y.3800-series – Quantum key distribution networks – Applications of machine learning. Figure 7-2 in the same attempts to break down the training process into more detail.

[22] See "IEEE SA - The IEEE Standards Association" (2023), https://standards.ieee.org/ieee/1913/6720/.

modifying, or integrating with the protocols in these systems often requires interaction at a very primitive level. A defined interface with quantum communication devices can greatly simplify integration efforts and can enable the development of higher-level programming tools.


A. QUANTUM COMMUNICATION CONFIGURATION LANGUAGE/API
APPLICATION
APPLICATION
APPLICATION
B. NORTH BOUND INTERFACES
C. NETWORK CONTROLLER
CONTROLLER
D. CONTROL AND DATA PLANE INTERFACES
E. COMMAND & DATA ROUTERS
FORWARDING
FORWARDING
FORWARDING
F. OPENFLOW AGENT
CLASSICAL CHANNEL
DATA/COMMANDS
CLASSICAL CHANNEL
QUANTUM CHANNEL
QUANTUM CHANNEL
Command &N Control of Q devices; this part has been missing; filling this gap with YANG modules

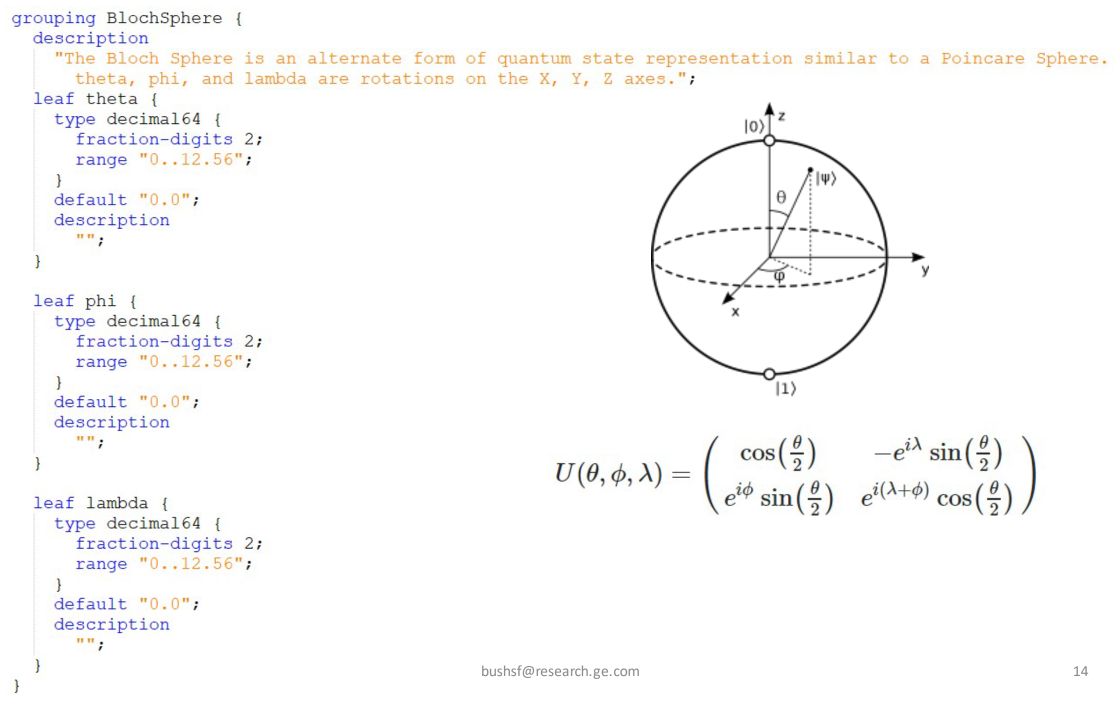
From IEEE 1913 ™/D1 Draft Standard for Software-Defined Quantum Communication (2015)

Figure 1:The software-defined network abstraction (Credit: Stephen F. Bush).

IEEE P1913 is focused on taking the same concepts that are used today in classical network definition and management and extending these concepts to current and future quantum devices and networks. Today's modern networks make heavy use of Network Configuration Protocol (NETCONF) and YANG—these technologies underlie and are the cornerstone of today's dynamic and virtualized networks—allowing rapid configuration and deployment of new technology and infrastructure and based upon commonplace standards.

IEEE P1913 is extending this into the quantum realm, allowing for the imagination of a space where detector modules from different vendors can be controlled and configured using the same network APIs—that is what NETCONF and YANG can do. This standard consists of core model components that can be extended and contributed to easily. It allows for remote control of devices over classical networks or any transmission media upon which the Internet Protocol runs (e.g., USB, RS-232).

The IEEE P1913 standard also defines how the networking world "sees" quantum technology for managing and controlling networks. Similar management will be required for quantum computing and sensing. It reduces cost by enabling quantum networking devices to fit seamlessly within network orchestration, automation of experiments, testing, certification, and network operation; decoupling of network devices from test and automation tools; flexible trade-offs for device and system settings; and monitoring system performance. The standard decouples hardware control from device implementation, and hence it does not constrain the method of implementation of the quantum aspects of the network, which are part of the lowest (physical) layer. This is done via IETF NETCONF/YANG (XML and JSON encoding), which provides a standardized way to operationally update and modify the configuration of a network device. YANG is the modeling language that describes the configuration, and NETCONF is the protocol that applies the configuration to the relevant device (see, for example, Figure 12).



```
grouping BlochSphere {
  description
    "The Bloch Sphere is an alternate form of quantum state representation similar to a Poincare Sphere.
      theta, phi, and lambda are rotations on the X, Y, Z axes.";
  leaf theta {
    type decimal64 {
      fraction-digits 2;
      range "0..12.56";
    }
    default "0.0";
    description
      "";
  }

  leaf phi {
    type decimal64 {
      fraction-digits 2;
      range "0..12.56";
    }
    default "0.0";
    description
      "";
  }

  leaf lambda {
    type decimal64 {
      fraction-digits 2;
      range "0..12.56";
    }
    default "0.0";
    description
      "";
  }
}
```

$$U(\theta, \phi, \lambda) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{i\lambda}\sin\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) & e^{i(\lambda+\phi)}\cos\left(\frac{\theta}{2}\right) \end{pmatrix}$$

bushsf@research.ge.com 14

Figure 2: A simple example of the YANG language (Credit: Stephen F. Bush).

The main abstraction utilized in IEEE P1913 is the Case diagram[23] and simplified Unified Modeling Language (UML) diagrams. The Case diagram was invented to account for the flow of messages through a network, ensuring duplicate information was removed from the management system (i.e., only orthogonal information is exposed via the management system). This abstraction is helpful, if properly extended to energy and quantum mechanical properties in a quantum network.

2.5 Global System for Mobile Communications (GSMA)

The GSMA Internet Group (IG)[24] document appears to be largely a summary of other work with the addition of a few key insights, in particular, the notion of a generalized model enabling "quantum as a service."

They note that many of the approaches so far to model quantum networks and services are done in terms of layers with different functionalities (e.g., quantum layer, key management layer), with imprecise definitions, focused on following the same general approach that has been successful in classical networks. These layered approaches assume that quantum information can be modeled in the same way as classical information, defining some sort of packetization patterns for qubits. The nature of the technologies suitable for generating and then measuring quantum signals, however, is based on the physical properties of communication channels, now and in the mid-term future. This has implications in the transfer of qubits from the transport perspective and how it is modeled.

[23] Jeffrey D. Case and Craig Partridge, "Case Diagrams: A First Step to Diagrammed Management Information Bases," SIGCOMM Comput. Commun. Rev. 19, 1 (Jan. 1989), 13–16. https://doi.org/10.1145/66093.66094.
[24] GSMA, *Quantum Networking and Service Version 1.0* (2021), https://www.gsma.com/newsroom/wp-content/uploads/IG-12-Quantum-Networking-and-Service.pdf.

Quantum communications technologies must be integrated into a general networking framework that supports a general system view, is aware of the network and is suitable to be controlled within it, ideally in an automatized way, such that optimal performance can be achieved within the technological and physical limits.

Rather than mimicking the model of the classical network, this network framework must be based on components that implement a well-defined functionality, with this functionality defining the flow of information among the components to implement a given architecture. This flow, in turn, defines the interfaces. Well-defined interfaces and clear component functionality allow for a modular view, which allows for a disaggregated building of the network, in the sense that different manufacturers can provide components that can work together.

The proposed model for a quantum services network reference framework considers the main logical components that exist in conceivable quantum networks and services, describes their relationship, points to the places where interfaces are necessary, and describes the information that must be exchanged through them.

This approach is specifically intended to support the industrialization of quantum networks. It allows for locating the points where standards are necessary in the form of agreed interfaces, disaggregating the functionality in components that can become a separate product, and making it easier to integrate these additional components in the existing telecommunications network and security ecosystem.

A quantum network is essentially a communication network, with quantum devices installed in some network points. A quantum node encapsulates all software and hardware components necessary to manage the functionality of a quantum protocol to deliver them to final applications. This might be, for example, secret keys in the case of QKD, or entangled states in a more general case.

A quantum host is the collection of quantum modules in the same node and all the components implementing the manipulation of quantum signals over a quantum channel that links the nodes. This is akin to the typical data forwarding plane in classical networking, and here the quantum part of the network can be abstracted in a similar way as the quantum forwarding plane (QFP) which, for example, in the specific case of QKD would encapsulate a full QKD protocol.

The QFP thus constitutes the foundation for the abstraction process of grouping the required network functionality in different components defined by their characteristics relevant for a particular purpose, while hiding or summarizing characteristics irrelevant to this purpose. The QFP encapsulates all the capabilities that would be added to a standard telecommunications network to make the network quantum.

To some extent, it is an attempt to clearly define, using a terminology from networks, the so-called quantum layer in early quantum network models. A clear boundary between the quantum forwarding plane and the rest would allow the discernment of which parts are new and which can be essentially taken from networking or service-oriented technologies.

In the rest of this section, the concrete case of a QKD network will be considered as it is the most mature example available. In QKD networks, the final product are the symmetric secret keys, hence the QKD protocols (from the production of the quantum states to classical post processing) belong to the QFP. The QFP concept can be equally applied to networks dedicated to entanglement distribution or the trivial network (one single node) of a QNRG. The QFP supports the consumption of its products following the as-a-service paradigm, what implies a natural convergence with current trends in cloud and distributed computing.

In a QKD network, the processing of quantum states results in symmetric keys that are tagged and stored in the local memory of the module, waiting to be delivered. This defines the QKD module, which is able to create a key between the two ends of the quantum channel. One or several of these modules interact with a Key Forwarding module, able to relay the keys from one node to the next in a chain of nodes. The next component is the QKD Control module, whose functionality is to control the QKD modules and their connections. In addition, since the maximum distance in a single link is heavily penalized because of the limitations in the physical medium, the network has to be capable of doing multi-hop links, where the key is relayed from one node to the next in a chain of nodes. To do this, part of the key is consumed, and this key is not available for the final application. This task is just key transport done on behalf of the network and is carried out, when needed, by a Trusted Repeater module, which also can be used to build interoperability in the network among QKD module manufacturers at the link level.

Note that we do not consider that the application issues its requirements directly to the control (e.g., demanding a given flow of key between two specific nodes in the network with certain characteristics), but they go through the Key Manager, which is the last component. If the Key Manager cannot satisfy the requirements, it will ask for the connections to be created by the network control. This design means that there is a unique entry point of the application in the QKD node, which facilitates both the implementation of the applications and the collection of all the requests so that the key management can decide on the priorities and preemptively ask for the creation of keys among nodes.

While QKD Control can be specific for each device (Forwarding) manufacturers, their control communications with the Key Managers and among nodes should be standardized. In the case of a centralized control, specifically in the Software Defined Networking (SDN) case, this would imply the standardization of the South Bound Interface from the SDN controller to a node controller (SDN Agent). This component-based approach also satisfies the design rule of implementing clearly distinct and non-overlapping functionalities. The functionalities that are above the quantum forwarding plane are essentially classical functionalities that can be considered extensions of components that are already in the market. Continuing with the QKD example, there are manufacturers of key management systems that would just need to add some specialized modules to their products to deal with the continuous supply of symmetric keys produced in a QKD network. In the same way, existing SDN controllers can be extended with modules to deal with the control needs of a QKD network.

In general, there are still several gaps to consider:

- The level of amplified spontaneous emission (ASE) noise from optical amplifiers, or of in-band crosstalk from classical channels, that can be tolerated by a quantum channel receiver
- The maximum transmission power that should be allowed to a classical channel not to interfere with a QKD channel
- Attenuation/reflection characteristics of all components and sub-systems in a quantum node/system
- Interfaces of all components and sub-systems in a quantum node/system

- Interfaces for interoperability and interworking of quantum nodes/systems within current networks
- Interfaces for management, control, and orchestration
- Assurance (policies and processes to ensure that services offered over networks meet a pre-defined service quality level for an optimal subscriber experience)

2.6 Quantum Internet Alliance

The Quantum Internet Alliance[25] (QIA) is a consortium of partners from quantum research groups and high-tech companies in Europe coordinated by QuTech from Delft University in the Netherlands. The QIA was a European project launched in October 2018, running until September 2021. The QIA targeted a blueprint for a pan-European Quantum Internet with ground-breaking technological advances, culminating in the first experimental demonstration of a fully integrated network stack running on a multi-node quantum network. Most of the QIA documents appear to be a re-packaging of Delft University papers and are covered elsewhere in the literature.

3 Quantum network simulation packages

This section examines quantum network simulation packages and explores their attempts to abstract the quantum network.

3.1 SimulaQron

SimulaQron[26] is not intended to simulate quantum networks, but rather to simulate/emulate quantum application development in a platform independent manner that would utilize a quantum internet. Thus, its abstractions are suited toward that end. It is implemented in Python.

It is assumed that a quantum internet consists of local quantum processors, which are interconnected by quantum communication channels that enable the transmission of qubits between the different processors. It is also assumed that quantum internet protocols would require both classical as well as quantum information to be exchanged between the network nodes, next to the execution of gates and measurements on a local quantum processor and that this requires quantum internet software to integrate classical communication programming practices with novel quantum ones. Application software can access the simulated local quantum processors to execute local quantum instruction and measurements, but also to transmit qubits to remote nodes in the network.

At the lowest level lies the quantum hardware, which in the most general case consists of a quantum processor capable of storing and manipulating qubits. The quantum processor has an optical interface over which it can generate entanglement with neighboring network nodes and send/receive qubits. Sending and receiving qubits may, depending on the implementation, be realized by first generating entanglement, followed by quantum teleportation.

The quantum hardware is controlled by a necessarily platform-dependent control system that may possess a classical communication interface to neighboring nodes, for example, to communicate the

[25] See "Quantum Internet Alliance" (2023), https://quantum-internet.team/.

[26] A Dahlberg and S Wehner, "SimulaQron—A Simulator for Developing Quantum Internet Software," *Quantum Science and Technology* 4:1, p. 015001 (2019), doi:10.1088/2058-9565/aad56e.

successful generation of heralded entanglement. Together the quantum hardware and low-level control system form the platform dependent quantum processing system.

SimulaQron provides a stand-in for such platform-dependent quantum processing systems and the quantum communication between them.

The most interesting abstraction is the universal interface, the Classical Quantum Combiner (CQC). CQC can be understood as an extended instruction set which—next to supporting "standard" quantum instructions such as performing gates or measurements—contains special features tailored to a quantum network. This includes, for example, commands to produce entanglement or transmit qubits. Applications can be realized in the platform independent part of the quantum internet system by sending the appropriate instructions using CQC to the underlying quantum processing system.

3.2 Quantum NETwork SIMulator (QuNetSim)

The Quantum NETwork SIMulator (QuNetSim)[27] is implemented in a Python framework, and its abstraction is to focus solely on the network layer. The goal of QuNetSim is to make it easier to investigate and test quantum networking protocols over various quantum network configurations and parameters.

An interesting claim related to its choice of abstraction is that QuNetSim can simulate in real time and is suited to control laboratory hardware. The claim is that quantum simulated devices can be replaced with physical hardware for "lab-in-the-loop" behavior. However, it seems unclear how real, low-level quantum devices operating with picosecond resolution can possibly interact with Python software on a non-real-time operating system in a meaningful way. An example of this would have been helpful in its explanation.

Another part of QuNetSim's abstraction is that it does not implement lower-level quantum device operation and relies on open-source qubit simulators to provide this capability as a "back end." This is an interesting approach and would seem to address the issue of having a standardized simulation of front and back end. Currently supported back ends are CQC/SimulaQron, ProjectQ, EQSN, and QuTiP.

QuNetSim claims to provide synchronization logic programmatically at a high level. The goal is to easily synchronize parties regarding qubit transmission and arrival. A distinction claimed in QuNetSim is that it adds this layer of synchronization, namely acknowledging when information arrives at the receiver.

Another abstraction is that QuNetSim gives each host an addressable quantum memory such that given an ID, they can fetch a qubit and manipulate it as desired.

3.3 Simulator for Quantum Networks and CHannels (SQUANCH)

Simulator for Quantum Networks and CHannels (SQUANCH)[28] is implemented as a Python library and appears to be rather highly abstracted.

The primary abstraction is a parallelized simulation of distributed quantum information processing agents. Agents run in parallel within separate processes and are connected by quantum and classical channels, which apply customizable error models to the transmitted information and synchronize agent clocks. Agents maintain internal clocks, which are updated when information is sent or received, allowing users to roughly quantify the performance of various networking protocols in terms of simulated elapsed time.

[27] S DiAdamo, J Nötzel, B Zanger, and M Mert Beşe, "QuNetSim: A Software Framework for Quantum Networks" (2020), arXiv:2003.06397.

[28] B Bartlett, "A Distributed Simulation Framework for Quantum Networks and Channels" (2018), arXiv:1808.07047.

SQUANCH includes rudimentary built-in timing features for agents to allow users to characterize the efficiency of protocols, taking specified values of photon pulse widths, signal travel speeds, and length of channels into account.

The QSystem is the most fundamental class, representing a multi-particle quantum state as a density matrix. Ensembles of quantum systems are efficiently handled by QStreams, and each QSystem has references to its constituent qubits. Functions in the Gates module can be used to manipulate the state of a quantum system. Agents are generalized quantum-mechanical "actors," which are initialized from a QStream instance and can alter the state of the quantum systems in their stream object, typically by interacting directly with qubits. Agents run in parallel from separate processes and are connected by quantum and classical channels, which apply customizable error models to the transmitted information and synchronize agent clocks.

3.4 Quantum Internet Simulator Package (QuISP)

The Quantum Internet Simulator Package (QuISP)[29] is implemented in OMNeT++ and introduces at least two abstractions: error handling and ruleset operation.

The QuISP abstraction related to error handling is that only deviations from the ideal quantum state need to be tracked and managed. Thus, the focus is on the amount of error rather than the details of the state itself. Deviations from the ideal state can be tracked efficiently, which will lead to more scalable simulation. Scalability is achieved by exploiting the fact that quantum networks only require a limited set of operations, and error can be discretized into a small set. Clearly, this abstraction hides the details of quantum operations and state and emphasizes management of error. Specifically, it tracks Pauli X, Y, and Z errors as well as relaxation and excitation errors. Relaxation captures energy loss due to environmental noise, and excitation captures unintended increases in energy level. Photon loss is accounted for as well. The quantum state at any instant in time is represented by an error probability vector. Each element represents the probability that the quantum state has been affected by a particular error at a specific time and the error probability vector is normalized. The evolution of this error probability vector is given by a transition rate matrix. The transition rate matrix is independent of time and the evolution of the qubit is modeled as a Markov process.

QuISP Ruleset operation has the feel of OpenFlow[30] flow tables, where an OpenFlow switch matches packets with one or more flow tables. OpenFlow flow tables contain flow entries, and packets are matched based on the matching precedence of flow entries, which is a classic example of SDN operation.

In QuISP, each node along a path has one RuleSet for the connection. Rules are represented vertically while communicating to the proper operation partners. Horizontal arrows represent partners that are coordinating actions and vertical arrows represent the order of execution.

The goal of this protocol is decentralized, autonomous but coordinated actions of the quantum repeaters with minimal classical inter-node communication. Every rule has a condition and corresponding action. Condition clauses are composed of single or multiple conditions to be met before an action is executed. An action clause is a set of operations including resource assignment changes, qubit manipulation, and

[29] R Satoh, "QuISP: A Quantum Internet Simulation Package" (2021), arXiv:2112.07093.
[30] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner, "OpenFlow: Enabling Innovation in Campus Networks. SIGCOMM Comput. Commun. Rev. 38, 2 (April 2008), 69–74, doi:10.1145/1355734.1355746.

classical message transfer. Once condition clauses are met, the corresponding action is immediately executed. The Actions are executed upon satisfaction of local conditions, usually relating to the number and quality of available quantum resources (Bell pairs). In quantum networking, shared Bell pairs must be managed by each node in a coordinated fashion and appropriately structured RuleSets provide this required consistency in terms of quantum operations.

3.5 Simulator of Quantum Network Communication (SeQUeNCE)

The Simulator of Quantum Network Communication (SeQUeNCE)[31] is implemented in Python. There seems to be little in terms of novel abstraction, but rather an attempt to provide an accurate and precise discrete event simulation of selected protocols.

The role of the application module is abstracted to consume entanglement. Quantum network applications are heterogeneous with varying requirements, including throughput and fidelity, so there is a single application with random requests. A resource manager module manages local resources. Like QuISP, the resource manager uses an internal set of rules to manage quantum memories.

There is a focus on realism of quantum states such that quantum states can be tracked. States can be encoded as time bins, in the polarization of light, or as states in quantum memories. The quality of entanglement is a key quantum network performance metric, and loss and decoherence that affect it must be modeled. Bra–Ket and density matrix representations of quantum states are utilized. Hardware models record entanglement fidelity.

There is also a focus on realistic timing such that simulation events are executed at their respective timestamps maintaining their ordering to avoid causality errors. Quantum networks are time-sensitive systems; the arrival times of photons determine their identity. The lifetime of qubits in memories is limited, requiring time bounds on certain operations. The simulator operates with picosecond precision.

SeQUeNCe uses a modularized design that separates functionality into modules containing protocols that can be reprogrammed. Rapid testing of a large number of scenarios is made possible by representing network configuration in JSON.

It is also a focus to track quantum states at the individual photon level. In contrast to classical packet-level network simulations, photons are generated with megahertz frequencies, increasing the number of simulated events by several orders of magnitude. Preparation to support efficient parallel simulation methods is considered.

Quantum network architectures have not been standardized yet, and it is noted that there are often-conflicting quantum network designs. To allow simulation of alternative and emerging quantum network architectures, there was a choice to make minimal assumptions and identify the nascent architectural principles that are common to most quantum network designs. It is claimed that this enables a simplified, modularized quantum network design with five modules (*Hardware module, Entanglement management module, Resource management module, Network management module, Application module*) that allow for modeling many potential future network architectures. A sixth module, the simulation kernel, generates events.

[31] Xiaoliang Wu, et al., "SeQUeNCe: A Customizable Discrete-Event Simulator of Quantum Networks," *Quantum Sci. Technol*. 6 045027 (2021), doi:10.1088/2058-9565/ac22f6.

3.6 The NETwork Simulator for QUantum Information with Discrete events (NetSquid)

The NETwork Simulator for QUantum Information with Discrete events (NetSquid)[32] is a Python discrete event simulator.

The abstractions provided by NetSquid are a direct mapping to the major communication network and quantum network components.[33] The class component abstractions from the diagram are Qubit, KetState, DMState (density matrix), StabState (stabilizer), GSLCState (graph states with local Cliffords), QState, Operator, Entity, Event, EventHandler, EventExpression, Model, ErrorModel, DelayModel, Port, Component, QuantumMemory, Channel, Node, Network, Connection, ServiceProtocol, NodeProtocol, Protocol, and Data-Collector.

3.7 Quantum Key Distribution Network Simulation Module (QKDNetSim)

QKD is a commercialized quantum network technology that has reached a relatively high level of maturity. Quantum networking can learn much from QKD successes and, most importantly, its failures. There are many QKD simulators, both free and commercial. This report focuses on a free version, QKDNetSim,[34] implemented as an NS-3 simulator module. It assumed that readers are familiar with the classical NS-3 simulation package.

QKDNetSim focuses on the classical traffic generated by a QKD system. It relies heavily on the AIT R10 QKD software[35] developed on previously implemented projects. The simulator appears to follow the ETSI QKD standards and is a good example of a practical, useful tool. Its abstraction appears to focus on the classical channel while minimizing details of the quantum channel.

3.8 Quantum/classical

In this section, the quantum/classical[36] abstraction is not a simulator per se, but rather a proposed abstraction simulated with QuNetSim. This abstraction strives toward a tight integration of quantum and classical networking with an apparent goal of utilizing classical networks with minimal change. Using hybrid classical-quantum data frames, classical and quantum data can be transmitted in an interleaved fashion using the largely classical approaches. The quantum frame structure has three components: a classical header, a quantum payload, and a classical trailer. The concept is that the classical abstraction of a packet and packet switching can be maintained in a quantum network. The packets can be structured similar to Ethernet frames.

To generate such frames, a control unit triggers frame generation, the classical and quantum parts are multiplexed, and they pass through a quantum reconfigurable optical add/drop multiplexer (Q-ROADM).

A suggestion is that the concept of burst-switched quantum networks might be used in the near term. It requires precise classical/quantum transmission scheduling to avoid storing quantum states. Thus, burst switching is utilized to send control information ahead of the payload "just in time." Switching decisions

[32] T Coopmans, R Knegjens, A Dahlberg, et al. "NetSquid, a NETwork Simulator for QUantum Information Using Discrete Events. Commun Phys 4, 164 (2021), https://doi.org/10.1038/s42005-021-00647-8.

[33] For an example, see Figure 10, Abstractions provided by NetSquid, in T Coopmans, R Knegjens, A Dahlberg, et al. "NetSquid, A NETwork Simulator for QUantum Information Using Discrete Events," Commun Phys 4, 164 (2021), doi:10.1038/s42005-021-00647-8.

[34] M Mehic, O Maurhart, S Rass, et al. "Implementation of Quantum Key Distribution Network Simulation Module in the Network Simulator NS-3." Quantum Inf Process 16, 253 (2017), https://doi.org/10.1007/s11128-017-1702-z.

[35] git clone http://sqt.ait.ac.at/git/qkd-public.git

[36] S DiAdamo, "Integrating Quantum Simulation for Quantum-Enhanced Classical Network Emulation" (2021), arXiv:2110.01437.

are made before the quantum payload arrives. As quantum memory becomes more feasible, transitioning to packet switching becomes possible.

The hybrid frames are demultiplexed and split into classical and quantum components via an optical switch. The quantum signal can (optionally) travel to a memory, the classical signal to a processor. The processor releases the quantum signal and reconstructs the frame. The classical and quantum signals are multiplexed onto the output fiber. Dynamic switching alone cannot overcome the distance limitations of quantum state transmission.

At metropolitan scale, dynamic switching methods can be used. Dynamic switching can be combined with quantum repeaters for long distance applications. For inter-networking, dynamically switched subnetworks can be connected via a teleportation channel.

A quantum Ethernet frame and type-length-value (TLV) structures are proposed in support of this mechanism.

3.9 QuNET

QuNET[37] is a highly abstracted quantum network simulator written in Julia (see VI B 6. Network abstraction). It is open-source software for simulating entanglement networks and benchmarking entanglement routing algorithms.

QuNet claims a novel quantum cost-vector analysis to simulate and benchmark routing in multi-user entanglement networks in a way that is highly scalable. Because it is highly abstracted, it accommodates both ground-based and space-based networks and implements efficient multi-user time-optimization for mitigating congestion when quantum memories are available.

QuNet's abstraction focuses on benchmarking entanglement networks and entanglement routing algorithms.

The primary distinction of QuNet is that it considers a subset of error channels that are expressible in terms of additive cost metrics. This allows for complex quantum networks with a simple set of weighted graphs—one graph for each cost. QuNet therefore operates at a high level of abstraction as opposed to a low-level link layer where the control sequences of individual network components are implemented.

In using this approach, existing graph theoretic routing algorithms can be exploited, which are computationally efficient and well-studied. Furthermore, these techniques can be extended to accommodate for quantum memories using temporal meta-graphs. A limitation of this approach is that it does not lend itself to modeling arbitrary channels, in particular non-commutative ones that cannot be expressed as additive metrics.

The goal of QuNet is to perform cost-vector analysis in quantum networks, based on user demand, to make optimal routing decisions. Because QuNet is highly algorithmically efficient, it could be employed for overseeing quantum networks and making real-time routing decisions for the purposes of classical control of the network in the response to live user demand.

[37] H Leone, NR Miller, D Singh, NK Langford, and PP Rohde, "QuNet: Cost Vector Analysis & Multi-Path Entanglement Routing in Quantum Networks" (2021), arXiv:2105.00418.

3.10 NetQASM

QASM originated as a language for formally defining a quantum circuit to render images for visualization purposes. As quantum computation evolved, the language was adopted as a way to specify quantum circuits as input to a quantum computer. NetQASM[38] extends this to quantum networks and has similar goals to SimulaQron in terms of exploring the intersection of quantum processing and communication. NetQASM, another Python tool, is a low-level instruction set architecture for quantum internet applications. It is a universal, platform-independent and extendable instruction set with support for local quantum gates, classical logic, and quantum networking operations for remote entanglement generation. It enables exploration of close integration of classical logic and communication at the application layer with quantum operations at the physical layer. Its abstractions, as a result, are geared toward this goal.

NetQASM claims to enable quantum network applications to be programmed in high-level platform-independent software, which is not possible using any other QASM variant. The tools include a higher-level software development kit (SDK) in Python that allows ease in programming applications for a quantum internet. The SDK can be used with NetSquid and SimulaQron.

In this abstraction, a quantum network processing unit (QNPU) is assumed to reside within end-nodes in a quantum network. NetQASM is an instruction set architecture that can be used to run arbitrary programs for the QNPU on end-nodes.

The abstraction of the quantum network and its components includes quantum network applications that run on the end-nodes. Their communication via classical message passing and quantum entanglement is abstracted away by a network stack. That is, it is not visible at the application layer how entanglement generation or classical message passing is realized. This may be via direct physical connections, or intermediary repeaters and/or routers. End-nodes hold two types of qubits:

(1) communication qubits that can be used to generate entanglement with remote nodes
(2) storage qubits that can be used to store quantum states and apply operations

A communication qubit may also be used as a storage qubit. The qubits within an end-node can interact through quantum gates and their state can be measured.

4 Conclusion

Table 1 summarizes the abstractions discussed in this report. **Source** identifies the venue describing the abstraction, typically either a publication or standard; **abstraction** is an attempt at a very concise description of the "novel" abstraction(s); the **high-level components** column attempts to provide more detail about significant parts of the abstraction; and the **controls and observables**[39] are the control and observable parameters exposed by the abstraction, which is the focus of a follow-up report. While many standards and publications related to quantum networking continue to be generated (arguably more standards than in the quantum computing domain), a significant and overarching abstraction for quantum networking remains elusive (e.g., like layering for classical networking).

[38] A Dahlberg, "NetQASM-a Low-Level Instruction Set Architecture for Hybrid Quantum-Classical Programs in a Quantum Internet" 7:3 (2022), doi:10.1088/2058-9565/ac753f.

[39] Controls and observables are the focus of a companion report, which describes findings on quantum network parameters.

Some of the main technical gaps that could benefit from quantum network abstraction would enable architecting solutions that take account of time-sensitivity, quantum device interoperability, and transduction. The most beneficial abstractions will be those that expose unique quantum benefits.

Table 1: Summary of Abstractions

Source	Abstraction	High-level Components
"Tools for quantum network design"	• Entangled links • Noise modeling • Time-dependencies	• Entangled pairs as "links" • Entanglement swapping • Noise modeling via depolarizing, dephasing, or amplitude damping • Time-dependent memory decoherence
"Experimental demonstration of entanglement delivery using a quantum network stack"	• Platform-independent applications • Link layer abstraction of physical-layer entanglement • Remote state preparation and a hardware abstraction layer (HAL)	• Abstraction into independent tasks and services • Network controller implements the platform-independent stack • Link layer schedules entanglement requests • More stringent timing requirements are positioned lower in the software stack
"The Virtual Quantum Optics Laboratory"	• N/A	• N/A
IEEE P1913 YANG Model for Software-Defined Quantum Communication	• Software-Defined Networking	• Separation of classical control from quantum operation (YANG model) • Quantum Case & UML diagrams
Quantum Internet Research Group (qirg)	• Heralded entanglement abstraction • Software-defined networking • Quantum resource discovery in the network • Lower (faster) vs higher level (slower) signaling based upon timing requirements	• Three basic schemes for heralded entanglement generation (there is no upstream or downstream end of the pair) • Control information (distinct from control plane messages) messages that operate at the granularity of individual entangled pairs as part of quantum data plane • Types of quantum network nodes • Mechanism to discover and locate quantum resources in the network
Industry Specification Group (ISG) on Quantum Key Distribution (QKD)	• Secure key management abstractions	• Trusted vs quantum connections
ITU-T Focus Group on Quantum Information Technology for Networks (FG QIT4N)	• Layering • Software-defined network context	• Quantum, key management, QKDN control, QKDN management and service layers
Y.3800-Y.3999 Quantum key distribution networks	• Modular functional architecture diagram • AI/ML abstractions for QKD networks	• Abstraction for integration of different quantum technologies
GSMA Internet Group (IG)	• Quantum services network reference framework	• Quantum services network components
Quantum Internet Alliance	• Most of the QIA documents appear to be a re-packaging of U Delft papers and are covered elsewhere in this report	• Most of the QIA documents appear to be a re-packaging of U Delft papers and are covered elsewhere in this report
SimulaQron	• Quantum communication channels interconnect local quantum processors • Platform-independent portion of the quantum internet for applications	• "Stand-in" for platform dependent quantum processing systems • Classical Quantum Combiner (CQC) is an extended instruction set with special features tailored to a quantum network
Quantum NETwork SIMulator (QuNetSim)	• Focuses solely on the network layer (lower layers are separate "back end" software)	• Emphasizes synchronization logic programmatically at a high level
Simulator for Quantum Networks and CHannels (SQUANCH)	• Parallelized simulation of distributed quantum information processing Agents • Channels (quantum and classical) interconnect Agents • Error models apply to the transmitted information	• Agents • Channels (quantum and classical) • Error models • QSystem • QStreams • Qubits

	• Agents maintain internal clocks • Timing considering photon pulse widths, signal travel speeds, length of channels, etc. into account • QSystem represents a multi-particle quantum state as a density matrix • Gates manipulate the state of a quantum system	• Gates
Quantum Internet Simulator Package (QuISP)	• Error abstraction • QuISP Ruleset operation has the feel of OpenFlow flow tables	• Only deviations from the ideal quantum state are tracked and managed • Only Pauli X, Y, and Z errors as well as relaxation and excitation errors • Photon loss • Quantum state at any instant in time is represented by an error probability vector • Evolution of error probability vector is given by a transition rate matrix and modeled as a Markov process
Simulator of Quantum Network Communication (SeQUeNCE)	• Focus on realism of quantum states • Quality of entanglement is a key quantum network performance metric, and loss and decoherence that affect it • Quantum networks are time-sensitive systems; the arrival times of photons determine their identity • Bounds on certain operations • Modularized design that separates functionality into modules that contain protocols that can be reprogrammed • Arrival times of photons determine their identity	• States can be encoded as time bins, in the polarization of light, or as states in quantum memories • Bra–Ket and density matrix representations of quantum states are utilized • Hardware models record entanglement fidelity • Quantum networks are time-sensitive systems • Lifetime of qubits in memories is limited • Simulator operates with picosecond precision • Modularized design that separates functionality into modules that contain protocols that can be reprogrammed
NETwork Simulator for QUantum Information with Discrete events (NetSquid)	• Quantum and classical component-level class abstraction	• QState, Operator • Entity • Event, EventHandler, EventExpression • Model, ErrorModel, DelayModel • Port, Component, QuantumMemory • Channel, Node, Network • Connection • ServiceProtocol, NodeProtocol, Protocol • Data-Collector
QKDNetSim	• Conforms to ETSI QKD standards • Focus on classical channel; minimizes details of quantum channel	• Relies heavily on AIT R10 QKD software for quantum impact on classical traffic • Conforms to ETSI QKD standards
"Integrating Quantum Simulation for Quantum-Enhanced Classical Network Emulation"	• Abstraction strives toward tight integration of quantum and classical networking • Hybrid classical-quantum data frames: classical header, quantum payload, classical trailer • Quantum Ethernet frame and type-length-value (TLV) structures	• Classical abstraction of a packet and packet switching can be maintained in a quantum network • Concept of burst-switched quantum networks - sends control information ahead of the payload "just in time" • Requires precise classical/quantum transmission scheduling to avoid storing quantum states
QuNET	• Quantum cost-vector analysis to simulate and benchmark routing in multi-user entanglement networks in a way that is highly scalable • Abstraction considers a subset of error channels that are expressible in terms of additive cost metrics	• Abstraction considers a subset of error channels that are expressible in terms of additive cost metrics • Operates at a high level of abstraction as opposed to representing details of a low-level link layer
NetQASM	• Universal, platform-independent and extendable instruction set with support for local quantum gates, classical logic, and quantum networking operations for remote entanglement generation	• Low-level instruction set architecture for quantum internet applications • Close integration of classical logic and communication at the application layer with quantum operations at the physical layer

	• Quantum network applications to be programmed in high-level platform-independent software • Quantum network processing unit (QNPU) is assumed to reside within end-nodes in a quantum network • Quantum network applications run on the end-nodes; communication via classical message passing and quantum entanglement is abstracted away by a network stack	• End-nodes hold: (1) communication qubits which can be used to generate entanglement with remote nodes and (2) storage qubits which can be used to store quantum states and apply operations