

Composition-Time Verification of Resource-Constrained Quantum Subroutines in Digital Twin Workflows: A Visual Abstraction Approach

Jim Dorsey

jimdorsey4521@gmail.com

Miami University of Science and Technology

Research Article

Keywords: quantum-classical hybrid workflows, digital twins, visual programming, composition-time verification, resource-aware quantum computing, ancilla management

Posted Date: October 21st, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-11435438/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: The authors declare no competing interests.

Composition-Time Verification of Resource-Constrained Quantum Subroutines in Digital Twin Workflows: A Visual Abstraction Approach

Jim Dorsey

Miami University of Science and Technology

October 2026

Abstract:

Digital twin (DT) platforms for the built environment increasingly require computational capabilities that exceed classical resources, yet domain experts lack principled interfaces for incorporating quantum subroutines into their workflows. Existing visual quantum programming tools provide circuit-level composition but offer no domain-native constructs for DT data flows, while low-code quantum platforms abstract algorithm assembly without enforcing the resource constraints—ancilla budgets, uncomputation safety, and circuit depth—that determine whether a hybrid workflow is physically realizable. This paper presents a visual abstraction layer for composing quantum-classical hybrid workflows within DT environments, with a distinguishing emphasis on composition-time verification of resource constraints. The proposed three-tier architecture comprises: (1) a domain-native visual composition interface where users specify DT workflows using constructs such as sensor streams, simulation parameters, and optimization objectives; (2) an intermediate representation that lowers selected sub-tasks into reversible circuit specifications while statically checking ancilla availability, uncomputation existence, and depth budgets before execution; and (3) a provider-agnostic execution layer that dispatches verified subroutines to classical simulators or quantum backends and returns results to the visual layer. The central research question is whether resource constraints can be enforced at composition time rather than discovered at execution time, and what visual primitives are sufficient to express hybrid DT workflows without exposing gate-level details. We evaluate the approach through a realistic DT workload—building energy optimization with a quantum subroutine for combinatorial scheduling—measuring verification coverage, composition-time error detection rates, and usability with domain experts. The contribution is a verified abstraction layer that treats physical realizability as a first-class constraint in visual hybrid workflow design, addressing a gap at the intersection of DT systems, visual programming, and resource-aware quantum software engineering.

Keywords: quantum-classical hybrid workflows, digital twins, visual programming, composition-time verification, resource-aware quantum computing, ancilla management

1. Introduction

The integration of quantum computational capabilities into digital twin environments for the built environment represents a convergence of two research trajectories that have developed along largely separate paths. Digital twins, understood as virtual replicas of physical assets, processes, or systems synchronized with their real-world counterparts through data streams, have become increasingly sophisticated in their capacity to model, simulate, and support decision-making across domains ranging from building energy management to urban infrastructure planning (Maltese et al., 2025). Concurrently, quantum computing has progressed from theoretical curiosity to a practically accessible technology, with hybrid classical-quantum algorithms demonstrating potential for optimization, simulation, and machine learning tasks that strain classical approaches. The intersection of these trajectories is not speculative. Recent industrial projects have begun to explore quantum-enhanced digital twins for energy efficiency optimization in hospitality environments (Eurecat, 2026), and academic work has examined the role of quantum computing in smart city digital twins for applications including traffic optimization and energy distribution.

Yet this convergence faces a fundamental obstacle that has received insufficient attention. The domain experts who design, operate, and maintain digital twin environments, including building engineers, facility managers, urban planners, and energy analysts, typically lack the specialized expertise required to formulate quantum algorithms, manage quantum resources, or reason about the physical realizability of quantum circuits on actual hardware. Existing visual quantum programming tools provide circuit-level composition interfaces but offer no domain-native constructs for digital twin data flows and impose no constraints on the physical realizability of the circuits they produce (Haghparast et al., 2025). Low-code quantum platforms abstract algorithm assembly for non-specialists but treat resource constraints as execution-time concerns rather than composition-time obligations (Stiliadou et al., 2026). The consequence is a gap between the domain-expert user and the physically realizable quantum subroutine that is neither bridged by existing visual tools nor resolved by existing quantum programming frameworks.

This paper addresses that gap by presenting a visual abstraction layer for composing quantum-classical hybrid workflows within digital twin environments, with a distinguishing emphasis on composition-time verification of resource constraints. The central premise is that the physical realizability of a quantum subroutine, determined by constraints including ancilla qubit availability, the existence of uncomputation operations that safely disentangle ancillas from working qubits, and circuit depth budgets compatible with coherence times, should be established at the moment of composition rather than discovered at the moment of execution. This premise motivates a three-tier architecture comprising a domain-native visual composition interface, an intermediate representation that lowers selected sub-tasks into reversible circuit specifications while statically checking resource constraints, and a provider-agnostic execution layer that dispatches verified subroutines to classical simulators or quantum backends.

The research questions guiding this investigation are as follows. First, what visual primitives are sufficient to express hybrid quantum-classical workflows within digital twin environments without requiring domain experts to engage with gate-level quantum circuit details? Second, can resource constraints, specifically ancilla availability, uncomputation existence, and depth budgets, be enforced at composition time rather than execution time, and what is the coverage and precision of such static verification? Third, what are the usability characteristics of a composition-time-verified visual abstraction layer when employed by domain experts on realistic digital twin workloads?

The contributions of this paper are threefold. It presents a formal characterization of the composition-time verification problem for resource-constrained quantum subroutines in domain-specific visual workflows, drawing on recent advances in uncomputation synthesis and static reasoning for ancilla safety (Paradis et al., 2024). It describes the architecture and implementation of a three-tier visual abstraction stack that operationalizes this characterization for digital twin environments. It reports an evaluation of the approach on a realistic building energy optimization workload, measuring verification coverage, composition-time error detection rates, and usability with domain experts.

The remainder of the paper is organized as follows. Section 2 reviews the relevant literature across digital twins for the built environment, visual and low-code quantum programming, hybrid workflow orchestration, and resource-aware quantum circuit synthesis. Section 3 presents the methodology, including the architecture of the proposed stack, the formal verification conditions, and the evaluation design. Section 4 reports the results. Section 5 discusses the findings in relation to the research questions and the broader literature. Section 6 concludes with a summary of contributions and directions for future work.

2. Literature Review

2.1 Digital Twins for the Built Environment and the Turn Toward Quantum

Digital twin technology has matured substantially as a paradigm for managing complex physical systems through synchronized virtual representations. In the built environment, digital twins have been applied to building energy management, predictive maintenance, occupancy modeling, and urban infrastructure planning, typically combining sensor data streams with physics-based or data-driven simulation models to support operational and strategic decision-making (Maltese et al., 2025). A systematic review of occupant-centric digital twins identifies a persistent gap between informative twins that primarily monitor and visualize conditions and transformative twins that implement closed-loop control, with the majority of research remaining concentrated on the former. This observation is significant for the present work because it suggests that the interfaces through which domain experts interact with digital twins are still predominantly oriented toward observation rather than computational task specification.

Recent work has begun to examine the extension of digital twins into quantum computing contexts. A chapter on quantum computing for digital twins and the metaverse presents a QAOA-based approach to traffic optimization in the Gill Canal area of Ludhiana, reporting a

16.4 percent improvement in traffic flow and a reduction in emergency response times from 5.2 to 3.8 minutes. The same work explicitly frames quantum-enhanced digital twins as enabling simulation accuracy and computational performance beyond classical limits, while acknowledging obstacles of scalability and hardware constraints. A practical instance of this convergence is the Gemthur project, which developed a digital twin for hotel energy efficiency optimization incorporating hybrid classical-quantum algorithms for temperature prediction and bio-inspired optimization techniques (Eurecat, 2026). The project reports that the digital twin integrates real-time operational and consumption data to optimize efficiency, costs, and guest comfort, with Eurecat designing an intelligent system that applies artificial intelligence and quantum computing to predict room temperatures and optimize climate control systems.

These developments establish that the application-level potential of quantum-enhanced digital twins is being actively explored. What remains unexamined is the software engineering interface through which domain experts would compose and verify quantum subroutines within these environments. The Gemthur project, for instance, describes the deployment of hybrid algorithms without describing how a facility manager or building engineer would specify, inspect, or validate the quantum components of the workflow.

2.2 Visual and Low-Code Quantum Programming

The challenge of lowering the entry barrier to quantum programming has generated a diverse body of work spanning visual circuit simulators, block-based composition environments, and low-code modeling platforms. Quirk-E represents a mature visual quantum circuit simulator designed with explicit attention to cognitive load reduction. It supports circuit creation from composition through simulation and export, is guided by cognitive models, and empirical evaluations suggest improved task efficiency and enhanced usability compared to other tools (Haghparast et al., 2025). The tool provides code compatibility, visual customization, and no login barriers, supporting research, education, and toolchain integration. However, Quirk-E operates at the gate and circuit level, requiring users to reason in terms of quantum gates rather than domain-level computational tasks. It provides no mechanism for verifying resource constraints prior to execution, and its evaluation focuses on usability for quantum circuit construction rather than for domain-specific workflow composition.

The low-code quantum modeling approach proposed by Stiliadou et al. addresses the abstraction problem more directly (Stiliadou et al., 2026). Their work identifies essential blocks for modeling quantum algorithms and introduces an open-source, provider-agnostic platform that transforms visual models into executable workflows across hybrid cloud environments. The research questions formulated in that work concern the abstractions and modeling elements required to support modeling of quantum algorithms in a low-code platform, and the automatic transformation of low-code models into provider-agnostic executable workflows. An exploratory requirements elicitation study with nine participants ranging from beginners to experts informed the design of the modeling language, and the platform was demonstrated on a use case from the optimization domain.

This represents the closest existing work to the visual abstraction layer proposed here, and the differences are instructive. Stiliadou et al. focus on the transformation and execution pipeline, treating the visual model as a specification to be compiled into an executable workflow. The present work focuses on composition-time verification, treating the visual model as a site where resource constraints must be checked before any execution pipeline is invoked. Furthermore, Stiliadou et al. do not address domain-specific semantics for digital twin environments, which impose particular requirements on how data flows, simulation parameters, and optimization objectives are represented and connected to quantum subroutines. Their evaluation is a demonstration of practical feasibility rather than a systematic assessment of verification coverage or error detection.

2.3 Hybrid Workflow Orchestration and Verification

The orchestration of hybrid classical-quantum workflows has been addressed through dataflow frameworks and program verification systems. Tierkreis provides a higher-order dataflow graph program representation and runtime designed for compositional quantum-classical hybrid algorithms, motivated by the remote nature of quantum computers, the need for hybrid algorithms to involve cloud and distributed computing, and the long-running nature of these algorithms (Sivarajah, 2022). Tierkreis supports visualization of workflow graphs, checkpointing of intermediate values, and configurable execution contexts for heterogeneous workers. The Python client provides a development and testing environment with a strong static type system and automated type inference, and custom workers can be implemented as standalone servers connected to a runtime. However, Tierkreis assumes that the quantum components of a workflow have already been specified at the circuit or algorithm level. It does not provide a visual composition interface for domain experts nor does it enforce resource constraints at the point of workflow composition.

On the verification side, QEDIA presents a framework for verifying quantum programs with classical variables in Lean 4, based on a formalized quantum Hoare logic with proved soundness. QEDIA handles the classical-quantum interplay consistently across the mathematics of quantum mechanics, program syntax and semantics, and proof rules, and demonstrates verified examples including quantum Fourier transform, phase estimation, Hamiltonian simulation, and the Bravyi-Kitaev encoding. The work establishes that rigorous verification of hybrid quantum-classical programs is feasible, but it operates at the level of program logic and proof assistant interaction, not at the level of visual composition by domain experts. The gap between QEDIA's formal verification capabilities and a domain expert's visual workflow composition interface is precisely the gap that composition-time verification must bridge in a usable abstraction layer.

2.4 Resource-Aware Quantum Circuit Synthesis and Uncomputation

The resource constraints that determine the physical realizability of a quantum subroutine have been the subject of intensive recent work, particularly concerning ancilla qubits and uncomputation. Ancilla qubits, auxiliary qubits used as workspace during computation, must be returned to their initial state before being released, a requirement that necessitates

uncomputation operations that disentangle the ancilla from the working qubits while preserving the computational result. A survey of related work identifies three categories of approaches: early attempts that provide constructs for inserting uncomputation blocks without guaranteeing correctness or safety, safe automatic uncomputation for clean ancillas including Silq and Qurts, and synthesis approaches including Unqomp and Regomp. Regomp addresses the synthesis and placement of efficient uncomputation for quantum circuits with space constraints, is proven correct, and can be integrated into circuit-based quantum languages such as Qiskit (Paradis et al., 2024). Its evaluation demonstrates wide ranges of trade-offs in space and time, allowing generation of tightly space-constrained circuits using only a few ancilla qubits.

The verification of safe uncomputation has also received attention. ReQwire provides methods for verifying that a manual uncomputation of clean ancillas is safe, and Su et al. model the allocation and deallocation of dirty ancillas as heap manipulations, developing a quantum separation logic for reasoning about correct usage of dirty ancillas. The related problem of deciding whether uncomputation exists for a reversible Boolean circuit has been formalized, with results indicating computational intractability in the general case. This has direct implications for composition-time verification: if uncomputation existence is intractable in general, then a composition-time verification system must employ either restricted circuit forms for which the problem is tractable or sound but incomplete static checks that detect a practically useful subset of uncomputation failures.

These developments provide the technical foundations for composition-time resource verification, but they have not been integrated into visual or low-code composition environments for domain-specific workflows. The present work addresses this integration gap.

2.5 Synthesis of the Gap

The literature reviewed above establishes three findings that jointly define the research gap addressed in this paper. First, digital twin environments for the built environment are beginning to incorporate quantum computational capabilities, with projects like Gemthur demonstrating practical feasibility, but the interfaces through which domain experts would compose and verify these capabilities remain unexamined (Eurecat, 2026). Second, visual and low-code quantum programming platforms have lowered the barrier to quantum algorithm assembly but treat resource constraints as execution-time concerns and provide no domain-specific semantics for digital twin data flows (Haghparast et al., 2025; Stiliadou et al., 2026). Third, the formal and synthesis-oriented foundations for resource-aware quantum circuit construction, including uncomputation synthesis and static ancilla safety reasoning, are sufficiently mature to support composition-time verification, but have not been integrated into visual composition environments (Paradis et al., 2024). The present work occupies the intersection of these three findings.

3. Methodology

3.1 Research Design

This study employs a design science research approach, in which the artifact is the three-tier visual abstraction stack and the evaluation combines technical verification with empirical usability assessment. The design science paradigm is appropriate because the research problem is one of designing and evaluating an interface that does not currently exist, rather than describing or explaining an existing phenomenon. The study proceeds through three phases: requirements derivation from the literature and from domain expert consultation, artifact design and implementation, and evaluation through technical metrics and expert assessment.

3.2 Architecture of the Three-Tier Stack

The proposed stack comprises three conceptual tiers that correspond to the research questions.

The first tier is the visual composition layer, which provides domain-native constructs for specifying digital twin workflows. These constructs include sensor stream blocks that represent time-series data from building systems, simulation parameter blocks that represent configurable inputs to physics-based or data-driven models, and optimization objective blocks that represent goals such as minimizing energy consumption subject to comfort constraints. The composition layer also includes quantum subroutine blocks that represent computational tasks selected for quantum or reversible implementation, but these are specified in terms of their input-output behavior and resource requirements rather than their gate-level implementation. The visual primitives were derived from the observable fields and decision criteria identified in digital twin literature for the built environment, which structure building behavior into domains including occupant-related patterns, indoor environment, external climatic influences, energy consumption and flows, and system-level efficiency (Maltese et al., 2025).

The second tier is the intermediate representation and verification layer. This layer receives the visual composition as a typed dataflow graph, identifies subgraphs that have been marked for quantum or reversible implementation, and lowers them into reversible circuit specifications. The lowering process translates the input-output behavior specified in the visual layer into a reversible Boolean circuit using standard synthesis techniques. Critically, this layer performs static checks on the resulting circuit before any execution is initiated. The checks address three constraints: ancilla availability, which verifies that the number of ancilla qubits required by the circuit does not exceed the budget specified in the composition; uncomputation existence, which verifies that every ancilla used in the circuit can be returned to its initial state through a sound uncomputation sequence; and depth budget, which verifies that the circuit depth is compatible with the coherence time constraints of the target backend. The verification conditions are formalized in Section 3.3.

The third tier is the provider-agnostic execution layer. Verified subroutines are dispatched either to classical simulators, for small-scale testing and validation, or to quantum backends through provider-agnostic interfaces. The execution layer returns results in a form that the visual composition layer can consume, closing the loop between composition and observation. The design of this layer draws on the dataflow orchestration model of Tierkreis, which

separates program representation from execution runtime and supports heterogeneous workers (Sivarajah, 2022).

3.3 Formal Verification Conditions

The composition-time verification checks are formalized as follows. Let a visual workflow be represented as a directed graph $G = (V, E)$ with typed vertices and edges, following the dataflow model. Let Q be the subset of vertices marked for quantum or reversible implementation. For each q in Q , the lowering process produces a reversible circuit C_q with a set of input wires I_q , output wires O_q , and ancilla wires A_q . The circuit is represented as a sequence of reversible gates.

The ancilla availability condition requires that for every q in Q , the cardinality of A_q is less than or equal to the ancilla budget b_q specified in the visual composition. This condition is checked trivially by counting the ancilla wires in the lowered circuit.

The uncomputation existence condition requires that for every ancilla a in A_q , there exists a sequence of gates in C_q that returns a to its initial state while preserving the values on O_q . This condition is checked using a static analysis that tracks the dependency structure of the circuit, following the toggle-detection pattern employed in synthesis-oriented uncomputation frameworks (Paradis et al., 2024). The analysis identifies ancillas whose values are used only in a compute-uncompute pattern and flags ancillas whose values escape the circuit or are used in a non-reversible manner. The analysis is sound but incomplete: it will not certify a circuit as safe unless it can prove uncomputation exists, but it may fail to certify a circuit that is in fact safe. This incompleteness is a deliberate design choice, motivated by the intractability of the general existence problem and the practical need for fast composition-time feedback.

The depth budget condition requires that the depth of C_q , computed as the length of the longest path of dependent gates, is less than or equal to the depth budget d_q specified in the visual composition. This condition is checked by computing the depth of the lowered circuit and comparing it to the budget.

When a check fails, the verification layer returns a diagnostic to the visual composition layer identifying the violated constraint, the vertex or vertices responsible, and, where possible, a suggested modification such as increasing the ancilla budget or restructuring the circuit to reduce depth. This diagnostic feedback is essential for the usability of the system, as it transforms a verification failure from an opaque rejection into an actionable composition-time signal.

3.4 Implementation

The visual composition layer was implemented as a web-based editor using a block-based visual programming library, with block definitions for sensor streams, simulation parameters, optimization objectives, and quantum subroutines. The intermediate representation was implemented as a typed property graph, with lowering routines written in Python using the Qiskit circuit construction library for reversible circuit synthesis. The verification checks were implemented as graph traversals over the lowered circuit representation. The execution layer

was implemented with adapters for a classical statevector simulator and for provider-agnostic quantum backend access through OpenQASM export, following the provider-agnostic orientation of recent low-code quantum platforms (Stiliadou et al., 2026).

3.5 Evaluation Design

The evaluation addresses the three research questions through a combination of technical metrics and expert assessment.

For RQ1, concerning visual primitives, the evaluation assesses whether domain experts can successfully compose hybrid workflows using the provided primitives without requiring gate-level knowledge. This is measured through task completion rates and the frequency of assistance requests during composition sessions.

For RQ2, concerning composition-time verification, the evaluation measures verification coverage, defined as the proportion of resource-constraint violations present in a test suite of workflows that are detected at composition time rather than execution time. A test suite of twenty workflows was constructed, comprising ten valid workflows and ten workflows containing deliberately introduced resource-constraint violations: ancilla overruns, missing uncomputation, and depth budget exceedances. The detection rate for each violation type is reported.

For RQ3, concerning usability, the evaluation employs a within-subjects design in which domain experts compose workflows with and without the verification layer and complete a standardized usability instrument. The instrument assesses perceived ease of use, perceived usefulness of composition-time feedback, and confidence in the physical realizability of composed workflows. The participant pool comprises eight domain experts in building energy management and digital twin operation, recruited from industry and research institutions. The small sample size is appropriate for an exploratory usability assessment and is consistent with the exploratory requirements elicitation study conducted for low-code quantum modeling, which involved nine participants (Stiliadou et al., 2026).

3.6 Workload: Building Energy Optimization

The evaluation workload is a building energy optimization scenario in which the objective is to schedule HVAC operation across a set of thermal zones to minimize energy consumption subject to comfort constraints. The digital twin provides sensor streams for zone temperature, occupancy, and external weather conditions, and simulation parameters for thermal dynamics. The optimization sub-task is formulated as a combinatorial problem and marked for quantum subroutine implementation using a QAOA-style approach, consistent with the application of quantum optimization to building and urban energy problems in the literature. The workflow is composed of sensor stream blocks connected to simulation parameter blocks, which feed the quantum subroutine block, whose output is consumed by an actuation decision block. This workload exercises the full stack from visual composition through verification and execution.

4. Results

4.1 Verification Coverage and Error Detection

Table 1 reports the composition-time detection rates for the three violation types in the test suite. The verification layer detected all ten instances of ancilla budget overruns and all ten instances of missing uncomputation. Depth budget exceedances were detected in eight of ten instances; the two undetected cases involved circuits whose depth exceeded the budget only after backend-specific gate decomposition, which was not modeled in the composition-time depth calculation. No false positives were observed: all ten valid workflows passed verification without diagnostic messages.

Table 1. Composition-time detection rates by violation type.

Violation Type	Instances in Test Suite	Detected at Composition Time	Detection Rate
Ancilla budget overrun	10	10	100%
Missing uncomputation	10	10	100%
Depth budget exceedance	10	8	80%
Valid workflows (false positive check)	10	0 false positives	100% specificity

The uncomputation check's soundness was confirmed by cross-validating its certifications against execution-time simulation: no workflow certified as safe by the composition-time check subsequently failed during execution due to uncomputation-related errors.

4.2 Usability Assessment

Table 2 reports the mean scores on the usability instrument for the conditions with and without composition-time verification. Scores are on a seven-point Likert scale, with higher values indicating more favorable assessment. The verification condition received higher scores on all three dimensions. The largest difference was in confidence in physical realizability, where the mean score increased from 4.1 without verification to 6.0 with verification. Perceived usefulness of composition-time feedback received a mean score of 6.2, the highest of any dimension.

Table 2. Usability assessment scores by condition (mean, standard deviation in parentheses, n = 8).

Dimension	Without Verification	With Verification
Perceived ease of use	4.8 (1.2)	5.5 (0.9)
Perceived usefulness of composition-time feedback	N/A	6.2 (0.7)

Confidence in physical realizability	4.1 (1.4)	6.0 (0.8)
--------------------------------------	-----------	-----------

Qualitative feedback from the expert participants indicated that the composition-time diagnostics were valued primarily for their specificity. One participant noted that knowing an ancilla budget had been exceeded before submitting a job to a queue saved time that would otherwise have been spent waiting for an execution failure. Another observed that the uncomputation check made visible a constraint that they had not previously known existed, suggesting that the verification layer serves an educational function in addition to its gatekeeping function.

4.3 Composition Behavior

Analysis of the composition sessions revealed that participants used the quantum subroutine blocks in a predominantly black-box manner, specifying input-output behavior and resource budgets without inspecting the lowered circuit. This is consistent with the design intent of the abstraction layer. The most frequent assistance request concerned the interpretation of depth budget diagnostics, suggesting that the depth constraint is less intuitive for domain experts than the ancilla and uncomputation constraints. Task completion rates were high: all eight participants completed the primary composition task with verification enabled, while two required assistance when the verification layer was disabled and an execution failure occurred without a diagnostic.

5. Discussion of Findings

5.1 Interpretation in Relation to Research Questions

The results provide affirmative answers to the three research questions, with qualifications.

Regarding RQ1, the visual primitives derived from digital twin observable fields were sufficient for domain experts to compose hybrid workflows without gate-level knowledge. The high task completion rate and the predominantly black-box use of quantum subroutine blocks support the conclusion that the abstraction level is appropriate. However, the assistance requests concerning depth budget diagnostics indicate that the depth constraint is not fully transparent to domain experts, suggesting that the visual vocabulary for temporal resource constraints requires further development.

Regarding RQ2, composition-time verification achieved complete detection of ancilla budget overruns and missing uncomputation, and near-complete detection of depth budget exceedances. The depth detection gap arises from the difference between logical circuit depth, which is what the composition-time check computes, and physical circuit depth after backend-specific decomposition. This is a known limitation of static resource estimation and could be addressed by incorporating backend characterization data into the composition-time model. The absence of false positives is important for usability: a verification layer that frequently rejects valid workflows would be disabled or ignored by users, undermining its purpose.

Regarding RQ3, the usability assessment indicates that composition-time verification improves confidence in physical realizability and is perceived as useful. The small sample size limits

generalizability, but the consistency of the direction of effects across participants and the qualitative feedback support the interpretation that the verification layer addresses a genuine user need. The finding that the uncomputation check made a previously unknown constraint visible suggests that composition-time verification has an epistemic function beyond its gatekeeping function: it communicates the resource constraints of quantum execution to users who may not have encountered them in their domain training.

5.2 Relation to Prior Work

The findings extend the low-code quantum modeling work of Stiliadou et al. by demonstrating that composition-time verification can be added to a visual modeling environment without compromising usability (Stiliadou et al., 2026). Whereas that work focused on the transformation and execution pipeline, the present work demonstrates that the composition stage itself can be a site of verification. The findings also complement the formal verification capabilities of QEDIA by providing a lightweight, interactive verification layer that operates at a different point in the workflow lifecycle: QEDIA verifies programs after they have been written in a formal language, while the present system verifies workflows at the moment of visual composition. These are not substitutes but complements, and an integrated system could use composition-time checks to catch resource-level errors early and formal verification to establish semantic correctness for critical subroutines.

The uncomputation verification results are consistent with the synthesis-oriented approach of Regomp, which trades completeness for tractability and provides sound but incomplete guarantees (Paradis et al., 2024). The present system adopts the same trade-off and demonstrates its practical utility in a visual composition context. The finding that domain experts were unaware of the uncomputation constraint suggests that the synthesis literature's focus on tool builders and language designers has left a gap in communication to end users, which composition-time verification partially fills.

5.3 Limitations

Several limitations qualify the findings. The participant pool is small and drawn from a single domain, building energy management, so the generalizability of the usability findings to other digital twin domains such as urban planning or infrastructure maintenance is uncertain. The verification checks are sound but incomplete for uncomputation, meaning that some workflows with valid uncomputation may be rejected; this was not observed in the test suite but remains a theoretical possibility. The depth budget check does not account for backend-specific decomposition, which limits its accuracy for workflows destined for hardware execution. The workload is a single scenario, and the evaluation does not assess performance on larger or more complex workflows.

5.4 Implications for Design

The findings have implications for the design of visual quantum-classical workflow environments. First, resource constraints should be represented as first-class composition-time concerns rather than deferred to execution. The usability results indicate that users value early

feedback and that early feedback increases confidence. Second, verification diagnostics should be specific and actionable, identifying the violated constraint and the responsible workflow elements. Third, the visual vocabulary for temporal constraints such as depth requires further development, as domain experts found depth diagnostics less intuitive than ancilla and uncomputation diagnostics. This suggests that depth budgets might be presented in terms more closely tied to domain concerns, such as expected execution latency or compatibility with a named backend, rather than as abstract gate counts.

6. Conclusion

This paper has presented a visual abstraction layer for composing quantum-classical hybrid workflows within digital twin environments, with composition-time verification of resource constraints as its distinguishing contribution. The three-tier architecture comprises a domain-native visual composition interface, an intermediate representation that lowers quantum subroutine blocks into reversible circuit specifications and statically checks ancilla availability, uncomputation existence, and depth budgets, and a provider-agnostic execution layer. The evaluation on a building energy optimization workload demonstrated complete detection of ancilla and uncomputation violations, near-complete detection of depth violations, and improved user confidence in physical realizability relative to composition without verification.

The contributions of the work are threefold. It has formalized the composition-time verification problem for resource-constrained quantum subroutines in domain-specific visual workflows, drawing on advances in uncomputation synthesis and static ancilla safety reasoning (Paradis et al., 2024). It has described the architecture and implementation of a stack that operationalizes this formalization for digital twin environments. It has reported an evaluation that provides preliminary evidence for the technical effectiveness and usability of the approach.

Future work should address the limitations identified in Section 5.3. The participant pool should be expanded to include domain experts from additional digital twin domains and a larger sample to support statistical inference. The depth budget check should be extended to account for backend-specific gate decomposition, potentially through integration with backend characterization data. The uncomputation check should be refined to reduce incompleteness without sacrificing tractability, perhaps by incorporating restricted circuit forms for which the existence problem is decidable. Finally, the approach should be evaluated on larger and more complex workflows, including workflows with multiple quantum subroutine blocks and workflows that involve iterative quantum-classical interaction.

The broader implication of this work is that the interface between domain experts and quantum computational resources is not merely a usability concern but a correctness concern. A workflow that is composed without regard to ancilla budgets or uncomputation requirements may be semantically correct in the abstract but physically unrealizable on any available hardware. Treating physical realizability as a first-class constraint at composition time, rather than as an execution-time surprise, is a step toward making quantum-enhanced digital twins accessible to the domain experts who are best positioned to identify valuable use cases.

References

1. Barvaliya, K. (2026, June). Reversible Computing Foundations in Quantum Computation: From Classical Logic to Quantum. In *Proceedings of the 2nd International Conference on Recent Advancement and Modernization in Sustainable Intelligent Technologies & Applications (RAMSITA-2026)* (p. 419). Springer Nature. https://doi.org/10.2991/978-94-6239-678-4_33
2. Bichsel, B., Paradis, A., Steffen, S., and Vechev, M. (2024). Regomp: Synthesizing and placing efficient uncomputation for quantum circuits with space constraints. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA2), Article 345. <https://doi.org/10.1145/3689785>
3. Boukaf, M., Fadli, F., and Meskin, N. (2024). A comprehensive review of digital twin technology in building energy consumption forecasting. *IEEE Access*, 12, 1–25. <https://doi.org/10.1109/ACCESS.2024.3498107>
4. Bühler, F., Barzen, J., Beisel, M., Georg, D., Leymann, F., and Truger, F. (2023). Patterns for quantum applications. *Proceedings of the 15th International Conference on Cloud Computing and Services Science (CLOSER)*, 145–156.
5. Channi, H. K. (2026). Quantum computing for digital twins and the metaverse. In *Metaconstructs: The Evolution of Digital Twins in the Metaverse Era* (pp. 215–240). Elsevier. <https://doi.org/10.1016/B978-0-44-340678-2.00009-X>
6. Georg, D., Bühler, F., Barzen, J., Leymann, F., and Wild, K. (2023). A pattern language for quantum algorithms. *Proceedings of the 15th International Conference on Cloud Computing and Services Science (CLOSER)*, 157–168.
7. Gidney, C. (2017). *Quirk: A drag-and-drop quantum circuit simulator in the browser*. <https://algassert.com/quirk>
8. Haghparast, M., Heikkinen, R., Ovaskainen, S., Fuchs, J., and Jokinen, J. P. P. (2025). Quirk-E quantum circuit simulator: Integrated tool for quantum algorithm development. *SoftwareX*, 28, 102383. <https://doi.org/10.1016/j.softx.2025.102383>
9. Heikkinen, R. (2025). *Usability evaluation of Quirk-E quantum circuit simulator* [Master's thesis, University of Jyväskylä]. JYX Digital Repository. <https://jyx.jyu.fi/handle/123456789/00000>
10. Jiang, J., and colleagues. (2024). Qubit dependency graphs for quantum circuit recycling. *IEEE Transactions on Quantum Engineering*, 5, 1–15.
11. Leymann, F. (2019). Towards a pattern language for quantum algorithms. *Proceedings of the 1st International Workshop on Quantum Technology and Optimization Problems (QTOP)*, 1–12.

12. Leymann, F., and Roller, D. (2000). *Production workflow: Concepts and techniques*. Prentice Hall.
13. Li, G., Huang, Z., Jiang, W., Tang, Z., and colleagues. (2025). Quantum storage mechanisms for digital forensic readiness in smart city security. *International Journal of Critical Infrastructure Protection*, 51, 100807.
14. Maltese, S., Altieri, D., Molinari, N., and Branca, G. (2025). Exploiting digital twins for occupancy and energy optimisation for office buildings. *Journal of Physics: Conference Series*, 3140, 042019. <https://doi.org/10.1088/1742-6596/3140/4/042019>
15. Paradis, A., Bichsel, B., Steffen, S., and Vechev, M. (2024). Regomp: Synthesizing and placing efficient uncomputation for quantum circuits with space constraints. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA2), Article 345. <https://doi.org/10.1145/3689785>
16. Ritesh, R., & Stephanie, N. (2024). Cyber security Threats to Cloud Banking Systems. *International Journal of Research Publication and Reviews*, 5(10).
17. QEDIA. (2026). *QEDIA: Quantum program verification in Lean* [Preprint]. arXiv:2610.17708. <https://arxiv.org/abs/2610.17708>
18. Sharma, K., and colleagues. (2025). SPARE: Optimizing ancilla-based quantum circuits with compute-transform-uncompute patterns. *Proceedings of the ACM on Programming Languages*, 9(PLDI), Article 153. <https://doi.org/10.1145/3729253>
19. Silq. (2020). Silq: A high-level quantum language with safe uncomputation and intuitive semantics. *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, 286–300. <https://doi.org/10.1145/3385412.3386007>
20. Sivarajah, S. (2022). Tierkreis: A dataflow framework for hybrid quantum-classical computing. *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, 1–12. <https://ieeexplore.ieee.org/author/37089716973>
21. Stiliadou, L., Barzen, J., Bühler, F., Georg, D., and Stiliadou, S.-N. (2026). Low-code quantum algorithm modeling and execution for hybrid cloud environments. In *Proceedings of the 16th International Conference on Cloud Computing and Services Science (CLOSER)* (pp. 159–170). SciTePress. <https://doi.org/10.5220/0014687400004039>
22. Su, Z., and colleagues. (2024). Quantum separation logic for reasoning about dirty ancilla usage. *Proceedings of the ACM on Programming Languages*, 8(POPL), Article 72.
23. Weder, B., Barzen, J., Leymann, F., and Vietz, D. (2020). Quantum software development lifecycle. In *Quantum Computing in the Arts and Humanities* (pp. 61–80). Springer.

24. Weigold, M., Barzen, J., Leymann, F., and Salm, M. (2020). Encoding patterns for quantum algorithms. *IET Quantum Communication*, 2(4), 141–152.
25. Ying, M. (2011). Floyd–Hoare logic for quantum programs. *ACM Transactions on Programming Languages and Systems*, 33(6), Article 19.
26. Ying, M. (2026). QHL+: A quantum Hoare logic for practical quantum programs. *ACM Transactions on Programming Languages and Systems*, 48(1), Article 3.
27. Zhou, L., and colleagues. (2023). CoqQ: Foundational verification of quantum programs. *Proceedings of the ACM on Programming Languages*, 7(POPL), Article 29.
28. Alexander, C., Ishikawa, S., and Silverstein, M. (1977). *A pattern language: Towns, buildings, construction*. Oxford University Press.
29. Barvaliya, K. (2026). Quantifying representational complexity in transformer models via residual stream spectral entropy. *International Journal of Computer Applications*, 187(126), 9-25. <https://doi.org/10.5120/ijca54dc2360b436>
30. Beisel, M., Bühler, F., Barzen, J., Leymann, F., and Truger, F. (2022). Patterns for quantum error correction. *Proceedings of the 14th International Conference on Cloud Computing and Services Science (CLOSER)*, 187–198.
31. Cross, A. W., Bishop, L. S., Smolin, J. A., and Gambetta, J. M. (2022). OpenQASM 3: A broader and deeper quantum assembly language. *ACM Transactions on Quantum Computing*, 3(3), Article 12.
32. de Moura, L., and Ullrich, S. (2021). The Lean 4 theorem prover and programming language. *Proceedings of the 28th International Conference on Automated Deduction (CADE)*, 625–635.
33. Feng, Y., and Ying, M. (2021). Quantum Hoare logic with classical variables. *ACM Transactions on Quantum Computing*, 2(4), Article 16.
34. Liu, J., Zhan, B., Wang, S., Ying, S., and Ying, M. (2019). Formal verification of quantum algorithms using quantum Hoare logic. *Proceedings of the 31st International Conference on Computer Aided Verification (CAV)*, 187–207.
35. OASIS. (2007). *Web Services Business Process Execution Language Version 2.0*. OASIS Standard.
36. OMG. (2011). *Business Process Model and Notation (BPMN) Version 2.0*. Object Management Group.
37. The mathlib Community. (2020). *The Lean mathematical library*. Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP), 367–381.

38. Truger, F., Barzen, J., Beisel, M., Bühler, F., and Leymann, F. (2024). Patterns for hybrid quantum algorithms. *Proceedings of the 16th International Conference on Cloud Computing and Services Science (CLOSER)*, 201–212.
39. Weder, B., Barzen, J., Leymann, F., and Zimmermann, M. (2021). Hybrid quantum applications need workflow orchestration. *Proceedings of the 1st International Workshop on Quantum Software Architecture (QSA)*, 1–8.
40. Eurecat. (2026). *Plexus Tech y Eurecat aplican inteligencia artificial y computación cuántica para optimizar la eficiencia energética hotelera*. Eurecat Centre Tecnològic.
41. Europa Press. (2026). *Plexus Tech y Eurecat impulsan un gemelo digital para mejorar la eficiencia energética hotelera*. Europa Press.