USCE2QC

A Holistic Approach to Quantum Computing Dependability: From Noise Suppression in Ultrastrong Coupling Regime to the Edge-to-Quantum Continuum

D1: Analysis and design report

vers. 1.0

Version

Version	Date	Description	Authors
1.0	15/12/2024	This deliverable first reports on the project topic analysis, including related work and state-of-the-art literature review. It also documents the definition, analysis and design of WP2 and WP3 metrics, models and solutions. This report documents the initial analysis and design phases for all activities within Work Package 2 (QC Node Dependability via USC Regime) and Work Package 3 (QC Infrastructure Dependability via the Edge-to-Quantum (E2Q) Computing Continuum). It also includes a first version of USC regime-based solutions for quantum noise suppression and error detection/correction, as well as the E2Q architecture, models, mechanisms and strategies.	Salvatore Distefano Roberto Stassi Mansur Ziiatdinov Carmelo Munafo’

Table of Contents

Introduction **5**
Purpose of the Deliverable 5
Overview of the USCE2QC Project 5
Scope of D1: Analysis and Design Activities 6
Document Structure 6
Work Package 2: QC Node Dependability via Ultrastrong Coupling (USC) Regime - Analysis and Design **7**
Task 2.1: USC-based Noise Suppression Solutions 7
2.1.1. State-of-the-Art Analysis of Quantum Noise and Coherence in Superconducting Qubits 7
2.1.2. Review of Ultrastrong Coupling (USC) Regime Theory and Applications in Quantum Computing 8
2.1.3. Initial Analysis of USC Regime Impact on Qubit Coherence and Noise Suppression Mechanisms 8
2.1.4. Design of Novel USC-based Noise Suppression Techniques 9
2.1.5. Expected Outcomes and Challenges 9
Task 2.2: USC-based Quantum Error Detection/Correction Solutions 10
2.2.1. State-of-the-Art Analysis of Quantum Error Correction (QEC) Protocols for Superconducting Qubits 10
2.2.2. Exploration of Virtual Photons and Their Role in Quantum Error Detection/Correction 10
2.2.3. Initial Analysis of USC Regime's Potential for Enhanced Error Detection and Correction 10
2.2.4. Design of Protocols and Mechanisms for USC-based Quantum Error Detection/Correction 11
2.2.5. Expected Outcomes and Challenges 11
Task 2.3: USC-based Quantum Gates 11
2.3.1. Review of Standard Quantum Gate Implementations in Superconducting Circuits 11
2.3.2. Analysis of Gate Operations within the USC Regime 12
2.3.3. Initial Design of Novel Quantum Gates for Fast and Reliable Operation in the USC Regime 12
2.3.4. Expected Outcomes and Challenges 12
Task 2.4: Dependability Assessment of USC Solutions 12
2.4.1. Definition of Key Dependability Metrics for Quantum Computing Nodes 12
2.4.2. Design of Analytical Models for Assessing Quantum Node Dependability, Noise Characteristics, and Error Propagation 13
2.4.3. Design of Simulation Tools and Methodologies for Evaluating USC-based Solutions 13
2.4.4. Design of Frameworks for Comparing and Benchmarking Different USC Approaches 13
2.4.5. Expected Outcomes and Challenges 14
Work Package 3: QC Infrastructure Dependability via the Edge-to-Quantum (E2Q) Computing Continuum - Analysis and Design **14**

Task 3.1: E2Q Continuum Architecture 14
Task 3.2: E2Q Offloading Mechanisms 19
3.2.1. Survey and Review of Task Decomposition, Task Mapping, Service Discovery and Offloading Mechanisms in Distributed Systems 19
3.2.2. Analysis of Quantum Task Characteristics and Requirements for Offloading 24
3.2.3. Initial Design and Development of Intelligent Task Partitioning Algorithms for Hybrid Classical-Quantum Workloads 26
3.2.4. Initial Design and Development of Resource Mapping Algorithms for Offloading Tasks within the E2Q Continuum 27
3.2.5. Expected Outcomes and Challenges 28
Task 3.3: E2Q Dependability Policies and Strategies 28
3.3.1. Definition of Key Dependability Metrics for the E2Q Continuum 28
3.3.2. Initial Design of Analytical Models for E2Q Dependability Assessment 29
3.3.3. Initial Design of Mechanisms and Strategies to Define and Enforce Dependability Policies 30
3.3.4. Initial Design of Scheduling Algorithms for Dependable Task Execution in the E2Q Continuum 32
Task 3.4: E2Q Orchestration 33
3.4.1. Review of Orchestration Concepts and Tools in Distributed and Cloud Environments 33
3.4.2. Initial Design and Development of Mechanisms for Placing and Deploying E2Q Nodes and Services 34
Conclusion and Research Implications 35
Conclusions and Next Steps **36**
References **37**

Introduction

Purpose of the Deliverable

This document, Deliverable 1 (D1) titled “Analysis and Design Report”, serves as the initial foundational output of the USCE2QC project. Its primary purpose is to detail the comprehensive analysis and design phases undertaken during the first five months of the project, specifically focusing on the activities within Work Package 2 (QC Node Dependability via Ultrastrong Coupling Regime) and Work Package 3 (QC Infrastructure Dependability via the Edge-to-Quantum (E2Q) Computing Continuum). This report lays out the theoretical underpinnings, design specifications, and initial conceptualizations that will guide the subsequent development and validation phases of the project, providing a clear roadmap for achieving the stated objectives.

Overview of the USCE2QC Project

The USCE2QC project aims to address critical challenges in the realization of dependable and fault-tolerant quantum computers, particularly within the current Noisy Intermediate-Scale Quantum (NISQ) era. Recognizing that the path to truly functional quantum machines is long and complex due to issues like environmental noise causing decoherence and errors, USCE2QC adopts a dual-pronged approach. Firstly, it investigates the ultrastrong coupling (USC) regime to enhance the intrinsic dependability of individual quantum computing nodes. Secondly, it explores the emerging Edge-to-Quantum (E2Q) computing continuum to improve the dependability of hybrid classical-quantum infrastructure at a distributed level. The project seeks to implement proactive and reactive solutions for node-level dependability and to design and deploy robust E2Q architectures, offloading mechanisms, and dependability policies. As a collaborative effort between Università degli Studi di Messina and SmartMe.io s.r.l, the project embodies a unique blend of academic research and industrial application, striving to pioneer quantum computing exploitation in Southern Italy and foster new economic opportunities.

Scope of D1: Analysis and Design Activities

This report systematically presents the outcomes of the initial analysis and design activities for each task within Work Packages 2 and 3. For Work Package 2, the scope includes the foundational analysis of noise suppression techniques utilizing the USC regime, the conceptual design of quantum error detection and correction solutions based on USC, the preliminary design of novel quantum gates optimized for USC environments, and the initial definition of metrics, models, and tools for assessing quantum node dependability. For Work Package 3, the scope encompasses a comprehensive survey of existing distributed computing paradigms relevant to the E2Q continuum, the initial architectural design of the E2Q framework including resource discovery, the conceptualization of intelligent task offloading mechanisms for hybrid classical-quantum workloads, the preliminary design of dependability policies and strategies within the E2Q continuum, and the initial considerations for E2Q orchestration mechanisms. This deliverable consolidates these preliminary findings and designs, setting the stage for subsequent development and implementation efforts.

Document Structure

The remainder of this document is structured as follows: Section 2 provides a detailed account of the analysis and design activities carried out within Work Package 2, covering USC-based noise suppression, error detection/correction, quantum gates, and dependability assessment. Section 3 similarly elaborates on the analysis and design work performed under Work Package 3, focusing on the E2Q continuum architecture, offloading mechanisms, dependability policies, and orchestration. Finally, Section 4 offers an overall conclusion, summarizing the key findings and designs, outlining their anticipated impact on the project's objectives, and detailing the next steps leading to future deliverables.

Work Package 2: QC Node Dependability via Ultrastrong Coupling (USC) Regime - Analysis and Design

Task 2.1: USC-based Noise Suppression Solutions

2.1.1. State-of-the-Art Analysis of Quantum Noise and Coherence in Superconducting Qubits

Significant progress has been made in the realization of quantum computers, primarily utilizing superconducting circuits and ion traps. Other promising platforms, such as trapped Rydberg atoms, spin qubits in semiconductors, and photonic systems, are also actively being explored. Despite these efforts, none of these platforms are yet capable of performing practical calculations. Among others, the main issue hindering their development is the inevitable interaction with the environment, which leads to decoherence.

Quantum error correction is the way to achieve fault-tolerant quantum computation, but it is fundamental to have more reliable physical qubits that are protected against external noise. Using superconducting circuits, physical qubits can be protected at the hardware level. For instance, a Cooper pair box, when capacitively shunted, is protected against charge noise in the transmon regime, drastically increasing its coherence time. There are two opposite hardware-level strategies to protect qubits: one makes use of a few large elements in specific configurations, such as bifluxon and 0-π qubit; the other uses a large number of elements in a specific arrangement. Both strategies are challenging due to their stringent energy and configuration requirements. The last approach results in symmetry-protected ground states, also known as decoherence-free subspaces. A notable example is the 1D Kitaev model of spinless fermions, which exhibits a double degenerate ground state. The Kitaev model can be mapped to a transverse Ising spin model using the Jordan-Wigner transformation, but, after the transformation, it becomes susceptible to local symmetry-breaking noise. In general, these protected systems require a high degree of symmetry, such that their states are not sensitive to any slow perturbation (topological protection). Drawbacks of these protected states are that they are degenerate and also insensible to external manipulation. Preparing and reading out quantum states, as well as performing single- and two-qubit gates for quantum computation, requires a perturbation that resolves the degeneracy and, consequentially, the protection.

However, topological quantum computation offers a theoretical framework for performing calculations involving protected degenerate quantum states, but its practical implementation remains a distant goal.

2.1.2. Review of Ultrastrong Coupling (USC) Regime Theory and Applications in Quantum Computing

By using superconducting circuits, the ultrastrong coupling regime between artificial atoms and electromagnetic modes has been achieved. Since this regime is implemented on the same platform as some quantum computers, it can be exploited for quantum computing. For instance, Nataf et al. (Nataf, P. and Ciuti, C., 2011. Protected Quantum Computation with Multiple Resonators in Ultrastrong Coupling Circuit QED. Physical Review Letters, 107(19), p.190402.) propose an array of fluxonium qubits ultrastrongly coupled with an LC resonator to enhance their coherence time; Stassi et al. (Stassi, R. and Nori, F., 2018. Long-lasting quantum memories: Extending the coherence time of superconducting artificial atoms in the ultrastrong-coupling regime. Physical Review A, 97(3), p.033823.) proposed to couple longitudinally a flux qubit to a resonator in the ultrastrong coupling regime to increase its relaxation time.

2.1.3. Initial Analysis of USC Regime Impact on Qubit Coherence and Noise Suppression Mechanisms

The objective of this analysis phase is to define theoretical strategies for quantum noise suppression, with a particular focus on the ultrastrong coupling (USC) regime. This regime represents a frontier where the interaction between quantum systems is comparable to or exceeds their characteristic energies. Our analytical approach will involve the use of numerical simulations based on the Master equation to investigate how USC interactions can enhance the system's coherence time, overcoming the limitations of individual physical qubits. Specifically, our methodology is based on the simulation of the evolution of a logical quantum state formed by a hybrid system in ultrastrong interaction. The evolution of this system will be described and governed by a Master equation in Lindblad form. This approach will allow us to model and compare the behavior of the logical qubit with the evolution of a single qubit subjected to the same noisy environment. The analysis will focus on evaluating the impact of decoherence channels, particularly the processes of relaxation and pure dephasing. Through this theoretical framework, we aim to demonstrate the suppression of these noise rates, laying the groundwork for the design of more robust quantum architectures.

This methodology represents a crucial design phase and a solid theoretical foundation for subsequent development activities.

2.1.4. Design of Novel USC-based Noise Suppression Techniques

Our design approach is centered on creating a robust logical qubit by leveraging the advantages of the ultrastrong coupling (USC) regime. The USC regime provides a unique environment where the strong interaction between quantum systems can be exploited to achieve greater stability against environmental noise. This methodology is based on the following key design principles:

Intrinsic Noise Protection: We have designed a logical qubit by coupling multiple physical qubits in a specific configuration. The goal of this design is to create a system that is inherently protected from noise, leading to enhanced coherence times for both pure dephasing and relaxation processes. This contrasts with conventional approaches that rely solely on external error correction mechanisms.

High-Speed Quantum Gates: The design includes the implementation of a high-speed quantum gate, the √iSWAP gate, by leveraging the USC regime. This gate is designed to operate on a nanosecond timescale, which is crucial for minimizing the effects of decoherence and improving the fidelity of quantum operations.

Universal Gate Set: The proposed √iSWAP gate, in combination with single-qubit gates, forms a universal gate set. This design choice ensures that the system is capable of performing any quantum computation, providing a solid foundation for a versatile quantum computing architecture.

2.1.5. Expected Outcomes and Challenges

The goal of this task is to theoretically investigate a system of multiple physical qubits coupled in the ultrastrong coupling regime to form a logical qubit. We aim to show that this approach can lead to a significant enhancement of the logical qubit's coherence time, extending it beyond that of a single physical qubit. We also plan to demonstrate the feasibility of single- and two-qubit gates within this framework.

Task 2.2: USC-based Quantum Error Detection/Correction Solutions

2.2.1. State-of-the-Art Analysis of Quantum Error Correction (QEC) Protocols for Superconducting Qubits

Quantum error correction (QEC) is a fundamental pillar for achieving fault-tolerant quantum computation. Current QEC protocols aim to protect delicate quantum states from environmental noise by encoding logical information across multiple physical qubits. While significant progress has been made with techniques like surface codes and bosonic codes, these methods often require a high number of ancillary qubits and complex control systems, posing a significant challenge for scalability and implementation on noisy intermediate-scale quantum (NISQ) devices. The main issue remains the need for fast and high-fidelity measurement and control operations that can outpace decoherence processes to effectively detect and correct errors.

2.2.2. Exploration of Virtual Photons and Their Role in Quantum Error Detection/Correction

This task is dedicated to an exploratory analysis of non-conventional mechanisms for error detection. Specifically, we investigate the potential of virtual photons, which are a hallmark of the ultrastrong coupling (USC) regime, for quantum error detection. Virtual photons, being the transient, non-classical excitations that mediate strong interactions, provide a unique and unexplored resource. Our analysis will delve into how these virtual particle-like interactions can be leveraged to probe a quantum system's state and detect errors in real time, without the need for physical photons that could introduce additional noise and decoherence.

2.2.3. Initial Analysis of USC Regime's Potential for Enhanced Error Detection and Correction

The USC regime offers a promising platform for enhancing error detection. The strong, instantaneous interactions enable extremely fast operations, which can be completed on timescales much shorter than the coherence time of the qubits. Our analysis will focus on how this speed advantage can be used to perform rapid and high-fidelity quantum measurements, a prerequisite for any effective error detection protocol. We will investigate the theoretical models that allow us to utilize the inherent properties of the USC regime to create measurement schemes that are both non-destructive and highly sensitive to state changes, thereby providing a powerful tool for detecting errors as they occur.

2.2.4. Design of Protocols and Mechanisms for USC-based Quantum Error Detection/Correction

The design phase for this task focuses on the conceptualization of novel protocols and mechanisms for USC-based error detection and correction. Our design is based on the principles of Quantum Nondemolition Measurement (QNDM) and coherent state preparation. We propose to design a QNDM protocol that leverages the USC interaction to measure a qubit's observable without disturbing the qubit's state. Additionally, we will design protocols for preparing and manipulating coherent states, which are essential for initializing and controlling the bosonic modes involved in the measurement process. These mechanisms will form the core of a new error detection framework designed to take advantage of the unique properties of USC systems.

2.2.5. Expected Outcomes and Challenges

The primary goal of this task is to lay the theoretical groundwork for a new generation of error detection and correction protocols. While a full implementation is beyond the scope of this project, we expect to demonstrate the theoretical feasibility of using the USC regime for these purposes. A major challenge is the highly complex nature of USC systems, which requires advanced theoretical modeling and simulation. The expected outcome is a set of well-defined protocols and mechanisms that can serve as a foundation for future research and development in this critical area of quantum computing.

Task 2.3: USC-based Quantum Gates

2.3.1. Review of Standard Quantum Gate Implementations in Superconducting Circuits

Quantum gates are the fundamental building blocks of quantum computation, analogous to logic gates in classical computers. In superconducting circuits, standard implementations of quantum gates often face a trade-off between speed and fidelity. While techniques exist to realize both single- and two-qubit gates, their performance is inherently limited by decoherence, as the gate operation time is typically longer than the coherence time of the qubits. Therefore, a primary challenge in quantum computing research is to design gate implementations that can operate at a speed that outpaces the effects of environmental noise.

2.3.2. Analysis of Gate Operations within the USC Regime

This task involves a detailed analysis of the potential for gate operations within the ultrastrong coupling (USC) regime. The USC regime, characterized by interaction strengths comparable to or greater than the energy of the coupled systems, offers a unique opportunity to drastically increase gate speed. The analysis explores how the strong virtual processes inherent to this regime can be activated to create high-fidelity, high-speed gates. This approach is designed to overcome the limitations of conventional gate implementations, allowing for computational times on the order of a few nanoseconds.

2.3.3. Initial Design of Novel Quantum Gates for Fast and Reliable Operation in the USC Regime

Our design approach proposes a novel circuit for the implementation of a universal quantum gate. The proposed circuit consists of two transmons coupled to a flux qubit in the USC regime. This specific design allows for an effective coupling of the two transmons through virtual processes mediated by the flux qubit. This effective interaction is designed to enable the realization of a $\sqrt{i\mathrm{SWAP}}$ gate, which, when combined with a single-qubit gate, forms a universal gate set. This foundational design provides a pathway for performing any quantum computation on this platform.

2.3.4. Expected Outcomes and Challenges

The primary goal of this task is to lay the theoretical foundation for high-speed, high-fidelity quantum gates. We expect to design a gate that can operate significantly faster than standard implementations, thereby mitigating the effects of decoherence. The main challenge lies in the complexity of designing and controlling such an intricate circuit. However, a successful design would demonstrate the feasibility of exploiting the USC regime to create a robust and efficient gate set, paving the way for future experimental implementations.

Task 2.4: Dependability Assessment of USC Solutions

2.4.1. Definition of Key Dependability Metrics for Quantum Computing Nodes

To assess the dependability of our proposed USC-based solutions, we have defined a set of key metrics. These metrics go beyond standard benchmarks to specifically evaluate performance within the USC regime. The primary metrics include:

Coherence time: A measure of how long a quantum state can maintain its superposition and entanglement before being corrupted by environmental noise.

Gate fidelity: A measure of how accurately a quantum gate performs its intended operation.

Susceptibility: A critical metric that measures how sensitive a system is to specific types of noise. It quantifies the system's response to an external perturbation, providing insight into its robustness.

Pure dephasing rates: The rate at which the quantum state loses its phase coherence without losing energy to the environment.

Relaxation rates: The rate at which the quantum state loses energy and returns to its ground state.

2.4.2. Design of Analytical Models for Assessing Quantum Node Dependability, Noise Characteristics, and Error Propagation

Our analytical model is designed to assess the dependability of the USC system by focusing on its noise characteristics and error propagation. We have formulated a theoretical model to investigate the system's behavior, taking into account the effects of environmental interactions. This model is crucial for understanding how noise impacts the logical qubit and how errors propagate through the circuit.

2.4.3. Design of Simulation Tools and Methodologies for Evaluating USC-based Solutions

The core of our methodology is the design of a simulation framework based on the Master equation. This tool allows us to numerically simulate the evolution of a logical quantum state formed by multiple qubits in the ultrastrong coupling regime. This approach is designed to compare the performance of our logical qubit with that of a single, non-interacting qubit subjected to the same environmental noise, thereby demonstrating the benefits of our USC approach.

2.4.4. Design of Frameworks for Comparing and Benchmarking Different USC Approaches

We have designed a framework to benchmark different USC approaches by evaluating their impact on key dependability metrics. This framework includes protocols for performing numerical simulations and tests to assess the effectiveness of our noise suppression techniques and high-speed gates. This design phase is crucial for ensuring that our proposed solutions are not only theoretically sound but can also be rigorously tested and validated against other approaches.

2.4.5. Expected Outcomes and Challenges

The expected outcome of this task is a comprehensive assessment of the dependability of our USC solutions. We aim to demonstrate, through numerical simulations, that our logical qubit and high-speed gates can achieve a significantly higher level of robustness and fidelity compared to standard implementations. The main challenge lies in accurately modeling the complex dynamics of USC systems and ensuring that the simulations reflect real-world conditions.

Work Package 3: QC Infrastructure Dependability via the Edge-to-Quantum (E2Q) Computing Continuum - Analysis and Design

Task 3.1: E2Q Continuum Architecture

According to the nodes involved, distributed computing patterns are categorized as Edge computing if based on devices at the edge of the network (e.g. sensors, smart objects, smart-phones, single board computers), Fog computing if performed on proximity servers, or Cloud computing when deployed on virtual servers running on (remote) datacenters. Thereby, a hierarchy of nodes can be identified ranging from billions of simple (IoT) devices at the bottom Edge to thousands of Cloud datacenters on top, across millions of Fog servers in the middle, often mixed into Edge-Fog-Cloud computing blends, i.e. the so called computing continuum [1].

The computing continuum targets the integration of heterogeneous computing resources and services across different devices, environments, and platforms, allowing to access, share, provision, and exploit them regardless of the node or location, seamlessly, as a service, on-demand. It thus enables hybrid patterns where Edge (or Fog) devices can ask support to higher level Fog and/or Cloud nodes (bottom-up vertical task offloading), or even Cloud or Fog nodes may require low level Fog or Edge nodes for computing (top-down vertical task offloading).

In terms of resource, the computing continuum, as well as Edge, Fog and Cloud ones, may provide different types, i.e. processing, storage, sensing-actuation (I/O), and networking ones, enabling the implementation of distributed Von Neumann machine patterns [2]. This connects with the software defined paradigm, widely adopted in current Fog and Cloud contexts as Software Defined Data Center (SDDC) [3] and Infrastructure as Code (IaC) [4] to softwarize typical hardware resources. Thereby, different resources can be grouped and orchestrated altogether, as a whole, and thus provided as a service. Pushing the computing continuum software defined paradigm till sensors and actuators, although challenging, enables new avenues such as networked/distributed robotics, intelligence, teleoperation, remote sensing and so on.

Furthermore, quantum computing recently emerged in the (parallel) computing landscape with the hype of scaling up the problem tractability in the complexity class theory, a big potential so far partially demonstrated with some relevant quantum algorithms (e.g. Shor, Grover) [5]. Distributed quantum architectures and patterns have already been proposed and implemented [6]. An example is quantum Cloud Computing [7], i.e. the mainstream QC provisioning paradigm so far, allowing users to approach QC Clouds like IBM [8], D-Wave [9], Google [10], to name but a few, on demand. Quantum Fog Computing [11] exploits quantum nodes as proximity servers to share computation resources [12]. Quantum Edge Computing [13] ports the idea of edge computing to the quantum context, with an outlook on mobile quantum computing technologies (quantum processing unit, sensors, memories and networks). In addition to its applications in the processing previously discussed, quantum mechanics has a broader scope, encompassing networking, focusing on quantum data transmission and cryptography; sensors, dealing with measurements of e.g. temperature, weight, and other physical quantities; and storage, providing error-resistant and durable data memory and storage solutions [14]. In this light, despite the roadmap to quantum technologies integration and miniaturization are still ongoing, it is, however, possible to start thinking about a quantum computing continuum, a big challenge for the distributed computing research community. Currently, it makes sense to propose a classical-quantum hybrid continuum based on existing quantum technologies, mixed with classical ones, with potential projections on future developments such as mobile QPU, quantum Von Neumann machines and quantum edge computing [13].

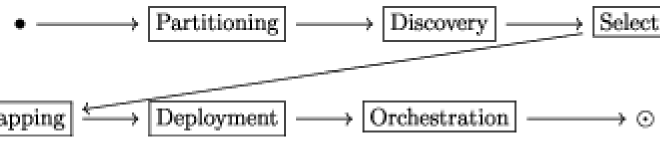

Figure 1. 3D Continuum workflow

The process to plan the 3D Continuum system, based on the application logic requirements, is here investigated. The process starts from a (Edge, Fog, or Cloud) node needing to run a (processing, storage, networking, sensing or even combined) task that cannot be performed locally (too complex, not available resources, etc.). It therefore sends an offloading request to the 3D continuum following the workflow of Fig. 1. Inspired by similar work [15], this 3D Continuum workflow consists of six steps, of which only the deployment and orchestration steps are mandatory, but all steps are usually performed to properly set up a computing continuum platform able to run the application tasks to outsource.

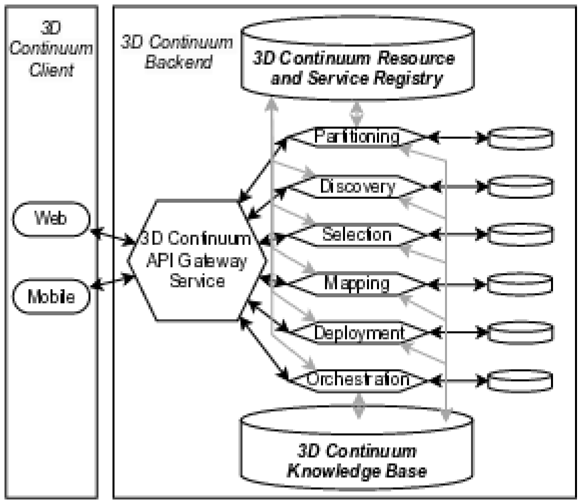

Figure 2. 3D Computing Continuum blueprint

The workflow above described allows to draft a blueprint of a 3D computing continuum management system or framework, shown in Figure 2. The rationale behind it is to implement each of the workflow steps by a specific component of the framework. Then, to cope with the different types of resources, levels and physics nature, also considering that some of such steps, as discussed above, are optional, a microservices architecture is adopted. Thereby, each microservice could even be deployed in any node or device of the 3D Continuum.

The 3D Continuum framework is thus organized into two main subsystems: the client frontend and the backend. The 3D Continuum Client frontend runs on the client node, i.e. the node aiming to outsource a complex problem, task or service to the computing continuum infrastructure. To allow any node to submit outsourcing requests to the infrastructure, web and mobile clients are planned, forwarding JSON or HTML formatted messages. It thus implements a user interface able to specify, configure and setup the requests functional (level: local, edge, fog, cloud; resource type: processing, memory, networking; physics: classical, quantum) and non functional (e.g. security, performance, dependability) requirements. The request is then forwarded to the framework for processing, where it is managed by the 3D Continuum Backend implementing a microservices architecture. The backend interfaces with an API Gateway Service that, rather than a one-size-fits-all style API, may expose a different, customized API for each client and/or request type. A pattern widely adopted in real applications such as, e.g., the Netflix API gateway, which runs client-specific adapter code to provide each client with an API that best suits its requirements, successfully implemented even in quantum computing contexts [19]. Based on the customized request type, it deploys the proper microservice workflow which may include all or a subset of the aforementioned steps. The latter are indeed represented by the corresponding backend microservices, namely: Partitioning, Discovery, Selection, Mapping, Deployment, and Orchestration. Furthermore, all such microservices interact with two basic 3D Continuum services: the 3D Continuum Knowledge Base and the 3D Continuum Resource and Service Registry, highlighted by grey arrows in Fig. 2.

One of the examples of workloads that can be run on the proposed architecture is quantum machine learning (QML) models. QML algorithms (e.g., quantum neural networks, quantum optimization) often require significant computational resources. The 3D Continuum allows distributing these workloads across edge, fog nodes, with quantum resources acting as "computational accelerators" for specific tasks (e.g., quantum kernel methods, quantum optimization). For example, edge devices could preprocess data, while cloud-based quantum processors handle complex QML tasks (e.g., training quantum models).

The 3D Continuum enables hybrid architectures where classical and quantum components collaborate. For instance:

- Edge: sensor data (e.g., for real-time analytics).
- Fog: Perform initial ML inference using classical models.

- Cloud: Execute QML tasks (e.g., quantum-enhanced clustering or regression) on quantum processors.
- Quantum Nodes in the Cloud: Act as "accelerators" for tasks like quantum Fourier transforms or variational quantum algorithms.

The 3D Continuum's focus on resource discovery and orchestration (as discussed in the report) aligns with QML's need for dynamic allocation of quantum and classical resources. For example: quantum resources (e.g., quantum processors) could be allocated to QML tasks only when needed, reducing costs and energy consumption.

Key Applications of QML in the 3D Continuum.

- Quantum-Enhanced Edge Analytics: edge devices (e.g., IoT sensors) could use lightweight classical ML models, while quantum resources in the fog/cloud handle complex tasks like: quantum kernel methods for non-linear classification, quantum Boltzmann machines for probabilistic modeling, etc.
- Distributed QML Training: The 3D Continuum could enable federated QML, where quantum models are trained across distributed nodes (edge/fog/cloud) for enhanced data privacy. For example: edge nodes send encrypted classical features to quantum nodes for training; quantum processors optimize model parameters using quantum algorithms (e.g., variational quantum eigensolvers).
- Quantum-Enhanced Optimization: QML can solve optimization problems (e.g., in logistics, finance) more efficiently than classical methods. The 3D Continuum provides the infrastructure to: distribute across edge/fog nodes, and use quantum processors for solving combinatorial optimization problems.

However, the 3D Continuum has also challenges in deployment:

- Quantum-Classical Integration: Ensuring seamless communication between classical nodes (edge/fog/cloud) and quantum processors.
- Dependability: quantum hardware is prone to errors (e.g., decoherence), requiring robust error correction and fault-tolerant designs (as emphasized in the report's focus on dependability).
- Latency: Quantum operations may introduce delays, necessitating efficient task scheduling in the E2Q Continuum.

The report emphasizes the integration of quantum computing into the edge-to-cloud continuum, which directly supports QML by enabling hybrid architectures (classical + quantum), facilitating resource discovery and orchestration for QML tasks, and addressing dependability (e.g., ensuring reliable quantum-classical interactions).

The 3D Continuum provides the infrastructure and architecture needed to deploy quantum machine learning at scale. By combining the distributed, heterogeneous nature of edge/fog/cloud computing with quantum resources, the 3D Continuum enables efficient, scalable, and adaptive QML solutions. This synergy is critical for advancing applications like real-time analytics, optimization, and AI in domains such as IoT, healthcare, and finance.

Another technique that can be employed to increase dependability of quantum algorithms is quantum fingerprinting. The quantum fingerprinting is a technique that compresses a long classical input into a much shorter quantum register while preserving the essential properties of the input, allowing different inputs to be distinguished. It was introduced by Buhrman et al. [16] to solve the EQUALITY problem in the simultaneous message passing model (i.e., Alice and Bob have their inputs x and y correspondingly, and simultaneously send one message to the referee who computes the output). It was then developed and applied to different problems, such as quantum cryptography [17], quantum string algorithms [18], etc.

Task 3.2: E2Q Offloading Mechanisms

3.2.1. Survey and Review of Task Decomposition, Task Mapping, Service Discovery and Offloading Mechanisms in Distributed Systems

Task Decomposition in Distributed Systems

Task decomposition is a critical aspect that enables efficient resource utilization and scalability for cloud environments. It involves breaking down complex tasks into smaller, manageable components that can be executed independently across distributed nodes. This approach not only enhances performance but also improves fault tolerance and adaptability in dynamic environments.

Microservices-Based Decomposition involves dividing extensive applications into smaller, independent services, each tailored for a specific function. This method facilitates adaptability in scaling and updating services without necessitating changes to the entire application. For instance, Netflix employs microservices [20] to manage streaming, user accounts, and recommendations separately, exemplifying modular architecture. Key advantages of this technique include fault isolation, which allows failures in one service to be addressed without impacting the entire system, and technology heterogeneity, enabling different services to use technologies best suited to their tasks.

Workflow-Based Decomposition uses directed acyclic graphs (DAGs) to divide tasks into atomic operations connected by dependencies. This approach is ideal for parallel processing. Apache Airflow leverages this technique [21] for orchestrating complex workflows in data engineering pipelines, such as ETL (Extract, Transform, Load) processes. DAGs ensure that tasks are executed in the correct order, respecting dependencies while maximizing parallelism. For example, Spotify employs DAGs to manage its recommendation engine, where data preprocessing, feature extraction, and model training occur as discrete, interdependent steps. The visualization of workflows through DAGs also aids in debugging and optimization.

Resource-Aware Decomposition optimizes task breakdown based on computational and communication needs, ensuring efficient hardware utilization. Kubernetes [22], for example, schedules workloads across cluster nodes based on resource availability. This technique powers cloud platforms like Google Cloud and AWS, where resource-aware scheduling ensures high availability and cost efficiency. Complementing the analysis of the computational requirements and data locality of tasks, resource-aware decomposition minimizes latency and maximizes throughput by enabling efficient use of resources.

Adaptive Frameworks and Interaction-Based Strategies adaptively decompose tasks by analyzing real-time workloads, often using AI for dynamic management. These frameworks allocate processing tasks between distributed components efficiently, optimizing real-time decision-making and enhancing system performance. Adaptive decomposition enables systems to respond to varying workloads dynamically, ensuring optimal resource utilization. Key techniques include feedback loops for continuous workload assessment and machine learning models for predictive task allocation. While adaptive systems offer high efficiency, they require sophisticated monitoring and control mechanisms to function effectively.

Task Mapping Strategies

Modern distributed systems employ various task mapping strategies to optimize resource utilization and performance. These strategies are essential for ensuring that tasks are executed efficiently across distributed nodes, particularly in environments with heterogeneous resources.

Dynamic Resource Allocation focuses on real-time system resource monitoring and adaptive scheduling to balance workloads effectively across nodes. This strategy helps in scaling resources elastically to meet demands efficiently. For example, Amazon Web Services (AWS) uses dynamic resource allocation in its EC2 instances, enabling scaling based on user demand. Techniques such as load balancing algorithms and predictive analytics are often integrated to ensure resources are distributed optimally. These mechanisms allow distributed systems to dynamically adjust to sudden workload changes, maintaining consistent performance and minimizing bottlenecks.

Service Discovery and Registration enables the automatic registration of services, ensuring seamless integration and efficient routing of requests. It also ensures dynamic adjustments to maintain system reliability. Kubernetes provides service discovery for containerized applications, facilitating integration between services. Service mesh frameworks like Istio enhance service discovery by providing additional features such as traffic management, observability, and security enforcement.

QoS-Aware Mapping addresses the quality of service (QoS) needs of applications by prioritizing tasks based on latency sensitivity and resource requirements. This ensures critical resources are reserved and Service Level Agreements (SLA) compliance for cloud providers such as AWS or Google Cloud is maintained. For example, in media streaming platforms, low-latency streaming is prioritized over background analytics tasks. Techniques such as priority queues and policy-based scheduling are often used to enforce QoS requirements, ensuring high-priority tasks are executed without delay.

Offloading Mechanisms

Offloading mechanisms in distributed systems are essential for optimizing resource utilization and enhancing performance. They enable the delegation of computational tasks to specialized resources, ensuring efficient execution and scalability.

Container-Based Offloading allows for lightweight containerization, offering portability across multiple platforms. This technique ensures runtime environment isolation, enabling fast deployment and scaling. Docker [23] is widely adopted for container-based offloading, as it enables rapid deployment of microservice applications by encapsulating applications and their dependencies into containers. Additionally, container orchestration tools like Kubernetes can further enhance container-based offloading by automating deployment, scaling, and management of containerized applications across clusters.

Serverless Computing focuses on Function-as-a-Service (FaaS) platforms that execute code in response to events. The serverless model automatically scales resources as needed and operates on a pay-per-use pricing model, providing cost efficiency and operational simplicity. AWS Lambda [24] and Azure Functions [25] are key examples of serverless computing platforms offering seamless scalability and event-driven execution. The main benefit of these technologies is the reduced operational overhead as the infrastructure is fully managed, allowing for more focus on application logic, while also providing automatic scaling based on demand. However, challenges such as cold start latency and limited execution time must be addressed to ensure optimal performance.

Hybrid Offloading integrates mixed resource types and specialized hardware for cross-platform execution. This mechanism enables optimal resource selection by leveraging the strengths of both classical and quantum resources, ensuring efficient task execution in heterogeneous environments. For instance, hybrid offloading is utilized in cloud gaming platforms to balance rendering tasks between local devices and cloud servers. In the context of quantum computing, hybrid offloading facilitates the partitioning of tasks between quantum processors and conventional CPUs/GPUs, optimizing the usage of computational resources for complex problems.

Cloud Environment Integration

Cloud environments play a pivotal role in modern distributed systems, providing scalable infrastructure and services that enhance application deployment and management. The integration of cloud environments with distributed systems introduces several key methodologies:

Container Orchestration enables the automated deployment and scaling of applications. It includes features like rolling updates, service mesh integration, and configuration management to streamline cloud-based workflows. Kubernetes, for instance, offers container orchestration capabilities and is widely used across industries to manage containerized applications efficiently. Rolling updates ensure zero downtime during application upgrades, which is critical for maintaining high-availability systems.

API Management facilitates service discovery, registration, and efficient routing. It also enforces security layers such as authentication and rate limiting while providing monitoring insights. Platforms like Google Cloud's Apigee and Microsoft Azure API Management are practical examples that enable businesses to manage APIs effectively. These platforms allow developers to design, secure, and monitor APIs as products, making it easier to expose functionality to partners and customers.

State Management addresses persistent and transient data handling requirements in distributed systems. Techniques like distributed caching, message queuing, and event streaming ensure consistency and performance. Technologies such as Redis for distributed caching and Apache Kafka for event streaming are widely adopted solutions for state management. For instance, Redis is often used for session management in web applications to ensure faster load times, while Apache Kafka excels in real-time data pipelines and event-driven architectures, making it suitable for scenarios requiring high-throughput and fault-tolerant messaging.

API Discovery

API discovery is an important building block for distributed systems, enabling services to easily locate and interact with each other. Various approaches have been developed to facilitate API discovery, each with its own strengths and weaknesses.

Static Documentation-Based Discovery: This traditional approach involves providing detailed documentation for APIs, typically in formats like OpenAPI (formerly Swagger). Documentation serves as a manual guide for developers to understand and interact with APIs. While effective for human users, it lacks dynamic adaptability for runtime discovery by systems.

Service Registry Tools: Tools like Consul, Eureka, and ZooKeeper facilitate dynamic API discovery by maintaining a registry of active services. These tools allow services to register their availability and provide mechanisms for clients to locate and interact with these services. They are particularly effective in microservices environments where services frequently scale and change dynamically.

Service Mesh Approaches: Modern service meshes such as Istio and Linkerd offer advanced API discovery capabilities. By intercepting all service communication, service meshes enable features such as traffic routing, service observability, and security policies, all while maintaining up-to-date service discovery. These tools integrate seamlessly with existing microservices infrastructures, though they introduce additional complexity and overhead.

3.2.2. Analysis of Quantum Task Characteristics and Requirements for Offloading

Quantum task execution in distributed systems requires meeting requirements associated with quantum computing in such a way that allows integration with classical computing tasks. This section outlines the essential components necessary for executing quantum tasks within a distributed framework.

Resource Requirements

Quantum Processing Units (QPUs) form the cornerstone of quantum computing infrastructure, offering the hardware backbone necessary for quantum information processing. These units feature qubits, the fundamental units of quantum information, capable of existing in superpositions that enable quantum parallelism. Beyond the hardware, QPUs rely on quantum circuit simulation capabilities to design and validate quantum algorithms, ensuring optimal performance before execution on actual quantum devices. At the heart of these computations are quantum gates—such as the Hadamard, CNOT, and Pauli gates—that manipulate qubit states to construct quantum circuits. However, the fragile nature of qubits due to decoherence and noise necessitates robust quantum error correction mechanisms. Techniques like surface codes and Shor's code help detect and correct errors without disturbing the quantum state. The hardware ecosystem supporting QPUs further includes control and measurement devices, cryogenic systems, and quantum software tools like Cirq and Qibo, which facilitate algorithm development and execution.

Classical computing resources play a complementary role in quantum systems, handling tasks such as pre-processing of quantum algorithms, post-processing of quantum results, circuit optimization, and error mitigation preprocessing. These processes not only prepare input data for quantum computations but also interpret the output, bridging the gap between quantum and classical paradigms. Such resources are also essential for managing the hybrid nature of quantum-classical workflows, where classical systems orchestrate quantum tasks and manage the flow of information between quantum and classical components, as well as managing communications between quantum and classical tasks.

Memory requirements in quantum computing are multi-faceted, encompassing the storage of quantum states and classical state buffering. Quantum state storage involves retaining superpositional states and entangled configurations, while classical state buffering facilitates seamless quantum-classical interactions.

State Management

Quantum states, represented as superpositions, are highly sensitive to environmental interactions that can cause decoherence—the loss of quantum coherence. Maintaining coherence involves isolating qubits from external noise while ensuring precise control during operations. Techniques like dynamic decoupling and error correction codes are vital for coherence preservation.

Entanglement, another cornerstone of quantum computing, allows qubits to exhibit correlations that are impossible in classical systems. Managing entanglement requires robust protocols for state preparation, measurement strategies, and inter-qubit communication. Mismanagement of entanglement can lead to errors that propagate across the system, undermining computational results.

State synchronization and recovery mechanisms are equally important for distributed quantum systems. These ensure that quantum states remain consistent across multiple processing units, enabling reliable execution of quantum algorithms. Techniques like quantum repeaters and entanglement distillation are employed to address these challenges, ensuring high fidelity in quantum state transmission.

Quantum-Classical integration

The integration of quantum computing with classical distributed systems presents several challenges that must be addressed to ensure efficient and effective operation. On the other hand, interfacing quantum and classical systems enables the execution of hybrid algorithms that leverage the strengths of both paradigms, such as quantum machine learning and optimization problems. Key solutions focus on hardware abstraction, state translation, and timing coordination to seamlessly bridge the gap between the two paradigms.

Hardware Abstraction is essential due to the diversity in quantum hardware architectures, each with unique capabilities and constraints. Implementing hardware-agnostic interfaces ensures that applications can run on multiple quantum processors without requiring extensive modifications. Abstraction layers for quantum resource management further simplify the development process, allowing developers to focus on algorithm design rather than hardware-specific details. Standardized APIs for quantum operations, such as those provided by frameworks like Qiskit and Cirq, play a crucial role in enabling cross-platform compatibility. Additionally, support for multiple quantum providers ensures flexibility and scalability in deploying quantum applications.

State Translation between quantum and classical data representations is another critical challenge. Efficient encoding and decoding protocols are necessary to convert data without significant loss of information or computational overhead. Standardized data formats and optimized state conversion mechanisms, such as intermediate representation layers, facilitate seamless integration. For instance, quantum assembly languages like OpenQASM serve as an intermediate format, bridging high-level quantum algorithms and low-level hardware execution.

Timing Coordination is pivotal for synchronizing quantum and classical operations, as discrepancies can lead to errors and inefficiencies. Event-driven architectures enable precise timing control by triggering operations based on specific conditions. Mechanisms like buffering and queuing systems manage the flow of data between components, ensuring smooth transitions. Asynchronous operation handling further enhances system performance by allowing quantum and classical tasks to proceed independently when possible, reducing idle times and improving resource utilization.

3.2.3. Initial Design and Development of Intelligent Task Partitioning Algorithms for Hybrid Classical-Quantum Workloads

The initial phase of the prototype design centered on developing task partitioning algorithms tailored for hybrid classical-quantum workloads. This involved identifying computational tasks suitable for quantum acceleration and designing strategies for their efficient partitioning.

Algorithmic Frameworks

Preliminary frameworks were established to determine task partitioning criteria, focusing on computational cost, data dependencies, and ease of management. We chose Temporal as the orchestration platform to manage these tasks, leveraging its capabilities to handle complex workflows and ensure robust execution across hybrid systems by building on the concept of "task queues" for partitioned workloads. Features such as retries, state management, and gRPC-based communication also proved useful for debugging and monitoring task execution.

Simulation Studies

Simulations were conducted to evaluate the performance of partitioned workloads, leveraging classical computing to validate the feasibility of proposed algorithms before implementation on quantum hardware, but ensuring that the implementation could be seamlessly integrated with quantum systems when available. This involved using classical simulators to mimic quantum operations, allowing for early detection of potential bottlenecks and inefficiencies in task partitioning. Due to its popularity and wide compatibility, we utilized the Qiskit framework for these simulations, which provided a robust environment for testing quantum algorithms and their integration with classical systems.

Integration with Temporal Workflow Systems

To enable the orchestration and partitioning of hybrid workloads, we built a small workflow that implements Shor's algorithm by isolating the quantum portion from the classical processing of the task and managing it through Temporal. This workflow demonstrated the feasibility of partitioning tasks into classical and quantum components.

3.2.4. Initial Design and Development of Resource Mapping Algorithms for Offloading Tasks within the E2Q Continuum

The next phase focused on developing prototypes of resource mapping algorithms to facilitate the offloading of tasks within the E2Q continuum while ensuring efficient use of resources. This involved creating strategies for task placement and resource management that could adapt to the dynamic nature of hybrid workloads.

Hardware-Aware Mapping Strategies

Incorporating knowledge of hardware specifications to optimize task placement, ensuring efficient utilization of both quantum and classical resources. The first prototype of the architecture was deployed using Docker Compose, which allowed us to simulate a hybrid environment by designating specific containers for edge and quantum resources.

Error Mitigation Techniques

Implementing strategies to account for the probabilistic nature of quantum operations, thereby enhancing the reliability of offloaded tasks. By leveraging Temporal's state management and retry mechanisms, the system ensured resilience against transient errors and supported reliable task execution.

3.2.5. Expected Outcomes and Challenges

We expect to produce a proof of concept demonstrating the feasibility of dynamic task partitioning and resource mapping in hybrid classical-quantum workloads. This includes a prototype that showcases a static resource mapping through Docker Compose, simulating the E2Q continuum with edge and quantum resources. The prototype will also include a workflow that implements Shor's algorithm, demonstrating the orchestration of hybrid workloads using Temporal.

Several challenges are anticipated, including the need for robust error handling in quantum operations, the complexity of managing dynamic workloads, and lowering the development overhead for integrating new microservices into the architecture. Additionally, ensuring that the system can adapt to evolving quantum capabilities and integrating with existing classical systems will require ongoing maintenance.

Task 3.3: E2Q Dependability Policies and Strategies

3.3.1. Definition of Key Dependability Metrics for the E2Q Continuum

The dependability of the Edge-to-Cloud Continuum is governed by several critical metrics that ensure the system’s reliability, efficiency, and security in dynamic, distributed environments:

Latency: The round-trip time required for data transmission between endpoints, critical for applications that demand real-time responsiveness. Latency varies from less than 1ms on-device to more than 20ms in the cloud [26].

Throughput: The amount of data processed over time, with higher throughput typically found in cloud layers compared to edge or device layers [26].

Availability: The proportion of time resources and services are operational and accessible, often increased via redundancy or replication.

Security and Confidentiality: Measures to prevent unauthorized data access or breaches, including cryptographic safeguards, access control, and secure communication channels [26].

Fault Tolerance and Reliability: The system's ability to continue functioning correctly despite failures, assessed through metrics like mean time between failures (MTBF) [26]. Fault tolerance mechanisms also include error detection, failover, and backup systems.

Additional dimension of quantum-physical nature of E2Q Continuum adds several quantum-specific dependability metrics that are essential for evaluating the reliability and robustness of quantum systems, particularly as they integrate with classical infrastructure and face unique challenges such as noise, decoherence, and probabilistic outcomes:

Reproducibility: In quantum systems, this refers to repeated execution of quantum circuits yielding statistically consistent results, measurable via statistical distances (e.g., Hellinger distance) between output distributions of repeated runs [27].

Fidelity: The accuracy of a quantum computation compared to an ideal outcome, crucial in assessing the reliability of quantum operations, especially under noise and decoherence [28].

Quantum Reliability: Unlike classical reliability, quantum reliability considers the preservation of quantum states and the ability to reproduce process trajectories rather than mere binary outcomes. Metrics now extend toward trajectory distinguishing, rather than just state fidelity, to cover the probabilistic quantum domain [28].

Security Against Quantum Attacks: Using quantum-resistant cryptography and measuring against quantum attack threats is crucial, especially for hybrid systems integrating blockchain and quantum nodes [29].

3.3.2. Initial Design of Analytical Models for E2Q Dependability Assessment

Dependability can be assessed through a variety of analytical models, each tailored to address specific aspects of system reliability, availability, and performance in complex, dynamic environments:

Queueing Networks are one such framework, designed to analyze the behavior of systems under load by modeling the flow of tasks or requests service nodes; they are particularly effective for evaluating performance metrics such as response times, throughput, and overload probabilities, as they account for the stochastic nature of task arrivals and service durations.

Stochastic Petri Nets and Process Algebras provide another approach, enabling the simulation of systems where failures and repairs occur probabilistically; these models incorporate randomness and concurrency, allowing researchers to capture the intricate interactions between components and predict how stochastic events, such as component failures or recovery processes, impact the overall system.

Markov and Semi-Markov Models are widely employed for their ability to quantify reliability, availability, and mean time to failure by representing system states and transitions in a probabilistic manner; Markov models assume memoryless transitions, where the future state depends only on the current state, while Semi-Markov Models extend this by incorporating time-dependent transitions, allowing for a more nuanced analysis of systems with memory or non-exponential failure distributions [30, 31, 32].

Finally, **Formal Model Transformation** offers a bridge between high-level software design and mathematical analysis, enabling the conversion of annotated models, such as those expressed in UML or SysML, into formal mathematical representations that can be solved for dependability properties; this approach integrates dependability analysis directly into the design lifecycle, allowing for iterative feedback and improvements by leveraging specialized solvers to verify properties like fault tolerance or failures [30].

Together, these models provide a comprehensive toolkit for assessing dependability, addressing both the deterministic and stochastic challenges inherent in modern systems.

3.3.3. Initial Design of Mechanisms and Strategies to Define and Enforce Dependability Policies

Dependability in complex systems, particularly in the E2Q Continuum, relies on a combination of general strategies and specialized mechanisms to ensure resilience, reliability, and security. These approaches are designed to mitigate risks arising from failures, environmental noise, and operational uncertainties, while also adapting to emerging technologies. By integrating both classical and quantum-specific methodologies, systems can achieve robust performance, minimize downtime, and maintain integrity under varying conditions.

General strategies for enhancing dependability include **replication**, which involves executing tasks across multiple nodes to ensure continuity in the event of a failure; this redundancy ensures that if one node malfunctions, another can seamlessly take over, thereby maintaining service availability. **Redundancy** extends this principle by maintaining additional hardware or software resources in standby, replacement of failed components and reducing the risk of system-wide outages. **Fault detection** mechanisms play a critical role in identifying anomalies in real time, often through continuous monitoring of system behavior, and trigger automated switches to alternate nodes or workflows to prevent service disruption. Diversity further strengthens dependability by employing heterogeneous hardware or software solutions, which reduces the likelihood of correlated failures that could arise from using identical components or configurations. Together, these strategies form a foundational framework for building resilient systems capable of challenges [26].

In the realm of quantum computing, where the inherent fragility of quantum states and susceptibility to noise pose unique challenges, specialized mechanisms are employed to ensure dependability. **Quantum Error Correction Codes** are a cornerstone of this effort, utilizing logical qubits encoded across multiple physical qubits to detect and correct errors caused by decoherence, thermal fluctuations, or other forms of quantum noise; these codes enable the preservation of quantum information despite imperfections in the physical system. **Entanglement Purification** addresses the degradation of quantum states during processing by refining the fidelity of entangled pairs, ensuring that quantum information remains accurate and reliable over long distances or complex operations. **Quantum Reproducibility Analysis** complements these efforts by quantifying the consistency of quantum circuit outputs through statistical measures such as Hellinger distances, allowing researchers to adjust parameters like sample sizes or error thresholds to enhance the reliability of probabilistic quantum computations [27]. These quantum-specific mechanisms are critical for advancing the practical viability of quantum systems, dependability in real-world applications.

By combining general strategies with quantum-specific innovations, systems can achieve a balanced approach to dependability, addressing both the universal challenges of reliability and the unique demands of quantum technologies. This dual focus not only enhances the robustness of existing infrastructures but also paves the way for the development of next-generation systems that can operate efficiently in increasingly complex and dynamic environments.

3.3.4. Initial Design of Scheduling Algorithms for Dependable Task Execution in the E2Q Continuum

In the context of advanced edge-to-cloud systems, effective task scheduling is critical to ensuring dependability, optimizing resource utilization, and meeting the demands of heterogeneous environments. As the 3D Continuum integrates quantum components for specialized tasks, scheduling strategies must adapt to address challenges such as latency, resource constraints, and dynamic workloads. By leveraging a combination of classical-aware algorithms, modern scheduling frameworks enable efficient allocation of tasks, enhance system resilience, and support the seamless operation of distributed applications. The following strategies and models form a cohesive toolkit for achieving these goals.

Task Scheduling with Enhanced Whale Optimization Algorithm (EWOA) is a bio-inspired optimization technique that mimics the foraging behavior of humpback whales to allocate resources and tasks in edge and cloud environments. By iteratively refining solutions based on fitness functions that prioritize minimizing execution time and reducing resource costs, EWOA addresses server-side overload and optimizes configurations of resource nodes, ensuring efficient utilization of limited edge and cloud capabilities [33]. **Cooperative Cloud-Edge Scheduling**, extends this approach by dynamically distributing workloads across edge, cloud, and, where applicable, quantum nodes. This strategy leverages real-time metrics such as latency, resource availability, and dependability constraints to balance computational demands, ensuring that tasks are executed in the most suitable environment while maintaining system stability. **Priority- and QoS-Based Scheduling** further refines this process by assigning high-priority tasks to low-latency edge nodes, which are optimized for rapid response, while offloading computationally intensive or security-sensitive workloads to other nodes, which offer superior processing power or cryptographic capabilities. This tiered approach ensures that critical operations are prioritized, while specialized tasks benefit from the unique advantages of quantum infrastructure. Finally, **Markov Decision Processes (MDPs) or Reinforcement Learning-Based Scheduling** introduces a dynamic, adaptive layer to task allocation, enabling systems to learn from historical data and adjust to uncertainties in heterogeneous environments. By modeling scheduling decisions as sequential decision-making problems, these methods optimize long-term performance metrics such as throughput, energy efficiency, and fault tolerance, even in the face of unpredictable workloads or resource fluctuations.

These strategies and models provide a cohesive toolkit for enforcing and evaluating dependability in advanced, distributed edge-to-cloud systems, especially as quantum components are incorporated for specialized tasks. By combining optimization algorithms, hybrid resource management, priority-driven allocation, and adaptive learning techniques, modern scheduling frameworks address the complexities of heterogeneous environments, ensuring robustness, scalability, and efficiency. As quantum technologies continue to evolve, the integration of these strategies will be essential for unlocking the full potential of next-generation computing infrastructures [26, 27, 28, 30, 33].

Task 3.4: E2Q Orchestration

The evolution of distributed computing has driven the development of sophisticated orchestration frameworks to manage resources across cloud, edge, and IoT environments. These systems address the growing complexity of heterogeneous infrastructure while optimizing performance for latency-sensitive applications. Specific orchestration framework platforms like Kubernetes [22] and Apache Mesos [39], as well as computing continuum approaches and solutions like Stack4Things [40], to manage resource allocation, task scheduling, and system configurations based on task requirements are necessary. In this regard, the gap yet to fill is mainly related to the resource type (mainly only processing ones are covered) and quantum nodes, even if some nice solutions have been already implemented [41].

3.4.1. Review of Orchestration Concepts and Tools in Distributed and Cloud Environments

Modern orchestration in cloud-edge continuum environments centers on **cloud-native principles** and **AI-driven decision-making**. Cloud-native computing—emphasizing virtualization, containerization, and microservices—forms the foundation for edge deployment, though challenges persist due to "scarce availability of established tools" [26]. Key concepts include:

- **Quality-of-Service (QoS) Awareness:** Systems prioritize latency-sensitive tasks by dynamically allocating resources closer to data sources [34].
- **Cognitive Resource Management:** Six European projects (CODECO, COGNIFOG, COGNIT, MLSysOps, ENACT, DECICE) have developed intelligent techniques for managing distributed resources across cloud and edge layers [36].

- **Decentralized Orchestration:** Microservices enable "fine-grained task management and adaptability to resource constraints across layers," allowing efficient handling of heterogeneous resources while maintaining service continuity [26].
- **AI-Driven Orchestration:** AI algorithms optimize resource allocation by analyzing real-time data patterns, reducing latency for applications like self-driving cars and smart manufacturing [35].

3.4.2. Initial Design and Development of Mechanisms for Placing and Deploying E2Q Nodes and Services

Placement strategies in modern computing systems emphasize **dynamic resource adaptation** and **centralized control** to optimize performance, reduce latency, and ensure scalability in complex environments. These approaches address the challenges of managing distributed workloads across edge devices, fog nodes, and cloud infrastructure by enabling flexible, real-time adjustments to resource allocation and task execution with adaptive mechanisms, such strategies empower systems to respond to fluctuating demands, maintain service quality, and leverage the strengths of diverse computational layers. This narrative explores three key mechanisms, Edge-to-Cloud Continuum Placement, Cloud-Native Edge Orchestration, and Microservices-Based Deployment, each and resilience in distributed architectures.

Edge-to-Cloud Continuum Placement refers to the strategic mapping of workloads across the spectrum of computing resources, from edge devices (such as sensors and gateways) to cloud servers. This mechanism prioritizes proximity to data sources, ensuring that processing tasks are executed as close to where data is generated as possible. By minimizing the distance data must travel, it reduces latency and bandwidth usage, which is critical for applications requiring real-time responsiveness, such as IoT systems or autonomous vehicles. This the continuum of edge, fog, and cloud layers to balance computational demands, allowing systems to dynamically shift workloads based on factors like network conditions, resource availability, and application-specific constraints [37].

Cloud-Native Edge Orchestration introduces a centralized framework for managing nodes, security protocols, and application deployments while enabling localized data processing at the edge. Unlike traditional architectures that rely on rigid, monolithic structures, this strategy employs cloud-native principles, such as containerization, automation, and microservices, to create a unified yet flexible managementizing control, it ensures consistency in policy enforcement, security measures, and resource allocation across distributed environments, while still allowing edge nodes to handle time-sensitive tasks independently. This dual-layer approach enhances scalability and resilience, making it ideal for scenarios where both global coordination and local autonomy are required [38].

Microservices-Based Deployment decentralizes orchestration mechanisms by breaking down applications into modular, independently deployable services. Each microservice operates as a self-contained unit, resources, such as varying hardware capabilities, network conditions, and computational demands, across edge and cloud layers. This strategy eliminates centralized bottlenecks by enabling peer-to-peer communication and dynamic scaling, allowing systems to adapt to real-time resource constraints and application requirements. By decoupling functions, it also improves fault tolerance and maintainability, as failures in one microservice do not necessarily impact the entire system. This approach, as noted in research on distributed computing, is particularly effective for heterogeneous environments where flexibility and rapid reconfiguration is required [26].

These mechanisms collectively avoid rigid, static topologies by embedding adaptability into their design. By combining dynamic resource allocation, centralized oversight, and decentralized execution, they enable systems to respond to evolving workloads, optimize performance, and maintain reliability in increasingly complex and distributed computing landscapes. As the demand for real-time processing and scalability continues to grow, such strategies will remain pivotal in shaping the future of edge-cloud computing.

Conclusion and Research Implications

The provided context establishes robust frameworks for cloud-edge orchestration using cloud-native, AI-driven, and cognitive approaches, but reveals critical gaps in quantum-specific management and explicit monitoring tooling. Future work must address:

1. Developing quantum-aware orchestration strategies for hybrid quantum-classical workloads.
2. Standardizing monitoring tools for edge-cloud continuum performance in the presence of quantum nodes.
3. Creating open-source frameworks to overcome the "scarce availability of established tools" in edge orchestration [26].

Without quantum-specific references in the literature reviewed, current orchestration systems remain focused on classical edge-cloud applications, leaving quantum integration as an emerging frontier requiring dedicated research.

Conclusions and Next Steps

Work Package 2 explores the potential of the ultrastrong coupling (USC) regime to enhance the dependability of quantum computing nodes through noise suppression, error correction, and gate design. The analysis highlights that while superconducting qubits face decoherence challenges due to environmental interactions, the USC regime offers a pathway to mitigate these issues by leveraging strong, instantaneous interactions. Key strategies include designing logical qubits via multiple physical qubits coupled in the USC regime, which inherently protects against noise and extends coherence times beyond single-qubit limitations. Proposed novel quantum gates, such as the √iSWAP gate enable faster, high-fidelity operations that outpace decoherence. Additionally, USC's unique properties, like virtual photons and quantum nondemolition measurements (QNDM), are investigated for real-time error detection and correction, reducing reliance on external mechanisms. Dependability metrics, such as coherence time, gate fidelity, and error propagation, are defined, supported by analytical simulation frameworks used to evaluate system robustness. Challenges include the complexity of USC systems and accurate modeling of their dynamics, but the work lays the theoretical groundwork for scalable, fault-tolerant quantum architectures. Overall, the task underscores USC's potential to revolutionize quantum computing by combining intrinsic noise protection, rapid operations, and advanced error management, paving the way for next-generation, reliable quantum nodes.

The integration of quantum computing into the edge-to-cloud (E2Q) continuum, which is to be explored in WP3, represents a transformative leap in distributed computing, yet it remains an underexplored frontier. The analysis highlights the critical need for quantum-aware orchestration frameworks, scheduling algorithms, and monitoring tools unique challenges of hybrid quantum-classical systems. While existing orchestration platforms like Kubernetes and Apache Mesos excel in managing classical edge-cloud workloads, they lack explicit support for quantum nodes, creating a significant gap in scalability, dependability, and performance optimization for next-generation applications.

References

[1] L. Bittencourt, R. Immich, R. Sakellariou, N. Fonseca, E. Madeira, M. Curado, L. Villas, L. DaSilva, C. Lee, and O. Rana, “The internet of things, fog and cloud continuum: Integration and challenges,” Internet of Things, vol. 3, pp. 134–155, 2018.
[2] S. Distefano, G. Merlino, and A. Puliafito, “Device-centric sensing: an alternative to data-centric approaches,” IEEE Systems Journal, vol. 11, no. 1, pp. 231–241, 2015.
[3] Y. Jararweh, M. Al-Ayyoub, E. Benkhelifa, M. Vouk, A. Rindos et al., “Software defined cloud: Survey, system and evaluation,” Future Generation Computer Systems, vol. 58, pp. 56–74, 2016.
[4] K. Morris, Infrastructure as code. O’Reilly Media, 2020.
[5] S. Jordan, “Quantum Algorithm Zoo,” 2011-2025. [Online]. Available: https://quantumalgorithmzoo.org/
[6] M. Caleffi, M. Amoretti, D. Ferrari, D. Cuomo, J. Illiano, A. Manzalini, and A. S. Cacciapuoti, “Distributed quantum computing: a survey,” arXiv preprint arXiv:2212.10609, 2022.
[7] S. S. Gill, A. Kumar, H. Singh, M. Singh, K. Kaur, M. Usman, and R. Buyya, “Quantum computing: A taxonomy, systematic review and future directions,” Software: Practice and Experience, vol. 52, no. 1, pp. 66–114, 2022.
[8] D. Castelvecchi, “Ibm’s quantum cloud computer goes commercial,” Nature, vol. 543, no. 7644, 2017.
[9] F. Hu, B.-N. Wang, N. Wang, and C. Wang, “Quantum machine learning with D-Wave quantum computer,” Quantum Engineering, vol. 1, no. 2, p. e12, 2019.
[10] I. L. Markov, A. Fatima, S. V. Isakov, and S. Boixo, “Quantum supremacy is both closer and farther than it appears,” 2018.
[11] W. J. Munro and K. Nemoto, “Quantum fog computing,” in Quantum Communications and Quantum Imaging XXI. SPIE, 2023, p. PC1269201.
[12] Z. Qu, K. Wang, and M. Zheng, “Secure quantum fog computing model based on blind quantum computation,” Journal of Ambient Intelligence and Humanized Computing, vol. 13, no. 8, p. 3807 – 3817, 2022.
[13] A. Furutanpey, J. Barzen, M. Bechtold, S. Dustdar, F. Leymann, P. Raith, and F. Truger, “Architectural vision for quantum computing in the edge- cloud continuum,” in 2023 IEEE International Conference on Quantum Software (QSW). IEEE, 2023, pp. 88–103.
[14] I. B. Djordjevic, Quantum Communication, Quantum Networks, and Quantum Sensing. Academic Press, 2022.
[15] R. Dautov and S. Distefano, “Automating IoT data-intensive application allocation in clustered edge computing,” IEEE Transactions on Knowledge and Data Engineering, vol. 33, no. 1, pp. 55–69, 2019.
[16] H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf, “Quantum fingerprinting,” Physical review letters, vol. 87, no. 16, p. 167 902, 2001.
[17] F. Ablayev, M. Ablayev, A. Vasiliev, and M. Ziatdinov, “Quantum Fingerprinting and Quantum Hashing. Computational and Cryptographical Aspects,” Baltic J. Modern Computing, vol. 4, no. 4, pp. 860–875, 2016. DOI : 10.22364/bjmc.2016.4.4.17.
[18] F. Ablayev, N. Salikhova, and M. Ablayev, “Hybrid classical–quantum text search based on hashing,” Mathematics, vol. 12, no. 12, 2024, ISSN: 2227-7390. DOI: 10 . 3390 / math12121858. [Online]. Available: https://www.mdpi.com/2227-7390/12/12/1858.

[19] J. Garcia-Alonso, J. Rojo, D. Valencia, E. Moguel, J. Berrocal, and J. M. Murillo, “Quantum software as a service through a quantum API gateway,” IEEE Internet Computing, vol. 26, no. 1, pp. 34–41, 2021.
[20] “Netflix TechBlog.” [Online]. Available: https://netflixtechblog.com/
[21] “Apache Airflow.” [Online]. Available: https://airflow.apache.org/
[22] "kubernetes.” [Online]. Available: https://kubernetes.io/docs/
[23] “docker.” [Online]. Available: https://docs.docker.com/
[24] “AWS Lambda.” [Online]. Available: https://aws.amazon.com/lambda/
[25] “Azure Functions.” [Online]. Available: https://learn.microsoft.com/en-us/azure/azure-functions/
[26] L. Belcastro, F. Marozzo, A. Orsino, D. Talia, and P. Trunfio. "Navigating the Edge-Cloud Continuum: A State-of-Practice Survey." arXiv preprint arXiv:2506.02003 (2025).
[27] E. Giusto, S. Núñez-Corrales, K.N. Smith, P. Cao, E. Younis, P. Rech, F. Vella, B. Baheri, A. Cilardo, B. Montrucchio, W. Jiang, S. Xu, S. Dasgupta, R.K. Iyer and T.S. Humble. "Dependable classical-quantum computing systems engineering." Front. Comput. Sci. 7:1520903. DOI: 10.3389/fcomp.2025.1520903 (2025).
[28] L.X. Cui, Y-M. Du, and C.P. Sun. "Quantum Reliability." Phys. Rev. Lett. 131, 160203. DOI: 10.1103/PhysRevLett.131.160203 (2023).
[29] J. Seol, J. Kim. "Quantum threat and dependability of quantum-safe blockchain-based distributed control systems and network." Issues in Information Systems. Volume 24, Issue 3, pp. 144-158. DOI: 10.48009/3_iis_2023_113 (2023).
[30] S. Bernardi, J. Merseguer, D.C. Petriu, "Dependability Modeling and Assessment in UML-Based Software Development." The Scientific World Journal, 614635. DOI: 10.1100/2012/614635 (2012).
[31] J.A. Faria, M.A.Matos. "An analytical methodology for the dependability evaluation of non-Markovian systems with multiple components". Reliability Engineering & System Safety, Volume 74, Issue 2, pp. 193-210. DOI: 10.1016/S0951-8320(01)00073-4 (2001).
[32] V. Wasnik, A. Tchangani, C. Martin, and F. Pérès. "A model for dependability analysis of a complex system." IFAC-PapersOnLine. Volume 58, Issue 8, pp. 97-102. DOI: 10.1016/j.ifacol.2024.08.156 (2024).
[33] L. Han, S. Zhu, H. Zhao, and Y. He. "An enhanced whale optimization algorithm for task scheduling in edge computing environments." Front. Big Data 7:1422546. DOI: 10.3389/fdata.2024.1422546 (2024).
[34] A. Orive, A. Agirre, H.-L. Truong, I. Sarachaga, and M. Marcos. "Quality of Service Aware Orchestration for Cloud–Edge Continuum Applications." Sensors, 22, 1755. DOI: 10.3390/s22051755 (2022).
[35] V. Ramamoorthi. "Exploring AI-Driven Cloud-Edge Orchestration for IoT Applications." International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), Volume 9, Issue 5, pp.385-393. DOI: 10.32628/CSEIT239072 (2023).
[36] J. Soldatos, R.C. Sofia, S. Azaiez, P. Townend, U. Wajid, C. Antonopoulos, and F. Stein. "Cognitive Management and Orchestration of Resources in the Edge Cloud Continuum-White Paper v1.1." (2025). https://zenodo.org/records/14629440/files/EdgeCloudOrchestration-WhitePaper_V1.1.pdf
[37] M.M. Hasan, T. Sultana, M.D. Hossain, A.K. Mandal, T.-T. Ngo, G.-W. Lee, and E.-N. Huh. "The journey to cloud as a continuum: Opportunities, challenges, and research directions." ICT Express. DOI: 10.1016/j.icte.2025.04.015 (2025).

[38] Y. Wu. "Cloud-Edge Orchestration for the Internet-of-Things: Architecture and AI-Powered Data Processing." IEEE Internet of Things Journal PP(99):1-1. DOI: 10.1109/JIOT.2020.3014845 (2020).
[39] “Apache Mesos,” 2012-2022. [Online]. Available: https://mesos.apache.org/
[40] F. Longo, D. Bruneo, S. Distefano, G. Merlino, and A. Puliafito, “Stack4things: a sensing-and-actuation-as-a-service framework for iot and cloud integration,” Annals of Telecommunications, vol. 72, pp. 53–70, 2017.
[41] Will Cunningham et al. (2025). AgnostiqHQ/covalent: v0.238.0-rc.0 (v0.238.0-rc.0). Zenodo. https://doi.org/10.5281/zenodo.14977421