IARJSET

ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 8, August 2024

DOI: 10.17148/IARJSET.2024.11827

A Hybrid Reference Architecture Enabling Quantum Computing Capabilities for Cloud Utilization: Toward a Quantum-Science Gateway

Shravya Bhat[1], Shri Ranjani S M[2], Milan Srinivas[3], Vinay Kumar[4]

BE, Department of AI and ML, Nitte Meenakshi Institute of Technology, Bangalore, India[1]

BE, Department of AI and ML, BNM Institute of Technology, Bangalore, India[2]

BE, Department of Computer Science and Engineering, VKIT, Bangalore, India[3]

BE, Department of AI and ML, BNM Institute of Technology, Bangalore, India[4]

Abstract: The increasing accessibility of quantum computing resources encourages research into the potential applications of this technology in a variety of scientific fields, including artificial intelligence, manufacturing, and finance.

While a large number of research scientists do their work largely using cloud computing infrastructures, access to real (sometimes distant) quantum hardware resources necessitates the deployment and appropriate setup of several software components. We provide a hybrid cloud-based reference architecture in this study that makes it easier to launch new experiments utilizing a variety of quantum computing resources. The technique makes it easier to access many remote quantum compute resources and to run distributed quantum computing simulations in conventional cloud settings. Because the reference design is so adaptable to many cloud platforms, there are many opportunities for applications.

Keywords: Cloud computing, quantum computing, reference architecture, simulation, science gateway.

I. INTRODUCTION

As end consumers may increasingly get near-term quantum computers [2], [3], [4], [5], [6], there is a clear need to make them easier to use for interested researchers to the greatest extent feasible by reducing the entry obstacle. Resources for quantum computing are still complex,to utilize with their apps, algorithms, and user interfaces are not even close to reaching the point of abstraction where they are simple to utilize for scholars in different domains than quantum specialists. Typically, in order to use these resources, a variety of development frameworks must be deployed, such have all the required instruments [7],[8].Although this may seem like a simple procedure, the workload for interested academics also grows as there are more of these frameworks available: the documentation for the various installation and configuration.

As a result, there is a glaring disconnect between quantum computing as a technique and its actual potential applications. This acts as a roadblock to the use of quantum computing as well as a disappointment for the advancement of the technology known as quantum computing, since as as long as the user base and actual application outcomes are still restricted, and technological advancement cannot proceed more quickly. Our objective is to construct a gateway that dramatically reduces the entrance barrier for scientists who want to study and utilize quantum resources to implement quantum computing. more readily and simply available algorithms and applications,launching from a familiar setting. Generally speaking, science gateways are online resources that give researchers. Given easy access to data and computational resources.

II. STATE OF THE ART

One workable way to make quantum computing broadly available is to use the consumption models that commercial clouds have proposed to access quantum computing resources [9]. As a result, tasks can be completed without the need to purchase and manage physical hardware thanks to Quantum Cloud Computing (QCC) or Quantum Computing as a Service (QCaaS) [10], [11], [12]. Moreover, utilizing the best practices and current knowledge included in cloud reference architectures may make it possible to develop brand-new, best-of-breed solutions that combine the advantages of quantum and cloud computing.

IARJSET
ISSN (O) 2393-8021, ISSN (P) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 8, August 2024
DOI: 10.17148/IARJSET.2024.11827

There is a vast body of academic literature available in the quickly developing fields of quantum computing, cloud computing, and reference architectures. Nonetheless, in order to maintain the focus of similar works, we are only examining them in three specific areas in this part that are closely related to our work. We start by talking about (i) reference architectures in general. We then concentrate on (ii) their application in cloud computing. Ultimately, we address (iii) quantum resources concerning accessible cloud-enabled products that can be integrated into a model architecture. It should be noted that we connect our answer in Section III to the related work principles that have been discussed.

A. ONLINE RESEARCH AND CLOUD COMPUTING

Software architecture plans enable the creation of new solutions by repurposing best practices and existing knowledge. A variety of definitions have been offered, for example in [13], [14], [15], and [16], with the major goals being to: (i) support reusability; (ii) integrate best practices; (iii) employ high or low abstraction levels; and (iv) cater to specific use cases. High abstraction level architectures, in general, usually include methodologies and principles of system design but do not include specific implementation references. In contrast, low-level designs concentrate on the specifics of implementation; in cloud computing, these details usually involve utilizing the software and platform services provided by a specific cloud provider. Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS) are the terms used to describe these kinds of services.

Three cloud computing service models are outlined in the NIST definition of cloud computing [20]. The following service models are layered with a top-down perspective: In Software as a Service (SaaS), for example, users receive applications (like web-based email) and the provider manages them on its infrastructure; in Platform as a Service (PaaS), users can deploy applications and services on the cloud without having to provision and manage the hosting resources; and in Infrastructure as a Service (IaaS), users can provision compute, storage, network, etc. without having to worry about the underlying cloud infrastructure.

Providers of public clouds give a variety of reference designs. All of these low-level designs are meant to help companies commercialize their cloud services. For instance, AWS provides 64 reference architectures for big data and data analytics [29] and Microsoft Azure offers 118 architecture blueprints for data analytics [28]. Conversely, the "Industrial Internet Reference Architecture" [30] and the "Reference Architectural Model Industry 4.0" [31] are high-level concepts that reflect the collaborative work of multiple organizations and firms.

B. QUANTUM RESOURCES

The works showcased aim to address the growing accessibility of quantum computing resources. Similar to cloud systems, there are various categories into which quantum resources can be categorized. These categories may include the kind of computing model that is supported, the technology employed to implement qubits, or the accessibility of the resources.

A taxonomy for quantum computing is introduced along with a review of the literature on the subject in [36]. Several connected papers are categorized, analyzed, and research areas are highlighted for more study using this taxonomy. By examining quantum software tools and technologies, post-quantum cryptography, and developments in quantum computer hardware, the writers provide the state-of-the-art in quantum computing as of right now.

One optimization-focused method for determining the lowest values of an objective function that represents a problem is the quantum annealing model. Because of this, while using the annealing model, the user must express the problem as an objective function, with the optimal solution being represented as the objective function's minimum value. Mapping the correctly described function into a physical system, operating the system in accordance with quantum mechanics, and obtaining system samples are the tasks assigned to the quantum hardware.

Ocean SDK [44] is the name of the application development framework for D-Wave resources. This SDK, which is based on Python, provides a wide range of features for formulating issues for the D-Wave hardware. Additionally, it offers samplers and solvers for handling the given issues locally, on a QPU, or in Leap's cloud infrastructure, utilizing a hybrid approach that combines QPUs and traditional resources. Third-party IDEs, other than the Ocean SDK, are often supported locally and in the cloud. The specified Development Containers specification must be implemented by the IDEs. D-Wave suggests using a local IDE such as VS Code or GitHub Codespaces. Leap IDE, which was formerly accessible, has been replaced by this support.A summary of these providers with their most recent QPU capability and supported SDKs is provided in Table 1. The vast majority of quantum hardware providers implement the quantum circuit model with Superconducting architecture, such as Google Quantum AI [45], IBM Quantum [46], IQM [47], Oxford Quantum Circuits [48], and Rigetti [49], [50].

IARJSET

ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 8, August 2024

DOI: 10.17148/IARJSET.2024.11827

Some providers use a Trapped ion architecture like AQT [51], IonQ [4], and Quantinuum [52]. Lastly, some hardware providers use an Analog quantum approach with Neutral atoms architecture, like Pasqal [53] and QuEra [54], [55].

For example, Amazon Braket [56] serves as a proxy provider to access QPU devices from Oxford Quantum Circuits, Rigetti, and IonQ. Amazon Braket SDK is available to aid in development. Through its managed SageMaker notebook service, Amazon Web Services (AWS) also offers an interactive development environment akin to IBM Quantum Labs [57]. There are several simulators offered by AWS, each with unique features and costs. For a limited while, users can use the simulators without charge under the free tier; after that, they will be charged for any additional usage. QPU device usage is also charged.

Azure Quantum allows connectivity to QPU devices via a public cloud provider interface, which is a service akin to Amazon Braket. Among the hardware providers in the group are Pasqal, Rigetti, and IonQ. Cirq, Qiskit, and the Azure Quantum Development Kit provide access to Azure Quantum resources. For creating quantum programs, the later one additionally provides the high-level Q# programming language.

In conclusion, we can see strategies for granting access to quantum computing resources at both higher and lower levels. Higher level abstract approaches like iQuantum [10] try to provide a system model for contexts involving hybrid quantum computing, which includes brokering. These high-level methods, nonetheless, usually concentrate on modeling and simulation. On the other hand, many vendor-specific quantum computing libraries and meta-libraries exist at the lower level (e.g., Qiskit, Amazon Braket [56],

TABLE 1. Quantum resources: quantum processing unit providers.

Provider	Type	Architecture	Latest QPU	No. of Qubits	Release Date	SDK Support	Ref. Arch. Support
D-Wave	Quantum annealing	Superconducting	Advantage	5000+	2021	Ocean SDK	Yes
IBM Quantum	Circuit-based	Superconducting	Osprey	433	2022 Nov	Qiskit, CQ tket, QCW Forge, ProjectQ, PennyLane	Yes
IQM	Circuit-based	Superconducting	N/A	20	2023 Oct	Qiskit, CQ tket, ProjectQ, PennyLane,TensorFlow Quantum	Yes
OxfordQuantum Circuits	Circuit-based	Superconducting	Lucy	8	2022 Feb	Amazon Braket SDK, Qiskit, CQ tket	Yes
Rigetti	Circuit-based	Superconducting	ASPEN-M-3	80	2022 Dec	PyQuil, Cirq, Qiskit, CQ tket, Q#, QCW Forge, ProjectQ, PennyLane, TensorFlow Quantum	Yes
AQT	Circuit-based	Trapped ion	PINE system	24	2021 June	Qiskit, Cirq, CQ tket, ProjectQ, PennyLane, TensorFlow Quantum	Yes
IonQ	Circuit-based	Trapped ion	Forte	32	2022	Qiskit, Cirq, CQ tket, Q#, QCW Forge, ProjectQ, PennyLane, TensorFlow Quantum	Yes
Quantinuum	Circuit-based	Trapped ion	H2	32	2023 May	Qiskit, Cirq, CQ tket, Q#, PennyLane, TensorFlow Quantum	Yes
Pasqal	Analog quantum	Neutral atoms	Gen1	100	2022 May	Cirq, PennyLane, Pulser, TensorFlow Quantum	Yes
QuEra	Analog quantum	Neutral atoms	Aquila	256	2022 Nov	Amazon Braket SDK, Qiskit, CQ tket	Yes

TABLE 2. Quantum resources: cloud service providers.

Cloud service	QPC Support	SDK Support	Free Tier	Ref. Arch. Support
Amazon Braket	OQC, Rigetti, IonQ, QuEra	Amazon Braket SDK, Qiskit, PennyLane	1 hour free simulation time / month	Yes
Azure Quantum	Rigetti, IonQ, Quantinuum	Qiskit, Cirq, Q#	$500 USD free credit for 6 months	Yes
QC Ware	IBM Quantum, Rigetti, IonQ	QC Ware Forge	Free 1 min QCT	No
D-Wave Leap	D-Wave Advantage	Ocean SDK	Free 1 min QPU access per month	Yes
IBM Quantum Platform	IBM Quantum	Qiskit, CQ tket, QCW Forge, ProjectQ, PennyLane	Free 10 mins QPU access per month	Yes
PASQAL Cloud Services	Pasqal	Cirq, PennyLane, Pulser, TensorFlow Quantum	N/A	Yes
QuEra Cloud	QuEra	Amazon Braket SDK, Qiskit, PennyLane	Amazon Free Tier	Yes

IARJSET

ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 8, August 2024

DOI: 10.17148/IARJSET.2024.11827

III. REFERENCE ARCHITECTURE DESIGN

The quantum reference architecture's design is shown in this section. The requirements that served as the basis for the work's beginning are discussed, along with the elements that have been chosen to take part in the architecture reference,and the way it is put into practice.The reason behind the creation of the reference architecture represented the growing accessibility of quantum computer power. As it was demonstrated in the earlier section, other implementations of QPU devices are now accessible for trial use. So, it was natural to look at if they may be beneficial for scientific domains that the HUN-REN Cloud users represent.Since D-Wave resources were readily available and free of cost within a specified resource consumption quota, we began experimenting with them. It turns out that D-Wave's Ocean SDK is rather simple to use, and several examples have already been given, demonstrating the usefulness of their quantum devices in a variety of application domains. We have chosen a few cases from the list and begun analyzing them thoroughly. From there, we have also developed some original examples.

TABLE 3. Quantum computing tooling: SDKs and environments.

Tool Name	Type	Supported Programming Languages	Description	Supported Providers	License	Ref. Arch. Support
Qiskit	SDK	Python	"An open-source SDK for working with quantum computers at the level of pulses, circuits, and algorithms."	IBM Quantum, IQM, OQC, Rigetti, AQT, IonQ, Quantinuum, QuEra	Apache 2.0	Yes
IBM Quantum Learning	Learning Environment	Python (JupyterLab)	Hosted environment for experimenting with quantum computing resources.	IBM Quantum	Proprietary	No
Amazon Braket SDK	SDK	Python	"A Python SDK for interacting with quantum devices on Amazon Braket."	Amazon Braket (OQC, Rigetti, QuEra and IonQ)	Apache 2.0	Yes
Cirq (Google)	SDK	Python	"A python framework for creating, editing, and invoking Noisy Intermediate Scale Quantum (NISQ) circuits."	Rigetti, AQT, IonQ, Quantinuum, Pasqal	Apache 2.0	Yes
Azure Quantum Development Kit	SDK	Q#, Python (qdk-python)	The required SDK to interface with the Azure Quantum service.	Rigetti, IonQ, Quantinuum	MIT	No
pyQuil	SDK	Python	Build and execute Quil programs using Python	Rigetti Quantum Cloud Services	Apache 2.0	Planned
Ocean SDK	SDK	Python	"Ocean is D-Wave's suite of tools for solving hard problems with quantum computers."	D-Wave	Apache 2.0	Yes
CQ tket	SDK	Python	"pytket is a python module for interfacing with tket, a quantum computing toolkit and optimising compiler developed by Quantinuum."	IBM Quantum, IQM, Rigetti, AQT, IonQ, Quantinuum, Amazon Braket SDK, Azure Quantum	Apache 2.0	No
QCW Forge	SDK	Python	"Forge is a service that lets you develop and run quantum software. An interface to this service is exposed to Python developers via the Forge client library."	IBM Quantum, Rigetti, IonQ	Proprietary	No
ProjectQ	SDK	Python	"ProjectQ is an open source effort for quantum computing."	IBM Quantum, AQT, IonQ, Amazon Braket SDK, Azure Quantum	Apache 2.0	No
TensorFlow Quantum	SDK	Python	"TensorFlow Quantum (TFQ) is a quantum machine learning library for rapid prototyping of hybrid quantum-classical ML models."	Rigetti, AQT, IonQ, Quantinuum, Pasqal	Apache 2.0	No
PennyLane	SDK	Python	"[..] a cross-platform Python library for programming quantum computers. [..] connects quantum computing with powerful machine learning frameworks."	Qiskit, Amazon Bracket, Cirq, Microsoft QDK, Honeywell, IonQ, AQT, Rigetti Forest, Qualcs, Orquestra, Quantum Inspire, ProjectQ and Strawberry Fields	Apache 2.0	Yes

Our goal was to immediately enable the HUN-REN Cloud researchers to begin exploring with quantum resources by making the results publicly available.

We established several criteria, including the following: (i) the published solution must contain all the parts required to begin creating quantum applications right away; (ii) it must contain several examples showing how to use quantum resources; and (iii) it must be simple to extend to accommodate more quantum resources.

A. SELECTED COMPONENTS

Our goal was to immediately enable the HUN-REN Cloud researchers to begin exploring with quantum resources by making the results publicly available. We established several criteria, including the following: (i) the published solution must contain all the parts required to begin creating quantum applications right away; (ii) it must contain several examples

IARJSET
ISSN (O) 2393-8021, ISSN (P) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 8, August 2024
DOI: 10.17148/IARJSET.2024.11827

showing how to use quantum resources; and (iii) it must be simple to extend to accommodate more quantum resources.Because JupyterLab is a de facto standard among scientists, has a large feature set for notebook functionality, is easy to deploy in a variety of environments, integrates well with a variety of tools (like Matplotlib), and doesn't require external services, we chose it for the gateway role. We also wanted to offer some examples that demonstrate how quantum computers may be used to solve certain "real-life" problems. We have chosen the D-Wave clustering issue illustration for this. Moreover, Apache Spark's machine learning framework and the K-mean technique may be used to overcome this issue.

The purpose of this presentation is to demonstrate that not only can Apache Spark be used to tackle such a problem, but that quantum resources may be employed as well.

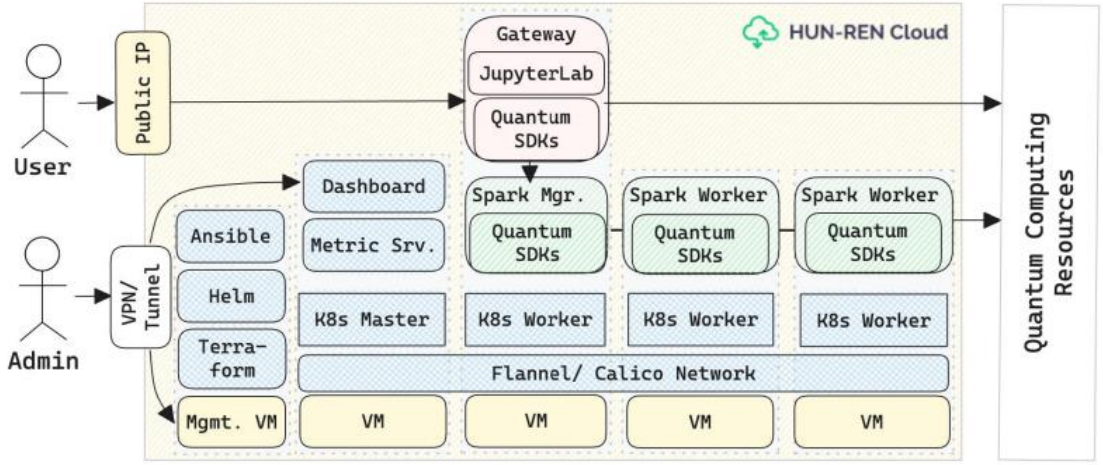

FIGURE 1. The Quantum - Science Gateway Hybrid Reference Architecture running on HUN-REN Cloud (components provided by the cloud are marked with yellow) utilizing the (i) Kubernetes Reference Architecture [64] (related components marked with blue grid); and extending the (ii) Apache Spark Reference Architecture [33] (related components marked with green stripes).

B. REFERENCE ARCHITECTURE IMPLEMENTATION

With the possibility to directly install the components on the hosts, the standard architecture is built on software containers if required. To expedite the deployment process and offer portability, we advise a coordinated implementation of either through Docker Compose or the reference architecture Docker. The photos listed below are used to deploy the Science Gateway picture (1) is the reference architecture,comprising examples, SDKs, and JupyterLab; (2) the Apache picture of the Spark Master; and (3) one or more instances of the picture for the worker using Apache Spark. The uppermost element Figure 2 displays the reference architecture's structure.

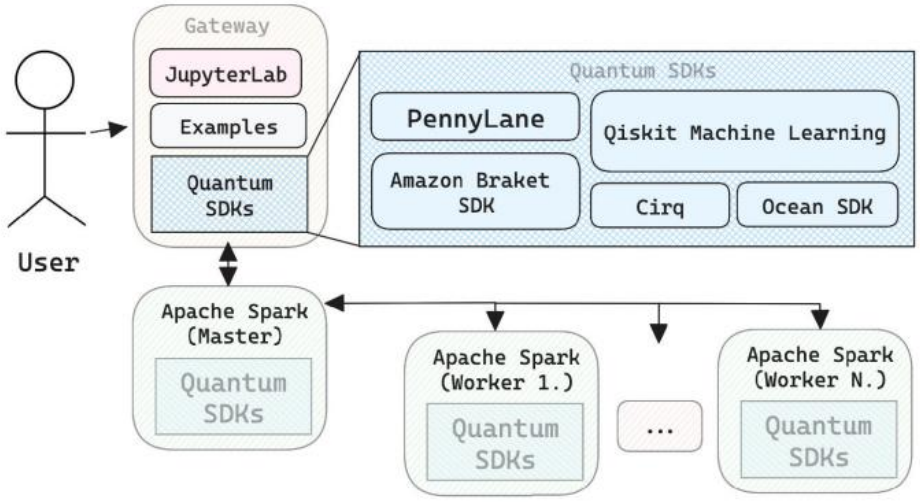

FIGURE 2. Component structure of the quantum reference architecture.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 8, August 2024

DOI: 10.17148/IARJSET.2024.11827

1) A compact base image, such as a Debian image with the bare minimum. In this instance, all of the other components—such as JupyterLab and Spark libraries—must be deployed.

2) A particular base image with a predetermined set of installed requirements (such as Java for Spark or JupyterLab itself).Since Java is needed to execute the examples using Apache Spark, the OpenJDK Docker image (openjdk:8-jdk-slim) was chosen as the foundation image. The image's deployment processes are outlined in a Dockerfile; these include installing JupyterLab, installing Python packages, installing quantum SDKs, deploying sample notebooks, and deploying the startup script. The starting script's duties include setting the password (depending on the environment) and creating the default Jupyter configuration.

C. COMPONENT COMPOSITION

The container pictures that represent the various components are now ready, and the following step requires them to be planned. There are several orchestration choices.For example, Kubernetes, Docker (via Docker Compose or Swarm),or (via Terraform) virtual computers. Docker is a software application,a simple tool to display individual containers, however more effort is needed to bring up several containers.

Which are able to speak with one another. Docker Construct is a simple-to-use instrument that raises the composition of many containers, providing replication in addition. Additionally, it permits users to install the Docker service stack swarm group. On the other hand, Kubernetes is a popular instrument for monitoring massive service stacks made with many container instances, however it calls for significant resources to use, as well as a lengthy operation.

D. AVAILABILITY OF THE REFERENCE ARCHITECTURE

The produced reference architecture has its own Git repository [67], hosted on a GitLab deployment, and is accessible via the HUN-REN Cloud interface [66]. This storehouse contains all the code required to duplicate the components pictures. Furthermore, GitLab's CI/CD functionality is used to create and release updated versions of the pictures.

The code has been modified. Additionally, the Git repository has brief instructions for using the reference architecture in the documentation within the Cloud HUN-REN. Moreover, a page on Wikipedia [68] is consistently updated with instructions on how to gain access to the many suppliers of quantum gear resources.

IV. DEPLOYMENT AND USABILITY EXPERIENCES

This section looks at the reference architecture's deployment options in different scenarios and assesses how usable the deployment is.Both single node and orchestrated deployments are supported by the reference design. Docker Compose is advised for deployments using a single machine. On any host or virtual machine, though, the reference architecture can also be deployed directly.

Additionally, Kubernetes can be deployed on a single node using programs like Minikube. Kubernetes is the suggested option for multi-node or coordinated deployments. HUN-REN Cloud can build a new cluster using the Kubernetes Reference Architecture [64] or utilize an already-existing Kubernetes cluster, including hosted services like Azure Kubernetes Service (see Figure 1). The latter is illustrated in more detail in Figure 1.

A. LOCAL IMPLEMENTATION

In this case, the user is deploying the reference architecture on its own computer. In this situation, it is assumed that the user is running Docker Compose and Engine on a recent version of Linux. In this instance, the steps to launch the reference architecture are as follows:

The user needs to do two things: 1) clone the reference architecture's Git repository; and 2) launch the desired version (either the full compose file for the complete version, or the default compose file for the basic version).

Using the password specified in the docker-compose.yml file, the interface may be accessed via HTTP on port 8888/TCP with this straightforward deployment technique that just runs the gateway component (JupyterLab).

IARJSET
ISSN (O) 2393-8021, ISSN (P) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 8, August 2024
DOI: 10.17148/IARJSET.2024.11827

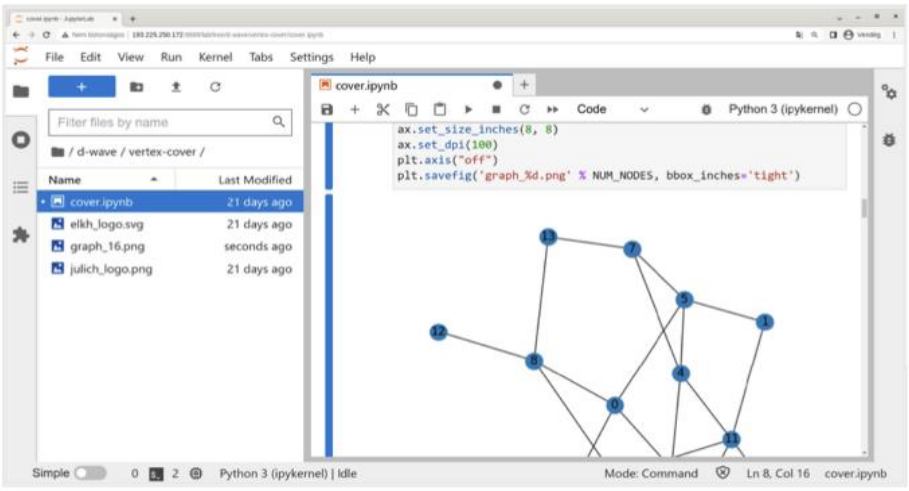

FIGURE 3. The Gateway (JupyterLab) of the reference architecture running in HUN-REN Cloud.

B. CLOUD-BASED DEPLOYMENTS

As the reference architecture aims at offering a quick-start package for experimenting with quantum resources for HUN-REN Cloud users, it is natural to check the deployment of the reference architecture in HUN-REN CloudRegardless of the deployment mechanism (Docker single host, Docker multi-host, or Kubernetes), the expanded Spark Reference Architecture leverages the Standalone cluster manager type [70] to ensure consistent behavior across all deployments.
The user can install the Kubernetes cluster on HUN-REN Cloud by following the comprehensive guidelines. With the aid of the supplied docker-compose.yml file, the Kompose tool, and the Kubernetes deployment files, they can quickly deploy the Quantum reference architecture on top of the Kubernetes cluster following the deployment. Following deployment, the user can utilize the 8888 port to contact the JupyterLab interface via the Kuberetes

C. APPLICABILITY STORIES

Lastly, a deployment of the reference architecture was put to the test to see how well it worked. The utility of the aforementioned frameworks and their relationship to the following quantum resources were evaluated in this scenario: Amazon Braket (with IonQ), IBM Quantum using Qiskit, and D-Wave (with the example Vertex Cover issue)The 03-Execution notebook, a prepared Qiskit example included in the reference design, was used to evaluate IBM Quantum Access. In this example, four qubits are prepared for a quantum circuit, they are each given a Hadamard gate, and then measurements are taken. Since every qubit is in an equal superposition state, random values between 0 and 15 should be returned by the measurements.

The example uses the cloud-based simulator_stabilizer, the local aer_simulator, and finally the real QPU device, ibm_oslo, to perform the circuit. The circuit was ran utilizing multiple shots in each example. Although the example was carried out correctly, there was a roughly 10-second wait in line for the QPU device executions.The last test case looks at the D-Wave resources' use and some benchmarking tests using an example of the minimal vertex cover problem. It looks at how long it takes to calculate the minimum vertex cover of Erdős-Rényi graphs with an increasing number of nodes.

The problem is solved in the example using the QPU-backed DWaveSampler and the local (CPU-based) ExactSolver. The example concludes with a diagram showing the processing times needed for the various devices. Figure 5 displays an example result for 10..35 nodes. It is evident that the processing time increases exponentially with the number of nodes when local resources are used. Utilizing the QPU devices takes essentially no time at all.

IARJSET

ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 8, August 2024

DOI: 10.17148/IARJSET.2024.11827

TABLE 4. Measurements of the Deutsch-Jozsa circuit using the IonQ [4] QPU through Amazon braket.

Case	1110	1111	1100	1010	1000	0110
Measurement Count	48	46	2	2	1	1

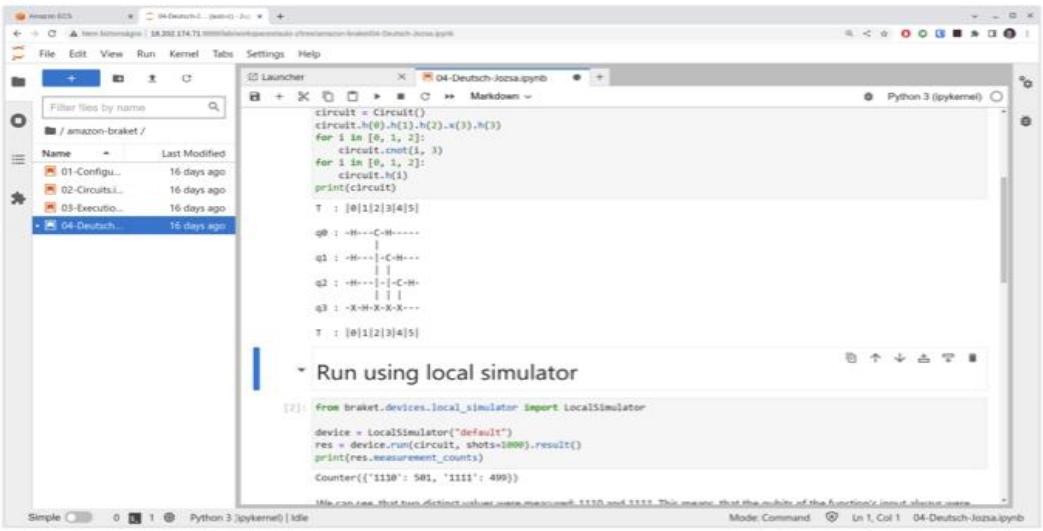

FIGURE 4. The gateway (JupyterLab) of the reference architecture running in Amazon Web services.

V. FUTURE WORK

The creation of the reference architecture is a crucial step toward building a quantum-science gateway that will greatly facilitate and increase the accessibility of quantum computing for scientists wishing to employ this extremely sophisticated instrument in their own studies. Taking into account that both quantumIt is crucial that potential users may quickly evaluate and contrast the options since hardware and quantum algorithms and applications are new and fast developing experimental domains. In many situations, alternative implementations support the jobs with differing efficiency.

This is made possible by the reference architecture mentioned above, which offers a single, open, portable, and cost-free environment from which a variety of quantum hardware and simulators may be accessed in a comparable manner.The standard architecture is adaptable enough to be implemented in a range of settings, including local settings, public cloud providers like Amazon Web Services, and community clouds like HUN-REN Cloud. This is made feasible through the use of standardized orchestration techniques and a modular software container-based architecture.

JupyterHub is also taken into consideration for multi-user support when it comes to the gateway. As of right now, the JupyterLab component only supports a single user mode; therefore, in order to support multiple users, several deployments of the reference design are needed. But using JupyterHub.

IARJSET
ISSN (O) 2393-8021, ISSN (P) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 8, August 2024
DOI: 10.17148/IARJSET.2024.11827

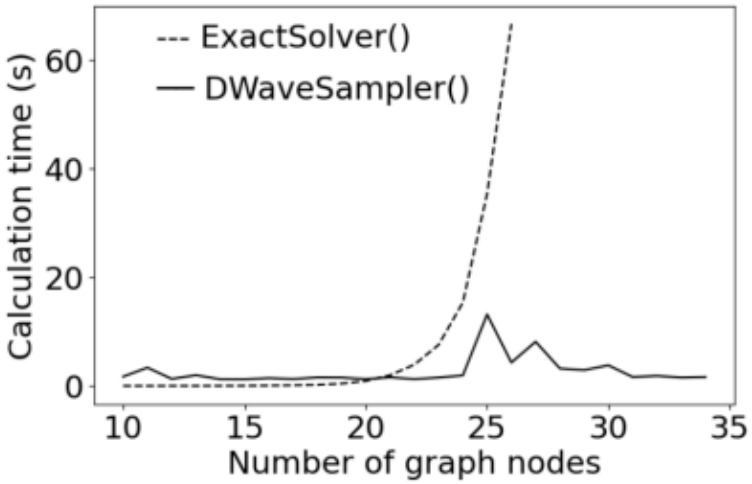

FIGURE 5. CPU and QPU processing times for the vertex cover problem.

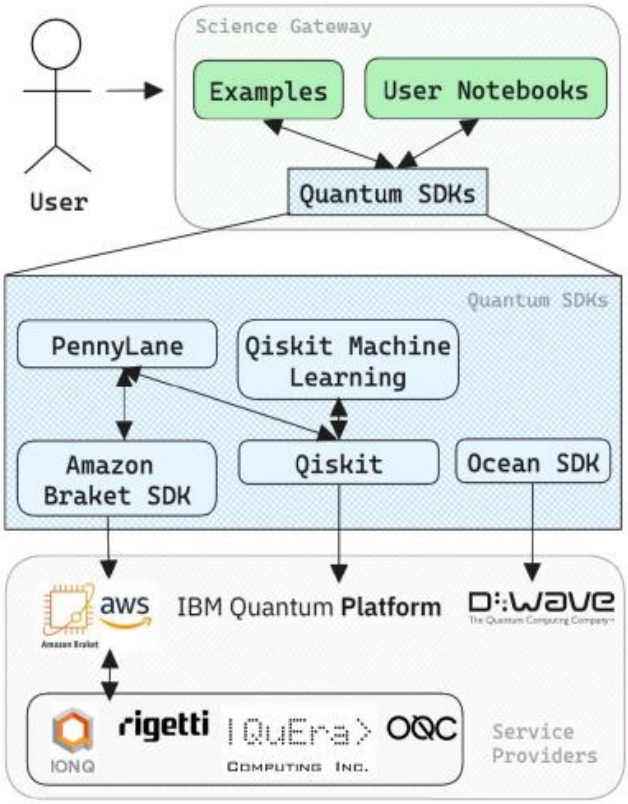

FIGURE 6. QPU access possibilities in the hybrid reference architecture.

The standard architecture is adaptable enough to be implemented in a range of settings, including local settings, public cloud providers like Amazon Web Services, and community clouds like HUN-REN Cloud.

This is made feasible through the use of standardized orchestration techniques and a modular software container-based architecture.

JupyterHub is also taken into consideration for multi-user support when it comes to the gateway. As of right now, the JupyterLab component only supports a single user mode; therefore, in order to support multiple users, several deployments of the reference design are needed. But using JupyterHub.

IARJSET

ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 8, August 2024

DOI: 10.17148/IARJSET.2024.11827

VI. CONCLUSION

We introduced a unique method in this research for making a wide range of genuine quantum resources accessible. Major quantum software development kits (SDKs) and frameworks, including those from IBM, D-Wave, and IonQ, as well as quantum hardware, including the Rigetti QPU and IonQ devices offered by Amazon Braket, are all included in our hybrid reference architecture.

Additionally, Qiskit Machine Learning and PennyLane, two frameworks that facilitate quantum resource experiments in machine learning, are included in the design. Figures 1, 2, and 6 provide a graphic summary of the architecture's substance. Additionally, we offer a number of illustrations and early benchmark findings, which may be used as useful manuals for formulating solutions to additional (predefined) issues.

REFERENCES

[1] M. Héder, E. Rigó, D. Medgyesi, R. Lovas, S. Tenczer, F. Török, A. Farkas, M. Emodi, J. Kadlecsik, Á. Pintér, and P. Kacsuk, ‘‘The past, present and future of the ELKH cloud,’’ Információs Társadalom, vol. 22, no. 2, p. 128, Aug. 2022.

[2] P. I. Bunyk, E. M. Hoskinson, M. W. Johnson, E. Tolkacheva, F. Altomare, A. J. Berkley, R. Harris, J. P. Hilton, T. Lanting, A. J. Przybysz, and J. Whittaker, ‘‘Architectural considerations in the design of a superconducting quantum annealing processor,’’ IEEE Trans. Appl. Supercond., vol. 24, no. 4, pp. 1–10, Aug. 2014.

[3] K. Boothby, C. Enderud, T. Lanting, R. Molavi, N. Tsai, M. H. Volkmann, F. Altomare, M. H. Amin, M. Babcock, A. J. Berkley, and C. B. Aznar, ‘‘Architectural considerations in the design of a third-generation superconducting quantum annealing processor,’’ 2021, arXiv:2108.02322.

[4] S. Debnath, N. M. Linke, C. Figgatt, K. A. Landsman, K. Wright, and C. Monroe, ‘‘Demonstration of a small programmable quantum computer with atomic qubits,’’ Nature, vol. 536, no. 7614, pp. 63–66, Aug. 2016.

[5] J. M. Amini, Y. Nam, N. Grzesiak, J.-S. Chen, N. C. Pisenti, M. Chmielewski, and C. Collins, ‘‘Benchmarking an 11-qubit quantum computer,’’ Nature Commun., vol. 10, no. 1, p. 5464, 2019.

[6] Y. Alexeev, D. Bacon, K. R. Brown, R. Calderbank, L. D. Carr, F. T. Chong, B. DeMarco, D. Englund, E. Farhi, B. Fefferman, and A. V. Gorshkov, ‘‘Quantum computer systems for scientific discovery,’’ PRX Quantum, vol. 2, no. 1, Feb. 2021, Art. no. 017001.

[7] V. Hassija, V. Chamola, V. Saxena, V. Chanana, P. Parashari, S. Mumtaz, and M. Guizani, ‘‘Present landscape of quantum computing,’’ IET Quantum Commun., vol. 1, no. 2, pp. 42–48, Dec. 2020.

[8] N. Killoran, J. Izaac, N. Quesada, V. Bergholm, M. Amy, and C. Weedbrook, ‘‘Strawberry fields: A software platform for photonic quantum computing,’’ Quantum, vol. 3, p. 129, Mar. 2019.

[9] S. J. Devitt, ‘‘Performing quantum computing experiments in the cloud,’’ Phys. Rev. A, Gen. Phys., vol. 94, no. 3, Sep. 2016, Art. no. 032329.

[10] H. T. Nguyen, M. Usman, and R. Buyya, ‘‘IQuantum: A case for modeling and simulation of quantum computing environments,’’ 2023, arXiv:2303.15729.

[11] F. Leymann, J. Barzen, M. Falkenthal, D. Vietz, B. Weder, and K. Wild, ‘‘Quantum in the cloud: Application potentials and research opportunities,’’ 2020, arXiv:2003.06256.

[12] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, ‘‘A quantum engineer’s guide to superconducting qubits,’’ Appl. Phys. Rev., vol. 6, no. 2, Jun. 2019, Art. no. 021318.

[13] What is a Reference Architecture—Enterprise it Definitions. Accessed: Jul. 23, 2023. [Online]. Available: https://www.hpe.com/us/en/whatis/reference-architecture.html

[14] P. Pääkkönen and D. Pakkala, ‘‘Reference architecture and classification of technologies, products and services for big data systems,’’ Big Data Res., vol. 2, no. 4, pp. 166–186, Dec. 2015.

[15] Microsoft Azure Documentation—Reference Architectures. Accessed: Jul. 23, 2023. [Online]. Available: https://docs.microsoft.com/enus/azure/architecture/browse/

[16] The TOGAF Standard, Version 9.2 Overview. Accessed: Feb. 12, 2022. [Online]. Available: https://www.opengroup.org/togaf

[17] R. Cloutier, G. Müller, D. Verma, R. Nilchiani, E. Hole, and M. Bone, ‘‘The concept of reference architectures,’’ Syst. Eng., vol. 13, no. 1, pp. 14–27, Mar. 2010.

[18] M. Weyrich and C. Ebert, ‘‘Reference architectures for the Internet of Things,’’ IEEE Softw., vol. 33, no. 1, pp. 112–116, Jan. 2016.

[19] A. C. Marosi, M. Emodi, Á. Hajnal, R. Lovas, T. Kiss, V. Poser, J. Antony, S. Bergweiler, H. Hamzeh, J. Deslauriers, and J. Kovács, ‘‘Interoperable data analytics reference architectures empowering digitaltwin-aided manufacturing,’’ Future Internet, vol. 14, no. 4, p. 114, Apr. 2022.