

Gestión de Recursos Cuánticos en la Era NISQ: Implicaciones y Perspectivas desde la Ingeniería de Software

Marcos Guillermo Lammers¹

Email: marcos.lammers@lifa.info.unlp.edu.ar

ORCID: <https://orcid.org/0009-0007-3954-2252>

, Federico Hernán Holik²

Email: holik@fisica.unlp.edu.ar

ORCID: <https://orcid.org/0000-0002-6776-5281>

, and Alejandro Fernández¹

Email: alejandro.fernandez@lifa.info.unlp.edu.ar

ORCID: <https://orcid.org/0000-0002-7968-6871>

¹ Universidad Nacional de La Plata, Facultad de Informática, Laboratorio de Investigación y Formación en Informática Avanzada (LIFIA), Argentina

² Universidad Nacional de La Plata, Instituto de Física (IFLP-CCT-CONICET), Argentina

Resumen Las computadoras cuánticas suponen un avance tecnológico radical en la forma de procesar la información al utilizar los principios de la mecánica cuántica para resolver problemas muy complejos que exceden las capacidades de los sistemas clásicos. No obstante, en la actual era NISQ (dispositivos cuánticos de escala intermedia ruidosa) el hardware disponible presenta diversas limitaciones, como un número acotado de qubits, elevadas tasas de error y tiempos de coherencia reducidos. La gestión eficiente de los recursos cuánticos —tanto físicos como lógicos— cobra especial relevancia en el diseño y despliegue de algoritmos cuánticos. En este trabajo analizamos el rol de los recursos en los distintos usos actuales de los dispositivos NISQ, identificando su relevancia e implicaciones para la ingeniería de software cuántico.

Palabras clave: Computación Cuántica, Gestión de Recursos Cuánticos, Ingeniería de Software Cuántico, Corrección de Errores, Estimación de Recursos Cuánticos (QRE)

Quantum Resource Management in the NISQ Era: Implications and Perspectives from Software Engineering

Resumen Quantum computers represent a radical technological breakthrough in information processing by leveraging the principles of quantum mechanics to solve highly complex problems beyond the reach of classical systems. However, in the current NISQ era (noisy intermediate-scale quantum devices), the available hardware presents several limitations,

such as a limited number of qubits, high error rates, and short coherence times. Efficient management of quantum resources —both physical and logical— is especially relevant in the design and deployment of quantum algorithms. In this paper, we analyze the role of resources in current uses of NISQ devices, identifying their relevance and implications for quantum software engineering. With this contribution, we aim to strengthen the field of Quantum Resource Estimation (QRE) and move toward scalable and reliable quantum software development.

Keywords: Quantum Computing, Quantum Resource Management, Quantum Software Engineering, Error Correction, Quantum Resource Estimation (QRE)

1. Introducción

La computación cuántica es un campo de investigación y desarrollo interdisciplinar que promete aplicaciones relevantes a futuro en áreas tales como la ciberseguridad, la resolución de problemas de optimización en la industria y las finanzas, y la simulación de moléculas para la industria química y farmacéutica.

Ya existen computadoras cuánticas, algunas de las cuales están disponibles comercialmente. En este trabajo, vamos a distinguir entre el concepto de *computadora cuántica universal y tolerante a errores* (CCUTE), y los dispositivos disponibles actualmente, a los cuales nos referiremos como dispositivos ruidosos o imperfectos. En la actualidad, no existen CCUTEs lo suficientemente robustas como para realizar aplicaciones comerciales relevantes tales como aplicar el algoritmo de Shor (Shor, 1994) para quebrar una clave criptográfica. Los expertos se refieren a los dispositivos disponibles en la actualidad con el acrónimo "NISQ", en referencia al concepto de Noisy Intermediate Scale Quantum (dispositivos cuánticos ruidosos de escala intermedia). Si bien los avances de los últimos años han sido importantes (Arute et al., 2019; Gao et al., 2024; IBM, 2024; IQM, 2023), la mayoría de los expertos coincide en que estamos a varios años de la era de las CCUTEs y de las aplicaciones comerciales relevantes. Algunos autores estiman que podríamos estar a décadas de la era de las CCUTEs capaces de resolver problemas comercialmente relevantes.

La era NISQ es un término acuñado por John Preskill (Preskill, 2018). Allí, el autor se refiere a computadoras de entre 50 y 100 qubits. Las describe como computadoras limitadas por el ruido a circuitos cuánticos poco confiables y reducidos. Pueden ser útiles en experimentos e investigaciones relacionadas con la física cuántica. Actualmente existen computadoras accesibles de 156 qubits y otras prometidas de más de 1000 (IBM, 2024). Sin embargo, el ruido sigue siendo un problema difícil de afrontar. Algunas computadoras actualmente disponibles basadas en trampas de iones tienen qubits menos ruidosos y compuertas de mayor fidelidad, pero tienen del orden de 50 qubits físicos. Sumado a esto, la dispersión del hardware existente, la falta de estándares, el elevado costo de los desarrollos y las dificultades técnicas en la escalabilidad, hacen que no sea viable la aplicación de algoritmos y circuitos cuánticos de interés comercial. Existen muchas promesas para computadoras tolerantes a fallos en el futuro (Aasen

et al., 2025; IBM, 2024; IQM, 2023). Muchos estados y empresas están trabajando en su desarrollo. Hasta ahora, a pesar de los avances, las computadoras tolerantes a fallos con un número razonablemente grande de qubits no se han hecho realidad. Sin embargo, los prototipos actualmente disponibles encuentran un uso creciente destinado a fines de investigación y desarrollo. Una ingeniería de software que tenga presente sus limitaciones nos permitirá facilitar su uso durante la era NISQ, así como estar mejor preparados para posibles actualizaciones tecnológicas en los próximos años.

Si bien no se observa en la actualidad un uso comercial extendido de los dispositivos NISQ, es importante remarcar que en los últimos años se ha desarrollado progresivamente una comunidad que involucra a distintos tipos de usuarios. Estos se concentran en equipos de investigación de instituciones públicas y empresas. Estas investigaciones son de diversa índole y pueden clasificarse (al menos) en tres grupos. Un primer grupo se enfoca en mejorar los dispositivos existentes, detectando y corrigiendo sus errores e identificando qué pasos hay que seguir en su desarrollo a futuro. Un segundo grupo se refiere al posicionamiento temprano de empresas, estados y grupos académicos respecto de la resolución de problemas a futuro (un ejemplo de esto es el desarrollo de equipos de investigación de empresas farmacéuticas que buscan aplicaciones en el mediano plazo). Finalmente, encontramos un tercer grupo que se concentra en estudiar si es posible obtener algún tipo de ventaja, ya sea comercial o de investigación pura, en el uso de dispositivos NISQ. Naturalmente, la separación entre estos tres grupos es difusa y existe un gran solapamiento entre las distintas tareas. Estos proyectos son evidencia de la existencia de una comunidad significativa de usuarios de dispositivos NISQ para los cuales el desarrollo de herramientas de software para alcanzar sus objetivos de investigación y desarrollo se ha vuelto un desafío cotidiano. Esto implica la necesidad de invertir una cantidad considerable de recursos en el desarrollo de dichas herramientas. Atendiendo a los avances de los últimos años, es importante tener en cuenta que este estado de cosas podría extenderse por varios años e incluso por décadas.

En ese contexto, podemos formular de forma precisa la pregunta que abordaremos en este trabajo: ¿Cuáles son las implicaciones para la ingeniería del software cuántico del hecho de que los usuarios de computadoras cuánticas tengan que lidiar con dispositivos NISQ, los cuales poseen recursos limitados y están sujetos a errores? La relevancia de esta pregunta está vinculada a otro hecho: la mayoría de las investigaciones en el campo conocido como *Ingeniería de Software Cuántico* se enfocan en CCUTEs. Es decir, se concentran en el estudio de dispositivos por venir sin tener en cuenta los desafíos que plantean las investigaciones que llevan a cabo los usuarios de dispositivos NISQ. La propuesta de este trabajo es identificar y sistematizar algunas de las prácticas de desarrollo que son relevantes para la ingeniería del software orientado al uso de dispositivos NISQ.

Este artículo se organiza de la siguiente manera: En la sección 2 se discute el rol de los recursos en la ingeniería del software. La sección 3 se adentra en el mundo de la computación cuántica, poniendo el eje en el rol de los recursos

en los dispositivos NISQ, y los problemas vinculados a la caracterización de su performance y su calibración. La sección 4 hace un breve repaso de los esfuerzos disponibles en trabajos publicados orientados a la caracterización y estimación de los recursos disponibles en procesadores cuánticos. Como resultado de ese análisis, se observa que la mayoría de los desarrollos están enfocados en CCUTEs. La sección 5 hace un análisis del rol de los recursos en los distintos usos que se hacen de los dispositivos NISQ en la actualidad, identificando su relevancia e implicancias para la ingeniería del software orientada al uso de computadoras cuánticas. Finalmente, la sección 6 expone conclusiones y desafíos a abordar en futuros trabajos.

2. El rol de los recursos en la ingeniería de software

La ingeniería de software, como disciplina, tiene entre sus objetivos principales la gestión eficiente de los recursos disponibles, reconociendo que estos son finitos y de ellos depende el éxito de los proyectos. Se ocupa de todas las fases del desarrollo de software, desde su diseño hasta su mantenimiento.

Para la ejecución de cualquier software es necesario tener en cuenta distintos tipos de recursos, tales como el tiempo de procesamiento, la memoria, archivos y dispositivos de entrada/salida. Estos recursos pueden ser requeridos por el software al instante de su creación o durante su ejecución. Esto significa que, desde los inicios, los ingenieros deben organizar, entre otras cosas, la utilización de los recursos de hardware y software para prevenir atascos y asegurar un desempeño óptimo. En la ingeniería de software tradicional, el desafío consiste en crear técnicas para generar software confiable que posea la suficiente adaptabilidad para gestionar la heterogeneidad, lo que incluye la capacidad para adaptarse a las limitaciones del hardware existente.

Los recursos de hardware influyen directamente en decisiones como la selección de arquitecturas de software y la optimización del rendimiento en cualquier tipo de dispositivo. Esto no es diferente en el uso de computadoras cuánticas. El hardware debe utilizarse de manera eficiente para cumplir con los requisitos de rendimiento y confiabilidad. El desafío principal es encontrar el punto de equilibrio entre los recursos escasos y los objetivos del proyecto, asegurando que el software sea eficiente, escalable y capaz de operar dentro de las limitaciones físicas disponibles. Para verificar esto, necesitamos conocer de manera confiable el hardware disponible y las necesidades del software.

3. Recursos de las computadoras cuánticas

A grandes rasgos podemos clasificar a los recursos de una computadora cuántica de acuerdo a sus niveles *físicos* y *lógicos*. Por un lado, tenemos los recursos físicos. Ejemplos de esto son el número de qubits físicos, su tasa de error y el tiempo de coherencia, el nivel de entrelazamiento que sea posible alcanzar, la conectividad de los qubits, y la fidelidad de las compuertas nativas. Por otro lado, están los recursos lógicos, como por ejemplo, el número de qubits lógicos, el conjunto de compuertas disponibles en la plataforma de acceso al dispositivo, la máxima profundidad del circuito cuántico soportada, los algoritmos

cuánticos pre-implementados, los protocolos de corrección de errores, limitaciones en el número y naturaleza de mediciones, y las herramientas de estimación de errores, entre otros. Estos recursos tienen que ser tenidos en cuenta en el modelo específico de la computación cuántica. Esa diferencia respecto a la computación clásica hace necesario un enfoque especial de ingeniería de software.

En computación cuántica, un gran número de qubits físicos suele ser considerado como una ventaja en lo que respecta a tener una mayor capacidad de procesamiento de datos. En teoría, una mayor cantidad de qubits físicos permitiría lograr más qubits lógicos en los cuales implementar protocolos de corrección de errores cuánticos. Es importante destacar que, aunque un gran número de qubits físicos es en principio deseable, también hay que tener en cuenta su calidad. Los qubits físicos con alta tasa de error no serán útiles, incluso aunque fueran numerosos. Este hecho refuerza la importancia de la calidad de los qubits como un recurso clave (Preskill, 2018). La tasa de error está íntimamente ligada al tipo de computadora cuántica (QC, por sus siglas en inglés) utilizada. Para lograr ejecutar algoritmos, obtener resultados confiables y que el sistema sea escalable, es necesario alcanzar una baja tasa de error.

El tiempo de coherencia nos dice cuánto tiempo un qubit puede mantener su estado cuántico antes de que la decoherencia lo degrade y se pierda la información. Para que un algoritmo cuántico se ejecute correctamente, el tiempo de coherencia tiene que ser lo suficientemente largo como para permitir que se realicen todas las operaciones del circuito cuántico. Si es demasiado corto, los qubits pueden perder las propiedades de superposición y entrelazamiento, y los resultados del algoritmo van a ser incorrectos y poco confiables. Muchos algoritmos, como el de Shor (Shor, 1994) o el de Grover (Grover, 1996), dependen de dichas propiedades para lograr su ventaja sobre los métodos clásicos. Un nivel de entrelazamiento débil implica que el algoritmo no tenga ninguna ventaja cuántica u obtenga resultados incorrectos.

Las compuertas nativas son operaciones básicas del hardware cuántico. Estas se implementan directamente en la arquitectura física del procesador cuántico y son específicas del tipo de tecnología utilizada (por ejemplo, qubits superconductores o iones atrapados). Son los "bloques de construcción" básicos a partir de los cuales se construyen los algoritmos cuánticos más complejos (Krantz et al., 2019).

Las compuertas de Clifford (Hadamard, Pauli-X, Pauli-Z y CNOT) son operaciones cuánticas básicas que permiten manipular y entrelazar qubits. Se pueden simular eficientemente en computadoras clásicas (Nielsen y Chuang, 2010, p. 464) lo que implica que un dispositivo que solo utiliza compuertas Clifford y mide y prepara estados en la base computacional, no explota ninguna ventaja cuántica (Aaronson y Gottesman, 2004). Si al grupo de Clifford se le agrega la compuerta T, se obtiene el conjunto Clifford+T el cual tiene la propiedad de ser universal. Un dispositivo cuyas compuertas nativas forman un conjunto universal tiene la capacidad de realizar, en principio, cualquier cálculo cuántico. Obviamente, las imperfecciones en los dispositivos NISQ imponen severas restricciones a la ventaja cuántica. Esto implica que en ingeniería de software,

la implementación de estas compuertas debe considerarse cuidadosamente para lograr una optimización de recursos (Bravyi y Kitaev, 2005).

La conectividad de los qubits es su capacidad, determinada por la topología del hardware, para establecer interacciones entrelazantes entre sí. Decimos que dos qubits están conectados si es posible aplicar una compuerta nativa entrelazante entre ellos. En sistemas con conectividad limitada, los qubits solo pueden interactuar con sus vecinos más cercanos, lo que requiere operaciones extras (como intercambios de estado con compuertas Swap) para entrelazar qubits no adyacentes. Esto aumenta la complejidad del circuito y el tiempo de ejecución de los algoritmos y, por lo tanto, la tasa de error global. Algunas arquitecturas tienen conectividad total. Esto implica que es posible aplicar una compuerta entrelazante directamente entre cualquier par de qubits elegidos, como es el caso de los procesadores basados en trampas de iones (IonQ, AQT). Por otro lado, los dispositivos basados en qubits superconductores (IBM, Google, IQM) suelen tener conectividad limitada.

El nivel de ruido consiste en fluctuaciones aleatorias y no controladas en los parámetros físicos que interactúan con los qubits. Estas fluctuaciones pueden originarse a partir de diversas fuentes, tales como el ruido térmico (variaciones de voltaje y corriente), fluctuaciones de amplitud o fase en los osciladores que generan pulsos de control, o campos eléctricos y magnéticos fluctuantes en el entorno local del qubit, como en superficies metálicas o interfaces sustrato-metal. Estas perturbaciones generan cambios no deseados en los parámetros del qubit, lo que provoca decoherencia y reduce la fidelidad de las operaciones cuánticas. El ruido puede clasificarse en dos tipos principales: ruido sistemático, que consiste en errores predecibles y repetibles, como una rotación incorrecta de un pulso de microondas debido a una calibración imperfecta, y ruido estocástico, que consiste en fluctuaciones aleatorias e impredecibles, como las causadas por el ruido térmico o la interferencia electromagnética (Krantz et al., 2019).

4. Gestión de recursos cuánticos hoy

Los enfoques actuales al estudio y gestión de los recursos en computadores cuánticos pueden clasificarse entre los que hacen estimación de recursos (Quantum Resource Estimation) y los que hacen evaluaciones comparativas (Benchmarking). Los primeros apuntan a determinar las necesidades de recursos de un algoritmo o sistema cuántico determinado. Los segundos buscan establecer bases de evaluación objetivas que permitan comparar tecnologías cuánticas.

4.1. Estimadores de Recursos Cuánticos

Actualmente, varios proyectos están en desarrollo dentro del área de investigación conocida como QRE (Estimador de Recursos Cuánticos). Estos proyectos se centran en estimar y optimizar los recursos necesarios principalmente para computadoras cuánticas universales tolerantes a fallos.

El estimador de recursos de Microsoft (Azure Quantum Resource Estimator) calcula el tiempo de ejecución y el número total de qubits basándose en una estimación de costo de la cantidad de compuertas necesarias para los circuitos cuánticos (van Dam et al., 2023).

Google desarrolló Qualtran, que está en etapa beta. Qualtran simula y prueba algoritmos utilizando abstracciones y estructuras de datos para generar automáticamente diagramas con información y tabular los requisitos de recursos. Ofrecen una biblioteca estándar de bloques de construcción para compilaciones que minimizan costos (Harrigan et al., 2024).

Zapata AI desarrolló BenchQ como parte del programa de DARPA Quantum Benchmarking Program (DARPA, 2023). BenchQ ofrece herramientas para estimar los recursos de hardware necesarios en la computación cuántica resistente a errores. Cuenta con un compilador de estados gráficos, modelos de fábricas de destilación, evaluaciones de rendimiento de decodificadores, un diseño de arquitectura para trampas de iones, implementaciones de algoritmos cuánticos específicos, entre otras funcionalidades. Además, los desarrolladores aclaran que hacen uso de la misma herramienta diseñada por Microsoft Azure Quantum Resource Estimation (QRE) (ZapataAI, 2025).

Existen enfoques como el de García-Alonso (García-Alonso et al., 2022) donde proponen un modelo de Quantum Software as a Service (QSaaS) mediante una API Gateway, demostrando la necesidad de abstraer la complejidad del hardware cuántico. Mientras que herramientas como Azure Quantum Resource Estimator se centran en métricas estáticas, otros enfoques como (García-Alonso et al., 2022) adoptan un paradigma de servicios para simplificar el acceso a recursos cuánticos que incluye un backend empresarial que gestiona telemetría y ejecuciones en múltiples proveedores comerciales (IBM, D-Wave, etc.).

M. Suchara y colaboradores de la Universidad de California, Santa Bárbara, en 2013, desarrollaron una herramienta llamada QuRE. La describen como una aplicación de estimación que calcula el costo de implementaciones prácticas de circuitos cuánticos en una variedad de tecnologías cuánticas físicas, exclusivamente para la codificación en computadoras tolerantes a fallos (Suchara et al., 2013).

Munich Quantum Toolkit (MQT) ofrece un conjunto de benchmarks para evaluar herramientas de software y automatización de diseño en computación cuántica llamado MQT Bench. Utilizan circuitos cuánticos pre-diseñados en cuatro niveles de abstracción, desde el nivel algorítmico hasta el nivel dependiente del hardware. Tienen análisis previos sobre cómo es conveniente la ejecución de diferentes computadoras de IBM, Rigetti, IonQ, OQC (Oxford Quantum Circuits) y Quantinuum, teniendo en cuenta sus compuertas nativas. Dan soporte para acceder mediante una interfaz web y un paquete de Python. Hacen uso del compilador de Qiskit y/o TKET y sus optimizadores por defecto. Tienen cinco conjuntos de compuertas nativas y siete dispositivos cuánticos con capacidades que van de 8 a 127 cúbits. Su objetivo es mejorar la comparabilidad, reproducibilidad y transparencia en la evaluación de herramientas de software cuántico (Quetschlich et al., 2023).

Otro trabajo, realizado por Saadatmand (Saadatmand et al., 2024), titulado "Fault-tolerant resource estimation using graph-state compilation on a modular superconducting architecture", se enfoca en la estimación de recursos necesarios para la computación cuántica tolerante a fallos (FTQC). Los autores emplean

una metodología basada en la compilación de estados gráficos (graph-state compilation) para evaluar los requisitos de recursos en sistemas FTQC. Analizan en arquitectura superconductora cómo la conectividad entre módulos, la latencia y otros factores impactan en los recursos totales necesarios para implementar algoritmos cuánticos.

En un trabajo realizado para DARPA y la NASA, Mozgunov, Marshall y Anand (Mozgunov et al., 2024) describen dos aplicaciones en las que su enfoque preferido es la simulación cuántica de sistemas abiertos. Proponen optimizaciones algorítmicas con parámetros específicos para las estimaciones de recursos aprovechando la invarianza traslacional, propiedad de un sistema o una función matemática de permanecer inalterada cuando se desplaza o traslada en el espacio que simplifica cálculos aprovechando estructuras que se repiten periódicamente, además del paralelismo en la aplicación de la compuerta T. Se implementan únicamente en FTQC. Las estimaciones de recursos para estos sistemas dependen centralmente de la huella del estado T, es decir, de los recursos necesarios para generar y mantener estados T.

4.2. Evaluación comparativa

En los proyectos descritos en la sección anterior, el enfoque principal para la estimación de recursos depende del algoritmo, su funcionalidad y propósito, o de los datos previos que tenemos sobre la computadora cuántica proporcionada por el proveedor de servicios. Para evaluar el hardware, también existen muchos proyectos de benchmarking basados en diferentes factores (Proctor et al., 2025). Esto también influye en los resultados de las pruebas, anuncios y conclusiones, tanto propias como en las de los competidores.

Cuadro 1. Comparación de herramientas para estimación y benchmarking en computación cuántica

Proyecto	Tipo	Métricas	NISQ	Open Source	Tecnología	Ref
Azure Quantum QRE	Estimación	Qubits, compuertas T, tiempo de ejecución	No	No	FTQC (super conductores)	van Dam et al., 2023
Qualtran (Google)	Estimación	Profundidad de circuitos, qubits físicos y lógicos	Parcial	No	FTQC/NISQ (genérico)	Harrigan et al., 2024
BenchQ	Bench marking	Fidelidad, conectividad, recursos para FTQC	Sí	Sí	FTQC (trampa de iones)	ZapataAI, 2025

Proyecto	Tipo	Métricas	NISQ	Open Source	Tecnología	Ref
QuRE	Estimación	Costo de circuitos en FTQC, compuertas nativas	No	No	FTQC (diversas tecnologías)	Suchara et al., 2013
MQT Bench	Bench marking	Compuertas nativas, ruido, compatibilidad	Sí	Sí	NISQ (multi-plataforma)	Quetschlich et al., 2023
FTQC graph-state compilation	Estimación	Recursos para FTQC, modularidad, conectividad	No	No	FTQC (super conductores modulares)	Saadatmand et al., 2024
DARPA NASA	Estimación	Huella de estados T, optimización algorítmica	No	No	FTQC (simulación abierta)	DARPA, 2023
Quantum Path [®]	Plataforma	Agnosticismo, integración híbrida, telemetría	Sí	No	Plataformas comerciales como IBM, D-Wave, Rigetti, AWS Braket, etc.	Garcia-Alonso et al., 2022

En 2018, el equipo de Google diseñó un método de benchmarking para la supremacía cuántica (Boixo et al., 2018, Bravyi y Kitaev, 2005). Este método se fundamenta en estudios teóricos sobre la dificultad de muestreo a partir de circuitos cuánticos aleatorios, la dureza computacional de su simulación clásica y sus implicaciones en la demostración de la supremacía cuántica, estableciendo conexiones con la complejidad computacional y las separaciones entre modelos cuánticos y clásicos. (Aaronson y Chen, 2017)

La implementación experimental del muestreo sobre circuitos cuánticos aleatorios llevó a Google a anunciar que el procesador Sycamore tardaba aproximadamente 200 segundos en muestrear un millón de veces una instancia de un circuito cuántico. Indicaban que la misma tarea para una supercomputadora clásica de última generación tardaría aproximadamente 10,000 años y que esta supremacía cuántica también anunciaba la era de las tecnologías cuánticas de escala intermedia ruidosa (NISQ). Además, afirmaron que con ese benchmarking se aseguraba una aplicación inmediata en la generación de números aleatorios certificables (Arute et al., 2019). Poco después, IBM (Pednault et al., 2019) propuso una simulación mucho más eficiente con hardware clásico para realizar la misma ejecución en 2.5 días y no en 10,000 años. Esta experiencia demuestra

que hay que ser cuidadosos en la selección de las pruebas y tests a realizar para sacar conclusiones correctas.

No todo Benchmarking es útil en todos los casos. En la referencia Proctor et al., 2025 se describen múltiples métodos disponibles, detallando cuáles son más adecuados para la era NISQ, otros para etapa fault-tolerant, de bajo nivel, alto nivel y cómo esto va a tener que ir cambiando necesariamente según el progreso tecnológico. También describen seis propiedades fundamentales para lograr un buen benchmark:

1. Deben estar bien fundamentados, justificando las métricas de rendimiento.
2. Un procedimiento tiene que ser claro y sin ambigüedades. Cualquier paso no especificado en ese procedimiento debe ser un parámetro configurable intencionadamente.
3. En su implementación no debería ser posible manipular los parámetros configurables para obtener resultados engañosos.
4. Los resultados no deberían corromperse cuando la computadora cuántica evaluada experimente errores pequeños o no previstos.
5. Tienen que ser eficientes y no utilizar una gran cantidad de recursos.
6. Por último, el benchmark debe ser independiente de la tecnología; solamente debería ser particular si lo que se va a medir es especial de esa arquitectura.

5. Desafíos en la gestión de recursos cuánticos en la era NISQ

Desde el punto de vista de la Ingeniería de Software, determinar los recursos disponibles en los prototipos de computadoras cuánticas, independientemente de la plataforma y de manera confiable para poder hacer una gestión correcta, sigue siendo un desafío por resolver.

Las técnicas de pruebas referidas en 4.2 y librerías o herramientas detalladas en 4.1, pueden ser útiles en diferentes casos. Aunque, además de que en su mayoría solo funcionan únicamente con CCUTEs, la aplicación de los mismos debe hacerse con anterioridad a la ejecución, tanto para la verificación de los requerimientos, como para la ejecución. Es una visión estática del uso de las computadoras cuánticas que no considera la particularidad de cómo funcionan ni la etapa de desarrollo tecnológico de utilidad en la que nos encontramos. En este último punto podemos hacer la salvedad con MQT Bench 4.1, que se basa en datos previamente obtenidos, pero no actualizados al momento de ejecutar el algoritmo. La conectividad de los qubits puede ser la misma si no cambia la topología del hardware, pero su nivel de entrelazamiento, su tiempo de coherencia o fidelidad pueden cambiar de un momento a otro.

Para un uso eficiente que tome en consideración los recursos utilizables y que se adapte a las necesidades del usuario, será necesaria una capa como parte del mismo software a ejecutar. En este trabajo remarcamos la necesidad de desarrollar una capa que tenga presente el flujo de trabajo del mismo algoritmo,

verificando que los recursos estén disponibles de manera dinámica, independientemente del hardware o del servicio cuántico disponible y que sean efectivos para el uso proyectado.

Existen recursos numerables e identificables de manera sencilla en una instancia previa a la ejecución, como cantidad de qubits necesarios, fidelidad, tiempo de coherencia, compuertas disponibles, topología según el hardware. (Saraiva et al., 2021). La información sobre estas características es usualmente proporcionada por el proveedor del servicio (aunque la frecuencia con la cual está disponible de forma actualizada puede variar de proveedor a proveedor).

Sin embargo, lo esperable para obtener ventaja cuántica es la capacidad de generar estados con los recursos físicos necesarios, apelando a un conjunto de compuertas nativas universales. Las capacidades que pueda llegar a tener un dispositivo dado para cumplir con ese objetivo pueden variar con el tiempo. La sensibilidad al ruido de los qubits puede perjudicar la ejecución y fiabilidad en los resultados del circuito/algoritmo. En particular, los estados logrados a partir de la utilización del conjunto de compuertas universales de cada dispositivo tienen que ser capaces de generar estados mágicos o "Non-stabilizerness", los cuales son los que permiten que los algoritmos cuánticos superen las capacidades clásicas. En un trabajo del 2025 (Macedo et al., 2025) hacen uso de las desigualdades de Bell (Brunner et al., 2014) especialmente diseñadas para que puedan actuar como testigos de estos estados mágicos. En otros trabajos de investigación previos (Arango, Holik et al., 2024; Holik et al., 2024) se analizan correlaciones cuánticas mediante diferentes medidas de entrelazamiento, comparando conjuntos de puertas universales a partir de estados generados por circuitos aleatorios cuánticos (QRC). Por otro lado, es importante remarcar la necesidad de desarrollar software de testeo que no implique exponer el algoritmo a usar. Las herramientas presentadas en (Arango, Holik et al., 2024; Holik et al., 2024) son un paso importante para cumplir con ese objetivo, dado que la caracterización de correlaciones usando circuitos cuánticos aleatorios permite tener una evaluación global de un conjunto de qubits dado.

6. Conclusiones y trabajo futuro

Concluimos que es necesaria una capa que evalúe los recursos disponibles para resolver de manera dinámica si es conveniente proceder a la ejecución, brindar una mayor confianza en los resultados obtenidos y garantizar la funcionalidad eficiente de esta tecnología. Haciendo uso de las investigaciones mencionadas en 5, es posible evaluar si los dispositivos disponen de los recursos requeridos por el usuario final con una capa dinámica (Figura 1) extendiendo mediante evaluación en tiempo real. Esta información puede ser útil a la hora de decidir entre diferentes prototipos de computadoras cuánticas.

A futuro, planificamos implementar esta propuesta como una capa de integración, la cual, en términos generales, presentaría las siguientes características:

1. Ofrecer una librería de funciones para consultar recursos disponibles, expresar requisitos y/o restricciones de código abierto.

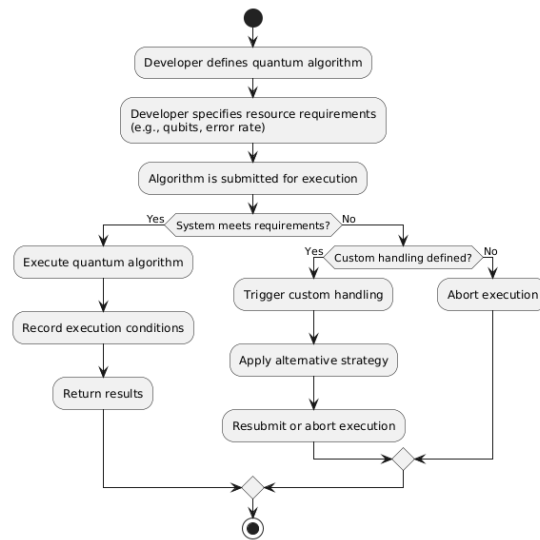


Figura 1. Diagrama de flujo de una capa de integración

2. Ser una librería abstracta, permitiendo implementaciones específicas para distintas computadoras cuánticas.
3. Ser extensible, de modo que se pueden agregar nuevas abstracciones de recursos a medida que evolucionen las plataformas.
4. Incluir un lenguaje propio para definir el manejo de las excepciones (por ejemplo, cuando no se cumple con determinado requerimiento).

Cuadro 2. Comparación entre la propuesta actual y herramientas existentes

Criterio	Propuesta	Herramientas Existentes
Enfoque Dinámico	Evaluación en tiempo real durante la ejecución (ruido, qubits disponibles, etc.). Monitorización de métricas NISQ.	Análisis estático previo. Datos históricos/fijos (ej: MQT Bench).
Soporte NISQ	Decisión adaptativa basada en estado actual del hardware (ej: tiempo de coherencia residual).	Mayoría orientada a FTQC. Soporte NISQ limitado sin gestión dinámica).
Agnosticismo	Librería abstracta adaptable a cualquier hardware. Extensible (métricas personalizables).	QuantumPath [®] agnóstico pero cerrado. No extensibles).
Integración	Middleware independiente (Fig. 1). Manejo de excepciones. Código abierto.	Soluciones monolíticas. Sin retroalimentación proactiva (benchmarks solo reportan métricas).

Referencias

- Aaronson, S., & Chen, L. (2017). Complexity-Theoretic Foundations of Quantum Supremacy Experiments. *LIPICs, Volume 79, CCC 2017*, 79, 22:1-22:67. <https://doi.org/10.4230/LIPICS.CCC.2017.22>
- Aaronson, S., & Gottesman, D. (2004). Improved simulation of stabilizer circuits. *Phys. Rev. A*, 70, 052328. <https://doi.org/10.1103/PhysRevA.70.052328>
- Aasen, D., et al. (2025, febrero). Roadmap to Fault Tolerant Quantum Computation Using Topological Qubit Arrays. <https://doi.org/10.48550/arXiv.2502.12252>
- Arango, A. C. G., Holik, F. H., et al. (2024). Geometrical Aspects Of Resources Distribution In Quantum Random Circuits. <https://arxiv.org/abs/2405.01650>
- Arute, F., et al. (2019). Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779), 505-510. <https://doi.org/10.1038/s41586-019-1666-5>
- Boixo, S., et al. (2018). Characterizing quantum supremacy in near-term devices. *Nature Physics*, 14(6), 595-600. <https://doi.org/10.1038/s41567-018-0124-x>
- Bravyi, S., & Kitaev, A. (2005). Universal quantum computation with ideal Clifford gates and noisy ancillas. *Physical Review A*, 71(2), 022316. <https://doi.org/10.1103/PhysRevA.71.022316>
- Brunner, N., et al. (2014). Bell Nonlocality. *Reviews of Modern Physics*, 86(2), 419-478. <https://doi.org/10.1103/RevModPhys.86.419>
- DARPA. (2023). Quantum Benchmarking [Accedido: 3 de marzo de 2025]. <https://www.darpa.mil/research/programs/quantum-benchmarking>
- Gao, D., et al. (2024, diciembre). Establishing a New Benchmark in Quantum Computational Advantage with 105-Qubit Zuchongzhi 3.0 Processor. <https://doi.org/10.48550/arXiv.2412.11924>
- Garcia-Alonso, J., et al. (2022). Quantum Software as a Service Through a Quantum API Gateway. *IEEE Internet Computing*, 26(1), 34-41. <https://doi.org/10.1109/MIC.2021.3132688>
- Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*, 212-219. <https://doi.org/10.1145/237814.237866>
- Harrigan, M. P., et al. (2024, septiembre). Expressing and Analyzing Quantum Algorithms with Qualtran. <https://doi.org/10.48550/arXiv.2409.04643>
- Holik, F., et al. (2024). Group-invariant estimation of symmetric states generated by noisy quantum computers. <https://arxiv.org/abs/2408.09183>
- IBM. (2024). IBM Quantum Development & Innovation Roadmap Explainer 2024 Update [Accedido: 31 de marzo de 2025]. https://www.ibm.com/quantum/assets/IBM_Quantum_Development_&_Innovation_Roadmap_Explainer_2024-Update.pdf
- IQM. (2023). IQM Quantum Computers Unveils Development Roadmap Focused on Fault-Tolerant Quantum Computing by 2030 [Accessed: 2025-02-28]. <https://www.meetiqm.com/newsroom/press-releases/iqm-quantum->

- computers- unveils- development- roadmap- focused- on- fault- tolerant- quantum-computing-by-2030
- Krantz, P., et al. (2019). A quantum engineer's guide to superconducting qubits. *Applied Physics Reviews*, 6(2), 021318. <https://doi.org/10.1063/1.5089550>
- Macedo, R. A., et al. (2025, marzo). Witnessing Magic with Bell Inequalities. <https://doi.org/10.48550/arXiv.2503.18734>
- Mozgunov, E., Marshall, J., & Anand, N. (2024, junio). Applications and resource estimates for open system simulation on a quantum computer. <https://doi.org/10.48550/arXiv.2406.06281>
- Nielsen, M. A., & Chuang, I. L. (2010). *Quantum Computation and Quantum Information* (10th Anniversary Edition). Cambridge University Press.
- Pednault, E., Gunnels, J. A., Nannicini, G., Horesh, L., & Wisnieff, R. (2019, octubre). Leveraging Secondary Storage to Simulate Deep 54-Qubit Sycamore Circuits. <https://doi.org/10.48550/arXiv.1910.09534>
- Preskill, J. (2018). Quantum Computing in the NISQ era and beyond. *Quantum*, 2, 79. <https://doi.org/10.22331/q-2018-08-06-79>
- Proctor, T., Young, K., Baczewski, A. D., & Blume-Kohout, R. (2025). Benchmarking quantum computers. *Nature Reviews Physics*, 7(2), 105-118. <https://doi.org/10.1038/s42254-024-00796-z>
- Quetschlich, N., Burgholzer, L., & Wille, R. (2023, julio). MQT Bench: Benchmarking Software and Design Automation Tools for Quantum Computing. <https://doi.org/10.22331/q-2023-07-20-1062>
- Saadatmand, S. N., et al. (2024, junio). Fault-Tolerant Resource Estimation Using Graph-State Compilation on a Modular Superconducting Architecture. <https://doi.org/10.48550/arXiv.2406.06015>
- Saraiva, L., Haeusler, E. H., Costa, V., & Kalinowski, M. (2021). Non-Functional Requirements for Quantum Programs. *2nd Quantum Software Engineering and Technology Workshop (QSET), co-located with IEEE International Conference on Quantum Computing and Engineering (QCE21), Virtual Conference, Oct 19*.
- Shor, P. (1994). Algorithms for quantum computation: discrete logarithms and factoring. *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 124-134. <https://doi.org/10.1109/SFCS.1994.365700>
- Suchara, M., Kubiawicz, J., Faruque, A., Chong, F. T., Lai, C.-Y., & Paz, G. (2013). QuRE: The Quantum Resource Estimator Toolbox. *2013 IEEE 31st International Conference on Computer Design (ICCD)*, 419-426. <https://doi.org/10.1109/ICCD.2013.6657074>
- van Dam, W., Mykhailova, M., & Soeken, M. (2023). Using Azure Quantum Resource Estimator for Assessing Performance of Fault Tolerant Quantum Computation. *Proceedings of the SC '23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis*, 1414-1419. <https://doi.org/10.1145/3624062.3624211>
- ZapataAI. (2025). Bench-Q: Quantum Benchmarking Framework [Accedido: 24 de febrero de 2025]. <https://github.com/zapatacomputing/benchq>