Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Traspasando fronteras: la sinergia de la accesibilidad web, la computación cuántica y la arquitectura hexagonal

Transcending boundaries: synergy of web accessibility, quantum computing and hexagonal architecture

Vlatko Jesús Marchan Sekulic

La Laguna, 25 de mayo de 2023

Dña. **Pino Caballero Gil**, con N.I.F. 45.534.310-Z Catedrática de Universidad adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora

D. **Daniel Escanez Expósito**, con N.I.F. 79.159.491-T miembro del Grupo de Investigación en Criptografía de la Universidad de La Laguna (CryptULL), como cotutor

C E R T I F I C A (N)

Que la presente memoria titulada:

”*Traspasando fronteras: la sinergia de la accesibilidad web, la computación cuántica y la arquitectura hexagonal*”

ha sido realizada bajo su dirección por D. **Vlatko Jesús Marchan Sekulic**, con N.I.F. 79.062.011-V.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 25 de mayo de 2023

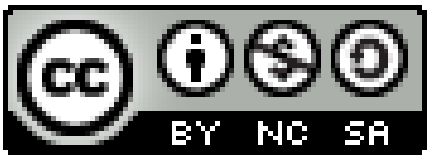
Agradecimientos

Este trabajo no hubiera sido posible sin el apoyo brindado por Pino y Daniel, por su excelente ayuda, y por estar siempre ahí para guiarme y brindarme los recursos que necesitaba para la finalización exitosa de mi proyecto final de grado.

Gracias a mi madre y a mi padre que me han apoyado en todo en el transcurso de mi carrera. Hago una especial dedicatoria a mi padre que hubiese estado muy orgulloso de mis logros Q.E.P.D.

Por último y no menos importante, hago una dedicatoria al resto de mis familiares y a todos mis amigos, por siempre estar ahí y aguantarme en todos esos momentos en los que les comía la cabeza con temas de cuántica y seguridad.

Licencia



Resumen

En el presente Trabajo Final de Grado se presenta el desarrollo llevado a cabo para la continuación de la línea de investigación en computación cuántica iniciada el año pasado con la librería de software libre QuantumSolver. Concretamente, se ha creado un entorno web accesible y auto-explicativo posibilitando la inclusión de todos los segmentos de la población a los contenidos desarrollados dentro de la librería. Asimismo, se han incluido en el nuevo módulo de la librería QuantumSolverSubrutine los algoritmos cuánticos de Simon y de Shor. El nuevo módulo se enfoca hacia algoritmos cuánticos que para su funcionamiento requieren de un post-procesado clásico para la obtención de los resultados. Por último, también se describe un diseño en arquitectura hexagonal aplicada para las necesidades futuras de la librería.

Palabras clave: Accesibilidad web, React, Desarrollo web, Qiskit, Algoritmo de Shor, Algoritmo de Simon, Transformada cuántica de Fourier, Qiskit, Computación cuántica, RSA, Computación híbrida, Arquitectura hexagonal.

Abstract

This Final Degree Project presents the development carried out for the continuation of the research line in quantum computing initiated last year with the free software library it QuantumSolver. Specifically, an accessible and self-explanatory web environment has been created to enable the inclusion of all segments of the population to the contents developed within the library; in addition, Simon's and Shor's quantum algorithms have been included in the new module of the library QuantumSolverSubrutine. The new module focuses on quantum algorithms that require classical post-processing to obtain results. Also, a hexagonal architecture design has been applied, keeping in mind the future needs of the library.

Keywords: Web Accessibility, React, Web Development, Qiskit, Shor's Algorithm, Simon's Algorithm, Quantum Fourier Transform, Qiskit, Quantum Computing, RSA, Hybrid Computing, Hexagonal Architecture.

Índice general

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

1.1. Estado del arte

En el último lustro, la computación cuántica ha adquirido un gran protagonismo como principal amenaza contra los cifrados utilizados en la actualidad (1). El grupo de cifrados que se encuentran más afectados son los cifrados de clave pública como *RSA* (2) y los basados en curvas elípticas. Dichos cifrados tienen un gran despliegue en la tecnología pues se usan, por ejemplo, para proteger las comunicaciones en Internet y en mensajería instantánea. Por tanto, el desarrollo de ordenadores cuánticos con suficientes cúbits significaría para la sociedad un gran problema porque los datos personales de todos los ciudadanos y las comunicaciones que realizan de manera habitual en Internet quedarían desprotegidos.

Uno de los principales hándicaps en la computación cuántica es la tasa de error, que se obtiene por el simple hecho de ejecutar un algoritmo. Esta tasa de error provoca que en las ejecuciones se obtengan datos erróneos y afecten a la precisión del cálculo cuántico. Por ello, la corrección de dichos errores es un requisito ineludible para la resolución de problemas como la factorización de números primos. Hace poco *Google* anunció lo que describe como su segundo hito en el camino hacia un ordenador cuántico útil (3) (4), demostrando que pueden reducir la tasa de error de los cálculos al hacer que su código cuántico sea mayor. Este hecho tiene especial relevancia debido a que cuanto mayor es el código, por lo general, se pierde rendimiento. Sin embargo, la tasa de error en los ordenadores cuánticos en la actualidad aún es considerable y necesita ser reducida significativamente con el fin de alcanzar un dispositivo de cómputo cuántico ideal.

Otro aspecto relevante del uso de las tecnologías, que últimamente ha cobrado gran importancia, es la inclusión de personas con discapacidad. De hecho, esta problemática la señala la *Organización Mundial de la Salud* (OMS) en sus últimos informes, (5) donde estima que alrededor de 1300 millones de personas, es decir, el $16\,\%$ de la población mundial padece una discapacidad importante. Además, cabe destacar la labor de concienciación que lleva a cabo la *Organización Nacional de Ciegos Españoles* (ONCE) (6) promoviendo el desarrollo de tecnologías inclusivas, diseñadas para satisfacer las necesidades de las personas con discapacidad, con el fin de conseguir su acceso y aprovechamiento de las últimas innovaciones tecnológicas. En concreto, se busca que la tecnología no sea un impedimento en la experiencia de los usuarios sino que se adapte para que se puedan utilizar sin depender de terceras personas, acabando con ello con las barreras de discriminación que tradicionalmente han perjudicado a este colectivo de la sociedad.

1.2. Objetivos

Los objetivos del presente proyecto se dividen en 3 grupos diferenciados.

En el primer grupo se engloban los objetivos planteados para la creación de un nuevo entorno web accesible, sustituyendo de esta manera el preexistente en la librería *QuantumSolver* (7). En este grupo no solo se contempla la accesibilidad del proyecto, sino que también se toma en consideración la estructura auto-explicativa de la página web. De esta manera se consigue que los usuarios menos experimentados sean capaces de entender su funcionamiento, sin necesidad de acudir a otras fuentes ni tener conocimientos previos. Además, se ha velado siempre por la estética y las funcionalidades.

En el segundo grupo se encuentra la mayor aportación científica del trabajo a la librería cuántica *QuantumSolver*, gracias a la incorporación de dos algoritmos cuánticos al nuevo módulo desarrollado llamado *QuantumSolverSubrutine*. Este nuevo módulo tiene como objetivo albergar algoritmos que para su funcionamiento requieren de una subrutina cuántica. Dichos algoritmos necesitan un post-procesamiento clásico de los resultados de la subrutinas cuántica ejecutadas. De hecho, en la comunidad científica ya se admite el potencial de las técnicas híbridas de computación cuántica-clásica por permitir la ejecución de algoritmos que combinan las mejores cualidades de los paradigmas clásicos y cuánticos de la computación.

En el tercer y último grupo de objetivos se encuentra el diseño planteado para aplicar una arquitectura hexagonal al proyecto de *QuantumSolver*. Los beneficios que se obtienen al seguir esta arquitectura en el desarrollo del código a largo plazo son las siguientes: sirve como guía para crear y desarrollar un proyecto, facilita la depuración del código en el desarrollo, asegura un entendimiento común del proyecto entre los encargados de la parte técnica, reduce el riesgo de errores, e incrementa la facilidad de uso de la aplicación y de su mantenimiento.

1.3. Fases

1.3.1. Estudio bibliográfico

El estudio bibliográfico tuvo lugar principalmente durante los primeros meses del proyecto. En este periodo de tiempo se llevó a cabo una investigación sobre las tecnologías web a implementar. Con esta finalidad se investigaron una serie herramientas de desarrollo web:

- *React* (8): biblioteca de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página.
- *Redux* (9) (10): librería de código abierto para manejar el estado de las aplicaciones.
- *MUI* (Material UI) (11): librería de componentes de React, de código abierto creada por *Google*, basada en *Material Design* (12), que proporciona una serie de pautas y lineamientos que sientan la base para crear diseños para aplicaciones web.

Tras finalizar el estudio de las herramientas web, se procedió a realizar un estudio bibliográfico sobre los conceptos básicos de la computación cuántica. Para este propósito, se recurrió sobre todo al material proporcionado por *IBM* (13) con el que se aprendió a utilizar *Qiskit*, kit de desarrollo de software para trabajar con ordenadores cuánticos a

nivel de circuitos, pulsos y algoritmos. Por otro lado, para comprender el marco teórico de la computación cuántica se recurrió principalmente al libro (14).

1.3.2. Planificación

La etapa de planificación se realizó una vez se finalizó el estudio bibliográfico. En primer lugar, se comenzó generando una primera aproximación del diseño de la página web. Para definir este diseño, se hizo una lluvia de ideas con el uso de mockups (15), con el fin de conseguir un diseño inicial de la página web. Por otro lado, se eligieron los algoritmos cuánticos a implementar en el proyecto, delimitando las etapas de investigación, implementación y resultados de los algoritmos.

1.3.3. Desarrollo

Con respecto al desarrollo del proyecto, la implementación del código desarrollado se llevó a cabo en el repositorio de *Github* (16). Dicho repositorio es un *fork* del proyecto principal, donde se recogen las aportaciones realizadas. Además, la metodología utilizada fue una metodología ágil, en la cual se realizaron *sprints* de varias semanas donde al finalizar cada apartado se llevaron a cabo reuniones con la tutora y el cotutor del Trabajo Final de Grado. En dichas reuniones se expusieron los resultados obtenidos y los problemas encontrados, y se marcaban los siguientes objetivos a perseguir.

1.4. Estructura de la memoria

El presente trabajo comienza con el capítulo de introducción. A lo largo de este capitulo se describe el estado del arte, en el que se destaca la gran importancia que ha adquirido la computación cuántica como principal amenaza contra los algoritmos de cifrado de clave pública en la actualidad. Además, se remarca cómo en el proceso de creación de nuevos entornos tecnológicos es crucial tomar en consideración a las personas que padecen algún tipo de discapacidad. También en este capítulo se describen los objetivos y las fases seguidas en la ejecución de este Trabajo Final de Grado.

El segundo capítulo se subdivide en tres apartados diferenciados. En el primero se procede a explicar las herramientas elegidas para el desarrollo del entorno web, mencionando las ventajas de utilizarlas. En el segundo se muestran las diferentes interfaces desarrolladas para la página web. Por último, se explican los resultados obtenidos en la auditoría de accesibilidad realizada al entorno web.

A lo largo del tercer capítulo se introduce el algoritmo de *Simon*, la implementación que se ha desarrollado de ese algoritmo, y los resultados obtenidos. El capítulo cuatro introduce el algoritmo de Shor, prestando especial atención a la *transformada cuántica de Fourier*, y describiendo la implementación desarrollada del algoritmo y sus resultados. En el siguiente capítulo se procede a explicar el modelo propuesto de la arquitectura hexagonal para la librería y las ventajas que supone con respecto a la estructuración actual del proyecto. Posteriormente, en el sexto y octavo capítulo se describen las conclusiones y líneas futuras a desarrollar en español y en inglés respectivamente. El último capítulo, recoge el presupuesto estimado del costo incurrido durante la ejecución del Trabajo Final de Grado. Finalmente, el apéndice incluye un congreso nacional en el que se ha aceptado un artículo basado en este trabajo, que será presentado en las próximas semanas.

Capítulo 2

Entorno web accesible

2.1. Introducción a las herramientas utilizadas

En este apartado se presenta la lista de herramientas seleccionadas para la creación del nuevo entorno web y las principales ventajas que aportan al proyecto. En primer lugar se decidió utilizar *React*, librería de desarrollo web que brinda al proyecto modularidad, eficiencia y rendimiento, además de una gran compatibilidad con otros *frameworks* y librerías. A continuación, se optó por emplear la librería *Redux* para manejar el estado de la aplicación, pues el empleo de *Redux* en el proyecto posibilita el manejo del estado de una forma centralizada, previsible y fácilmente depurable.

En cuanto al entorno de desarrollo local, se han seguido en todo momento las recomendaciones proporcionadas por la página oficial de *React* (17). Cuando se consultó por primera vez dicha documentación, se recomendaba el uso del entorno de desarrollo local "create react app" (CLI oficial de *React*, desarrollado por *Facebook*). Sin embargo, en la última actualización de la documentación presentada a mediados del mes de marzo los desarrolladores de *React* dieron al CLI creado por Facebook como obsoleto ye hicieron un comunicado para que todos los proyectos que utilizaran dicho CLI migraran a otros entornos. Por ello, en la misma semana del comunicado se decidió migrar el proyecto de forma inmediata. La herramienta de desarrollo que se decidió utilizar para llevar a cabo la migración fue *Vite* (18), que es un servidor de desarrollo local que tiene como objetivo una experiencia de desarrollo rápida y ágil en los proyectos web modernos. La migración a este nuevo entorno de desarrollo ha brindado tanto mayor rapidez en el despliegue del servidor local, como un mayor control de las variables de entorno del proyecto.

Una vez elegidas las herramientas para el funcionamiento del entorno web, se procedió a elegir la librería encargada de comunicar el *backend* y el *frontend* de la aplicación. Para ello se decidió utilizar *axios* (19), librería que tiene una serie de ventajas, entre las que destaca realizar peticiones con menos código, a diferencia de *Fetch*, que solo necesita una devolución de llamada *.then()* para acceder a los datos *JSON* solicitados y permite un mejor manejo de errores (*Axios* arroja 400 y 500 de manera automática al ocurrir, sin tener que ser manejados por el programador como ocurre en *Fetch API*).

Finalmente, se barajaron diferentes librerías de diseño y se decidió utilizar *Material UI (MUI)*. Esta es una librería de componentes de interfaz de usuario para *React* diseñada para construir aplicaciones web modernas, siguiendo los principios de *Material Design*, que utiliza una arquitectura de componentes para definir y crear los elementos de la interfaz del usuario. Cada uno de estos componentes se construye a partir de componentes de *React* y *html*. Debido a esto *MUI*, es un entorno de desarrollo que presta especial

atención a la accesibilidad de los componentes, permitiendo a los programadores añadir elementos de accesibilidad a los mismos. Además, la librería permite personalizar los estilos de los componentes y manteniendo un estilo coherente en toda la aplicación, utilizando temas centralizados.

2.2. Entorno web

La estructura de directorios utilizada para la página web sigue la distribución descrita en la Fig. 2.1. Se enfoca en conseguir la separación del código del funcionamiento de *React* y *Redux*.

Por un lado, la estructura de *React* se implementó de tal manera que se distinguieran las páginas y los componentes que conforman la *web*. Esto permite que en futuras iteraciones se reutilicen los diferentes componentes desarrollados. De esta manera, se estandariza la página y se evita el duplicado de código.

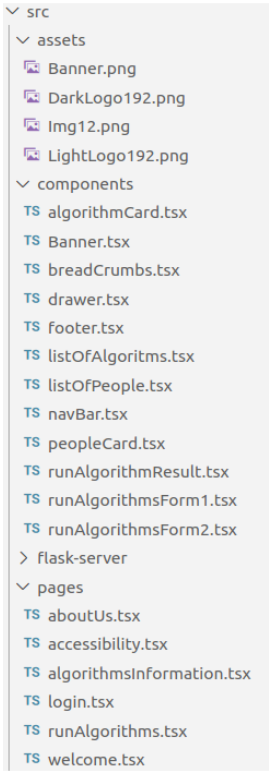
Por otro lado, la estructura de *Redux* sienta las bases del funcionamiento lógico de la *web*, realizando la separación en las diferentes partes:

- *Actions*: Describen un cambio ocurrido en la aplicación, pero no especifican cómo cambió el estado.
- *Reducers*: Se encargan de gestionar el estado, para ello toman el estado actual y una acción como argumentos, y devuelven un nuevo estado a la aplicación con la acción aplicada.
- *Services:* Realizan las peticiones al backend para el envío y recepción de la información.
- *themeFunctions*: Son funciones encargadas de la gestión del tema de *Material UI*.

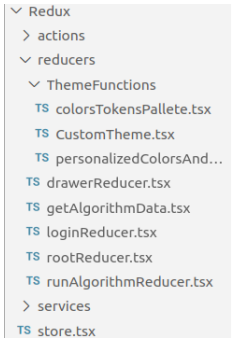
En el entorno web se ha implementado la funcionalidad para cambiar de modo oscuro y claro. Su finalidad es adaptarse a las diferentes situaciones en las que el usuario la utilice. De esta manera, si el usuario se encuentra en situaciones de alta luminosidad los modos web claros facilitan la lectura y la distinción de los elementos. Por otro lado, los modos oscuros se adaptan mejor a situaciones nocturnas y de baja luminosidad, permitiendo el cuidado de la vista y mejorando distinción de las letras de la web. Cabe mencionar que en el cambio de modalidad también se modifican los logos de la página adaptándose a un contraste suficiente para ser distinguido correctamente y sin dificultades (véase Fig. 2.2).

Además, para garantizar el cumplimiento constante de los estándares de accesibilidad, se emplearon herramientas automáticas con el fin de llevar a cabo la verificación. La principal ha sido *WAVE* (20), que es un conjunto de herramientas de evaluación que facilitan la identificación de errores de accesibilidad y el cumplimiento de las pautas *WCAG* (véase Fig. 2.3).

En las figuras Fig. 2.4, Fig. 2.4 y Fig. 2.8 se presentan las pantallas creadas en la aplicación web. Dichas pantallas se han desarrollado con un enfoque *mobile first* dando prioridad a que el diseño de la web que se adapte al entorno móvil. La importancia de este punto radica en la ergonomía que gana el entorno web adaptándose a dispositivos con pantallas de menor resolución sin perder su formato y estilo. Estas propiedades también se aplican al realizar Zoom desde la versión de escritorio, cumpliendo con ello otro punto fundamental para la accesibilidad, facilitando a las personas con problemas de vista ampliar la pantalla sin verse afectada la visualización contenido.

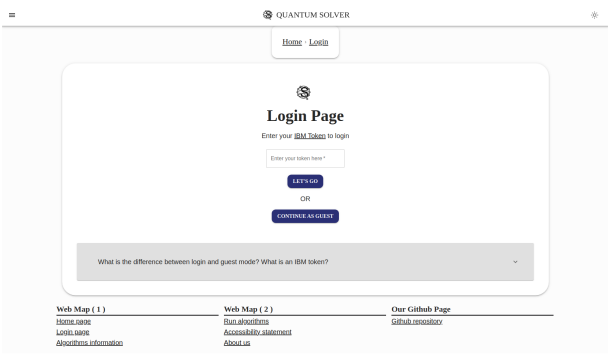


(a) Directorio *React*

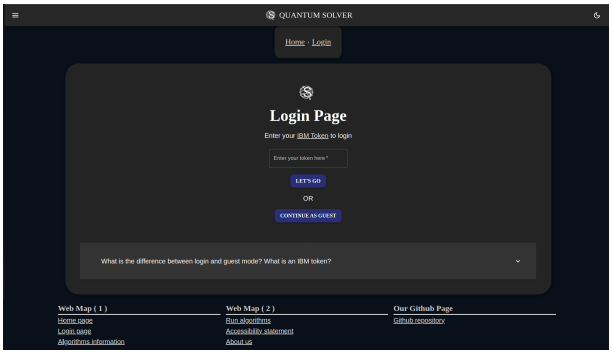


(b) Directorio *Redux*

Figura 2.1: Estructura de directorios



(a) Pantalla de Login claro



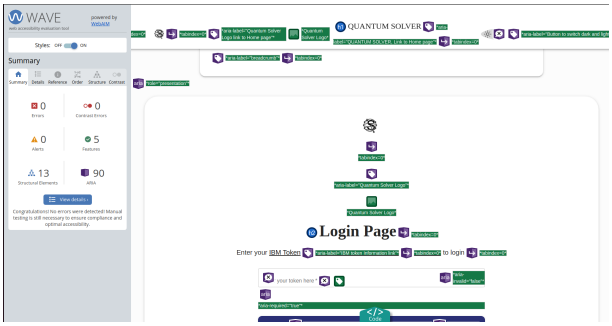
(b) Pantalla de Login Oscuro

Figura 2.2: Modos claro y oscuro de la página web

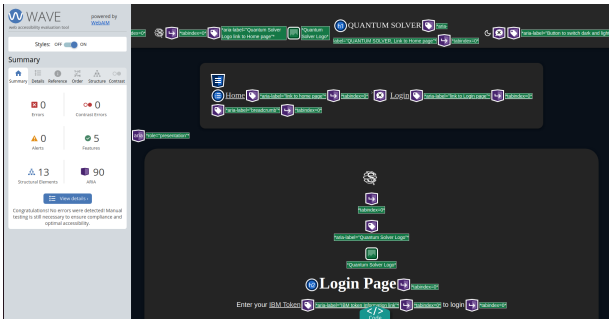
2.3. Auditoría de accesibilidad

El objetivo principal del desarrollo de la biblioteca de código abierto QuantumSolver es hacer accesible la computación cuántica al público en general. Para lograr este objetivo, se ha creado un nuevo sitio web diseñado para ser lo más accesible posible, siguiendo las normas WCAG 2.1 (21) de accesibilidad. Estas directrices son un conjunto de normas técnicas que constan de pautas agrupadas en cuatro principios: perceptible, operable, comprensible y robusto. Cada directriz incluye criterios de conformidad específicos que pueden ponerse a prueba y se clasifican en tres niveles: A, AA y AAA.

Las normas WCAG 2.1 AA representan la versión actualizada de las normas WCAG 2.0 e incorporan 17 nuevos criterios de conformidad que requieren ser considerados.

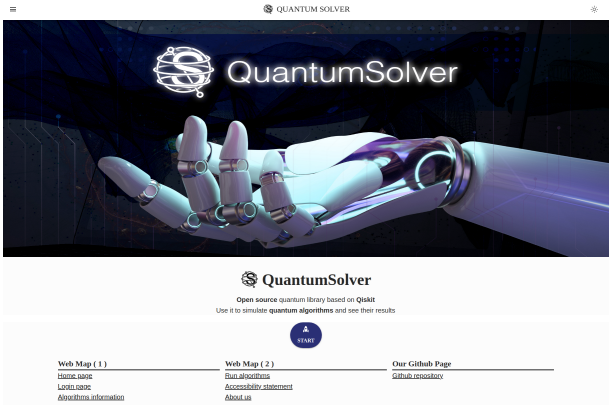


(a) Pantalla de Login claro

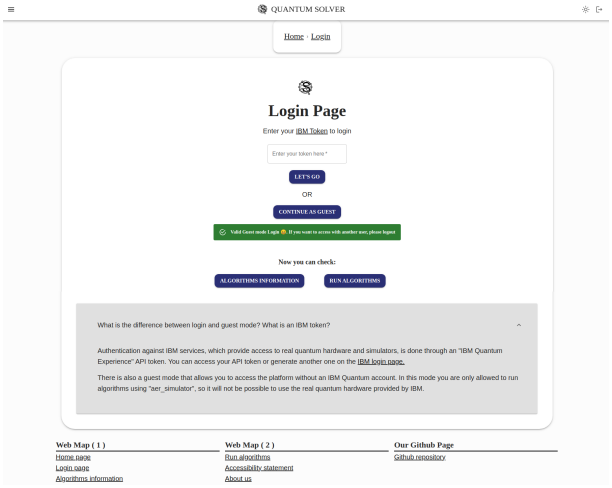


(b) Pantalla de Login Oscuro

Figura 2.3: Funcionamiento del verificador de la accesibilidad



(a) Pantalla de inicio



(b) Pantalla de Login

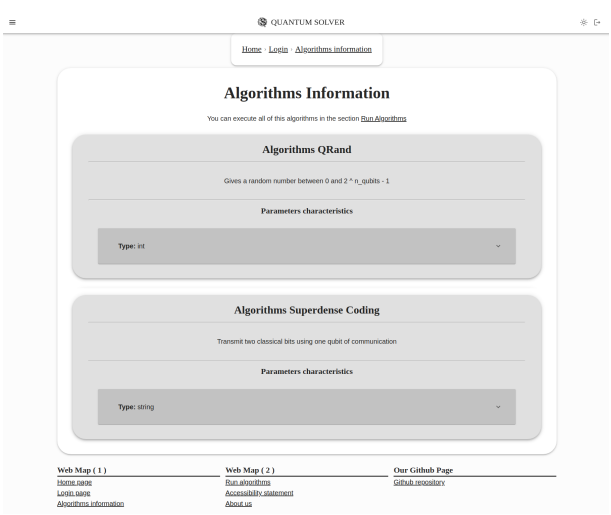
Figura 2.4: Páginas de inicio y login

1. Principio perceptible: criterios de conformidad para asegurar que la información y los componentes de la interfaz de usuario sean mostrados de manera comprensible para el usuario. Se dividen en 9 criterios de conformidad A y 11 criterios de conformidad AA.

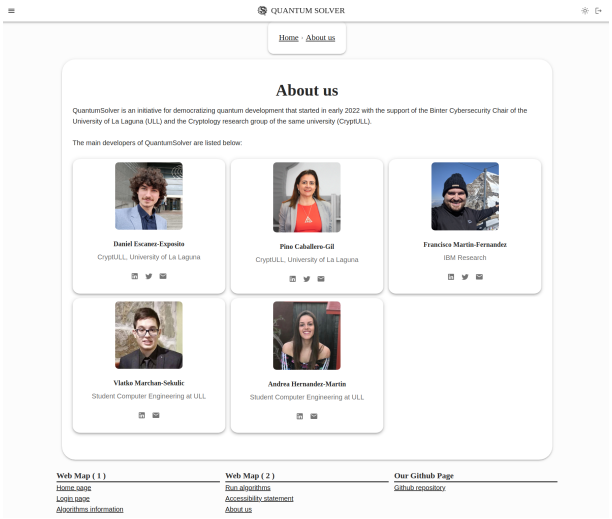
2. Principio operable: criterios de conformidad para garantizar la accesibilidad de la interfaz de usuario y la navegación en entornos web. Se siguen unas directrices específicas para lograrlo, las cuales permiten que las personas con movilidad reducida puedan utilizar la navegación por teclado. Estas directrices se basan en 14 criterios de conformidad A y 3 criterios de conformidad AA.

3. Principio comprensible: criterios de conformidad que aseguran que la información relacionada con las acciones realizadas por los usuarios sea clara y comprensible. Para lograrlo se establecen 5 criterios de conformidad A y 5 de conformidad AA.

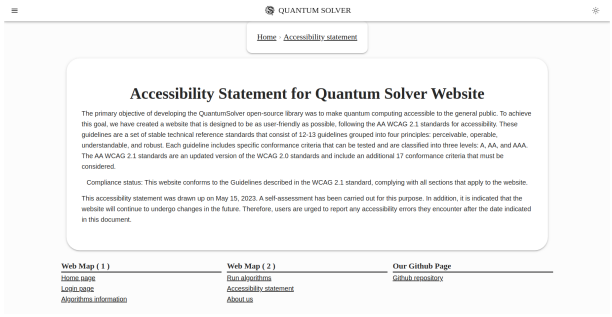
4. Robusto: criterios de conformidad que aseguran que el contenido sea lo suficientemente sólido y robusto como para ser interpretado correctamente por una amplia gama de agentes de usuario, incluyendo tecnologías de asistencia. Para lograrlo, se establecen 2 criterios de conformidad A y 1 criterio de conformidad AA. Estos criterios aseguran que el contenido sea accesible y comprensible para diversos dispositivos y tecnologías de asistencia, permitiendo una experiencia óptima para



(a) Pantalla de información de los algoritmos

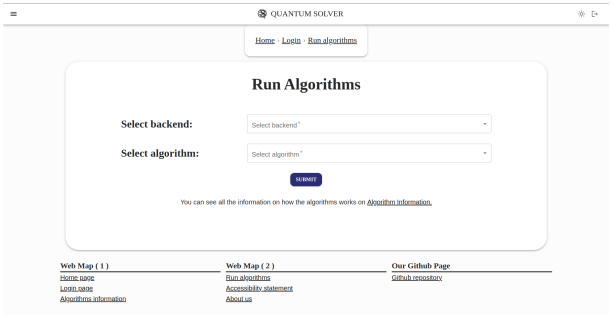


(b) Pantalla de sobre nosotros

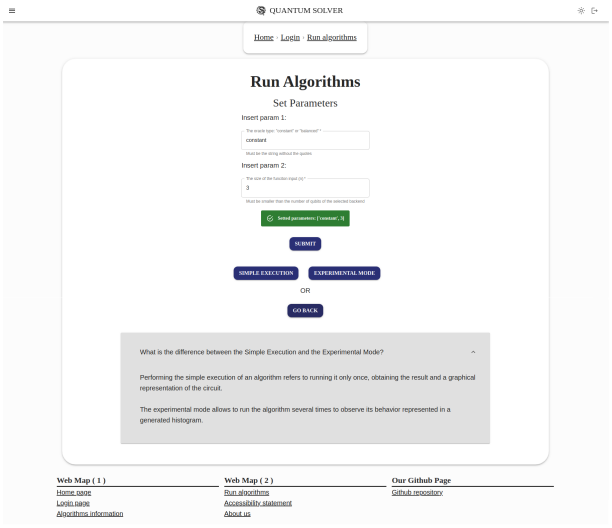


(c) Pantalla de declaración de la accesibilidad

Figura 2.5: Páginas de información



(a) Seleccionar backend y algoritmo

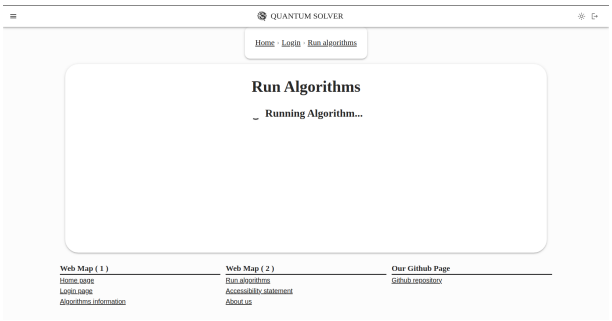


(b) Introducir parámetros y elegir tipo de ejecución

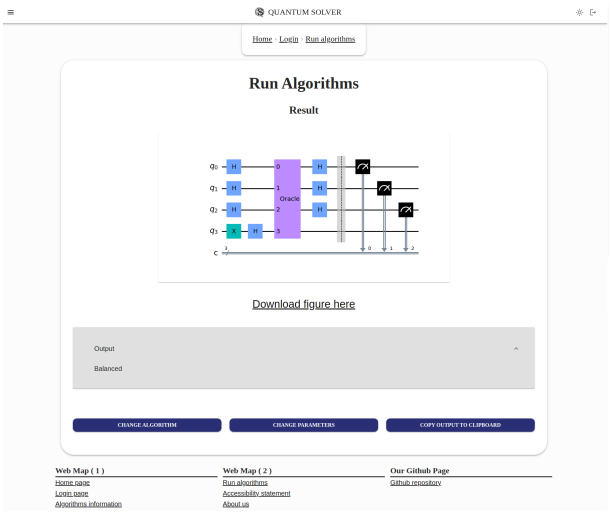
Figura 2.6: Páginas para la ejecución de algoritmos

todos los usuarios.

La muestra utilizada para llevar a cabo la auditoría abarca todas las rutas desarrolladas en la aplicación web. El motivo principal de utilizar una muestra tan amplia es garantizar

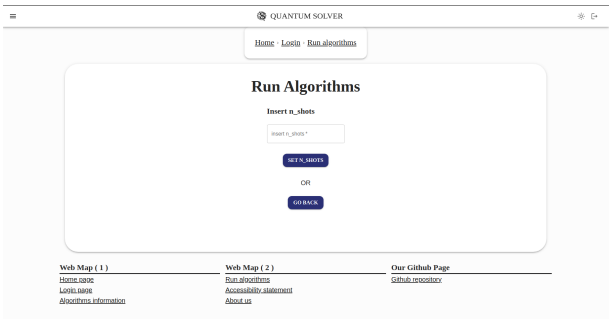


(a) Pantalla de carga de la ejecución simple



(b) Pantalla del resultado para la ejecución simple

Figura 2.7: Página de ejecución de algoritmos. Ejecución simple

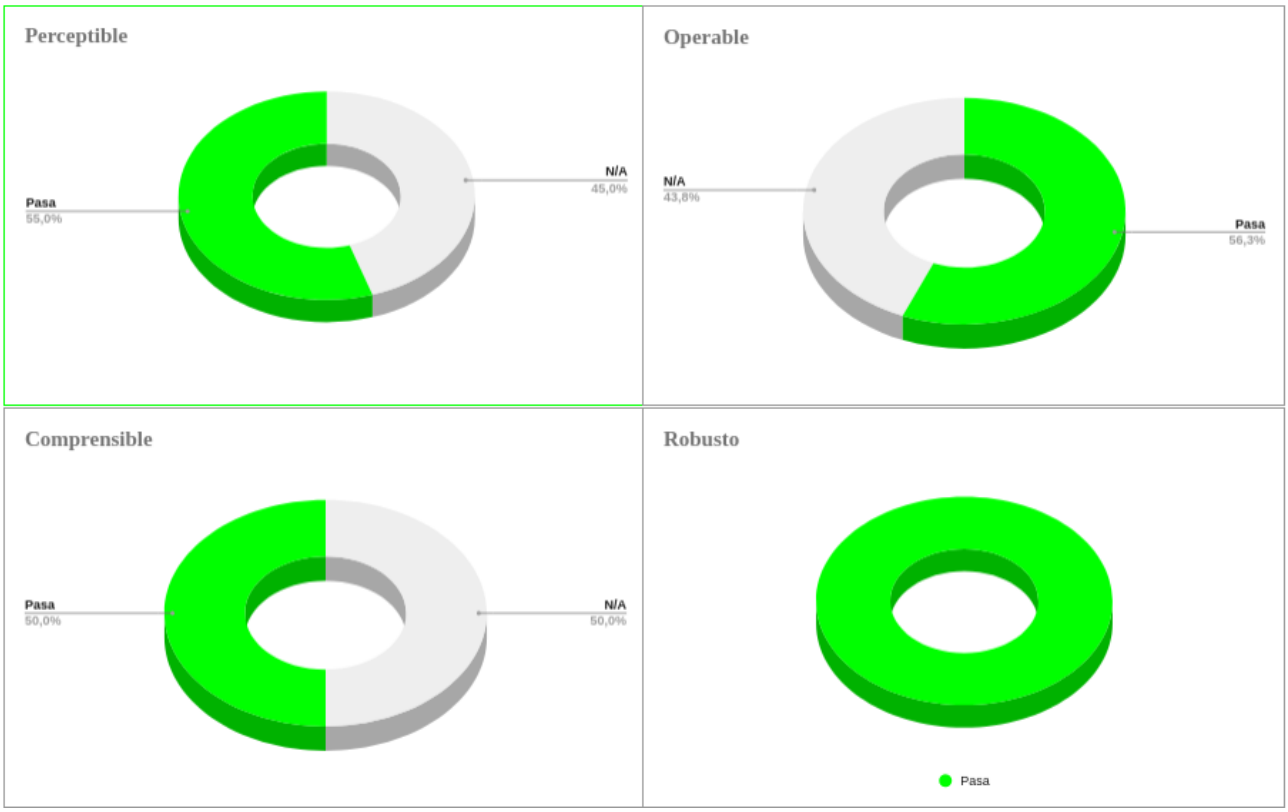


(a) Pantalla de introducción del número de ejecuciones del modo experimental

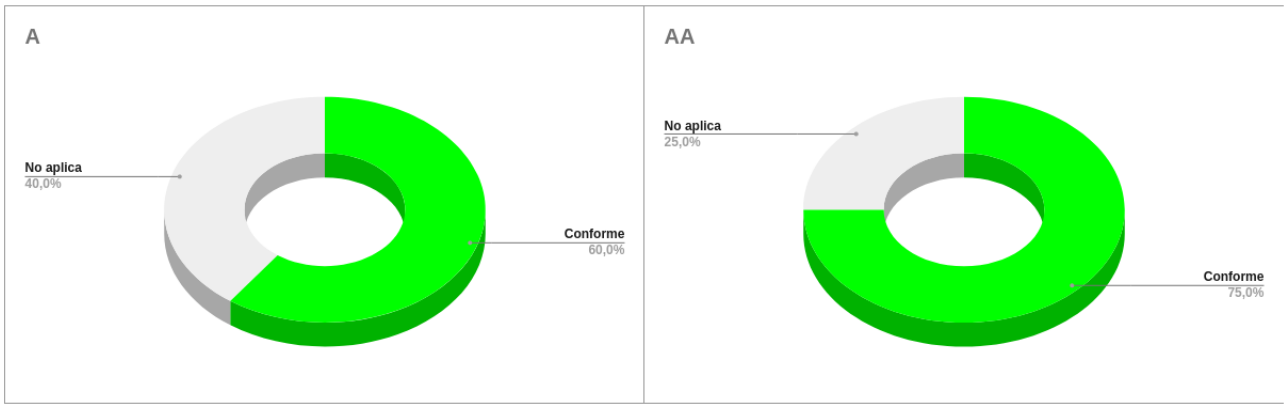
(b) Pantalla del resultado para el modo experimental

Figura 2.8: Página de ejecución de algoritmos. Modo experimental

la accesibilidad de todo el sitio web. Los resultados de la auditoría (22) realizada se pueden observar en la Fig. 2.9), donde se muestran dos gráficas. La primera muestra los niveles de cumplimiento para cada uno de los principios de la accesibilidad, mientras que la segunda representa el porcentaje total de cumplimiento de los de los niveles de conformidad. Para una correcta interpretación de los datos de las gráficas, los principios y los criterios de conformidad que se han implementado de manera satisfactoria se representan con color Verde, mientras que el color gris representan los que no son aplicables. Los criterios no aplicables fundamentalmente se tratan de funcionalidades que no incluye en su funcionamiento *QuantumSolver*. Un ejemplo de criterio que no aplica puede ser el principio Perceptible AA 1.2.4 Subtítulos (en directo), “se cumple si se proporcionan subtítulos para todo el contenido de audio en directo de los multimedia sincronizados“. Este criterio no se aplica ya que la página web no contiene ningún vídeo que requiera subtítulos.



(a) Porcentajes del cumplimiento de cada principio



(b) Porcentajes del cumplimiento de los criterios de conformidad totales

Figura 2.9: Porcentajes de cumplimiento y criterios de conformidad

Capítulo 3

Algoritmo de Simon

El problema de Simon (23) fue planteado como caso particular del problema del subgrupo oculto abeliano, subgrupo que tiene como característica que si cambias el orden de los elementos que estás sumando o multiplicando, el resultado que se obtiene es el mismo. Por ello, Simon permite encontrar el subgrupo oculto dentro de un grupo dado con la realización de consultas a una función que es periódica con respecto al grupo (24). Este fue el primer algoritmo cuántico en mostrar una aceleración exponencial en comparación con el mejor algoritmo clásico para resolver un problema específico. Además, inspiró los algoritmos cuánticos basados en la transformada cuántica de Fourier, como el algoritmo de factorización de Shor.

3.1. Definición del problema

El problema de Simon se focaliza en determinar qué condiciones prevalecen para una función f dada. Siendo $f : \{0,1\}^n \to \{0,1\}^n$ se garantiza sea inyectiva o sobreyectiva :

1. Inyectiva $(1 : 1)$: La función f asigna exactamente una salida única para cada entrada, como muestra la Ec. (3.1).

2. Sobreyectiva $(2 : 1)$: La función f asigna a una única salida dos entradas. Para conseguir este efecto, existe una cadena b no trivial que cumple la Ec. (3.2) donde el símbolo $\oplus$ denota la operación binaria *XOR*.

$$\forall x \in \{0,1\}^n, \nexists y \in \{0,1\}^n, f(x) = f(y) \wedge y = x \oplus b \tag{3.1}$$

$$\forall x, y \in \{0,1\}^n, f(x) = f(y) \iff y = x \oplus b \tag{3.2}$$

Un ejemplo sencillo de los dos posibles funcionamientos de la función f se puede observar en la Tabla (3.1).

El algoritmo de Simon consta de dos partes diferenciadas: la subrutina cuántica, encargada de buscar el periodo de la función f y el algoritmo clásico especializado en resolver el sistema de ecuaciones resultante de la etapa anterior, para conseguir el valor de b. El oráculo Q_f que codifica el funcionamiento de f en la subrutina cuántica debe tener unas características específicas que se comentan a continuación.

Al caracterizar una función que cumpla los requisitos antes descritos y que haga el efecto de *caja negra* se deben seguir una serie de pasos que se definirán en este apartado.

(1:1)	(2:1)
$F(0) \to 0$	$F(0) \to 0$
$F(1) \to 1$	$F(1) \to 1$
$F(2) \to 2$	$F(2) \to 0$
$F(3) \to 3$	$F(3) \to 1$

Tabla 3.1: Ejemplo de funcionamiento de la función F con b de tamaño 2

El oráculo recibe $|x\rangle |0\rangle^{\otimes n}$ como entrada y en base a un b predeterminado, el oráculo escribe su salida en el segundo registro de modo que transforma la entrada a $|x\rangle |f_b(x)\rangle$, aplicando de esta manera la función f, debido a que se define como $f_b(x) = f(x \oplus b)$ para todo $x \in \{0,1\}^n$. Para conseguir este efecto es necesario realizar los siguientes pasos:

1. Consiste en replicar el contenido del primer registro en el segundo $|x\rangle |0\rangle \to |x\rangle |x\rangle$. Esto se puede hacer debido a que los cúbits de entrada solo pueden estar en los estados base $|0\rangle$ o $|1\rangle$ y aplicando la operación *CX*.
2. Consiste en aplicar el comportamiento de f emulando la función inyectiva (1:1) o sobreyectiva (2:1). Esto se consigue viendo la forma de la cadena b pasada al oráculo definir su comportamiento.
   - *a*) En caso de que b sea todo ceros no se aplica ninguna operación extra.
   - *b*) En caso de que b no sea todo ceros implica que existe un índice mínimo j tal que $b_j = 1$. Conociendo esto se realiza una operación *XOR* con el segundo registro con b Ec. (3.3).
3. Consiste en aplicar permutaciones aleatorias en el segundo registro. Para generar el efecto de caja negra.

$$\text{Si } x_j \text{ en } b_j = 1 : |x\rangle |x\rangle \to |x\rangle |x \oplus b\rangle \tag{3.3}$$

3.2. Desarrollo del algoritmo

A continuación se desarrolla una explicación matemática formal sobre la evaluación del estado cuántico en la ejecución del circuito cuántico de Simon descrito en la Fig. 3.1.

1. Para una entrada de n-qbits los registros se inicializan a ceros (véase Ec. (3.4)).

$$|\psi_1\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n} \tag{3.4}$$

2. Se aplica la puerta Hadamard al primer registro, obteniendo Ec. (3.5).

$$|\psi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |0\rangle^{\otimes n} \tag{3.5}$$

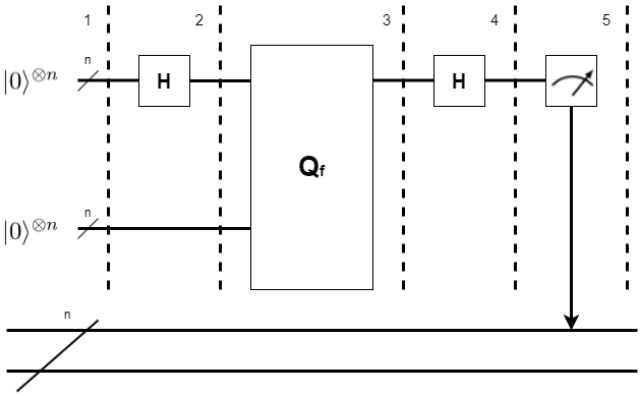


Figura 3.1: Circuito cuántico de Simon

3. El comportamiento general del oráculo sobre un registro cuántico arbitrario viene dado por la Ec. (3.6). Además, sobre el estado $|0\rangle^{\otimes n}$ se comporta estableciendo ese registro a la evaluación de la función sobre la entrada, como se describe en la Ec. (3.7). Tras realizar las transformaciones descritas, el oráculo Q_f opera sobre ambos registros dando como resultado Ec. (3.8).

$$Q_f(|x\rangle\,|a\rangle) = |x\rangle\,|a \oplus f(x)\rangle \tag{3.6}$$

$$Q_f(|x\rangle\,|0\rangle^{\otimes n}) = |x\rangle|f(x)\rangle \tag{3.7}$$

$$|\psi_3\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle\,|f(x)\rangle \tag{3.8}$$

Una vez aplicado el oráculo se consigue que el primer registro pueda llegar a tomar dos valores posibles $|x\rangle$ o $|y\rangle$ (véase Ec. (3.9)) dando como resultado la expresión Ec. (3.10).

$$\begin{cases} x & \\ y & = x \oplus b \end{cases} \tag{3.9}$$

$$|\psi_4\rangle = \frac{1}{\sqrt{2}}\left(|x\rangle + |y\rangle\right) \tag{3.10}$$

4. Se aplica nuevamente una puerta de Hadamard al primer registro, obteniendo con ello Ec. (3.11).

$$|\psi_5\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{z\in\{0,1\}^n} \left[(-1)^{x\cdot z} + (-1)^{y\cdot z}\right]|z\rangle \tag{3.11}$$

5. El último paso del circuito es medir el primer registro dando una salida únicamente si se cumple la expresión Ec. (3.12).

$$
\begin{aligned}
(-1)^{x\cdot z} &= (-1)^{y\cdot z} \\
x\cdot z &= y\cdot z \\
x\cdot z &= (x\oplus b)\cdot z \\
x\cdot z &= x\cdot z \oplus b\cdot z \\
b\cdot z &= 0 \pmod 2
\end{aligned}
\tag{3.12}
$$

Una vez se extrae el resultado de la subrutina cuántica se procede a realizar el procesado clásico del algoritmo de Simon. En el resultado de la rutina se medirá una cadena z cuyo producto interno con b es igual a cero. Repitiendo el algoritmo aproximadamente n veces, se podrán obtener n valores diferentes de z y se podrá escribir el sistema de ecuaciones definido en la expresión Ec. (3.13).

$$
\begin{cases}
b\cdot z_1 = 0 \\
b\cdot z_2 = 0 \\
\quad\vdots \\
b\cdot z_n = 0
\end{cases}
\tag{3.13}
$$

A partir de este sistema de ecuaciones es posible despejar y obtener el valor de b. Existen múltiples opciones para resolver el sistema de ecuaciones resultante.

3.3. Resultados obtenidos

Dentro de los resultados en la ejecución se puede observar claramente la diferencia del comportamiento del oráculo Q_f según si nos encontramos en el caso que la función f sea inyectiva (1:1) o sobreyectiva (2:1).

- Inyectiva (1:1): El circuito resultante generado por Q_f se puede observar en la Fig. 3.2 dando como resultado tras su ejecución el sistema de ecuaciones Ec. (3.14) que al ser resuelto se obtiene $b = 000$.
- Sobreyectiva (2:1): El circuito resultante generado por Q_f se puede observar en la Fig. 3.3 definiendo como resultado al ser ejecutado el sistema de ecuaciones Ec. (3.15) que al ser resuelto brinda un valor de $b = 101$.

$$
\begin{aligned}
b\cdot 101 &= b0\cdot 1 + b1\cdot 0 + b2\cdot 1 = 0 \pmod 2 \\
b\cdot 010 &= b0\cdot 0 + b1\cdot 1 + b2\cdot 0 = 0 \pmod 2 \\
b\cdot 001 &= b0\cdot 0 + b1\cdot 0 + b2\cdot 1 = 0 \pmod 2 \\
b\cdot 111 &= b0\cdot 1 + b1\cdot 1 + b2\cdot 1 = 0 \pmod 2 \\
b\cdot 011 &= b0\cdot 0 + b1\cdot 1 + b2\cdot 1 = 0 \pmod 2 \\
b\cdot 110 &= b0\cdot 1 + b1\cdot 1 + b2\cdot 0 = 0 \pmod 2 \\
b\cdot 100 &= b0\cdot 1 + b1\cdot 0 + b2\cdot 0 = 0 \pmod 2
\end{aligned}
\tag{3.14}
$$

$$
\begin{aligned}
b\cdot 101 &= b0\cdot 1 + b1\cdot 0 + b2\cdot 1 = 0 \pmod 2 \\
b\cdot 111 &= b0\cdot 1 + b1\cdot 1 + b2\cdot 1 = 0 \pmod 2 \\
b\cdot 010 &= b0\cdot 0 + b1\cdot 1 + b2\cdot 0 = 0 \pmod 2
\end{aligned}
\tag{3.15}
$$

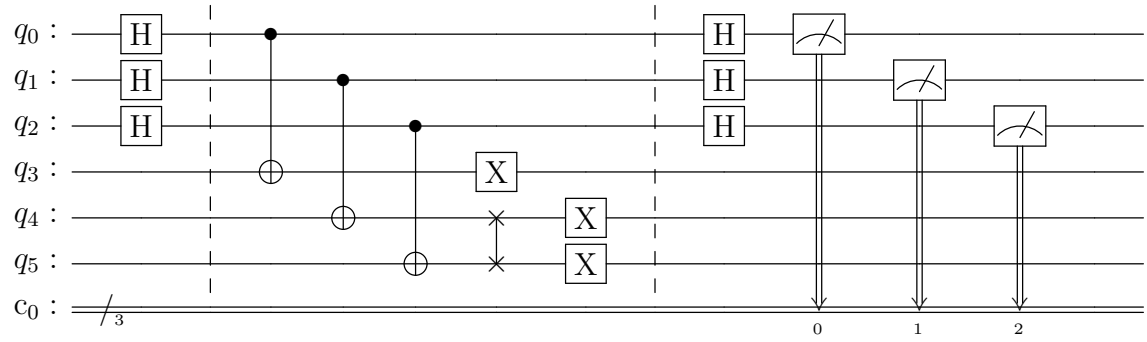


Figura 3.2: Ejecución con $b = 000$

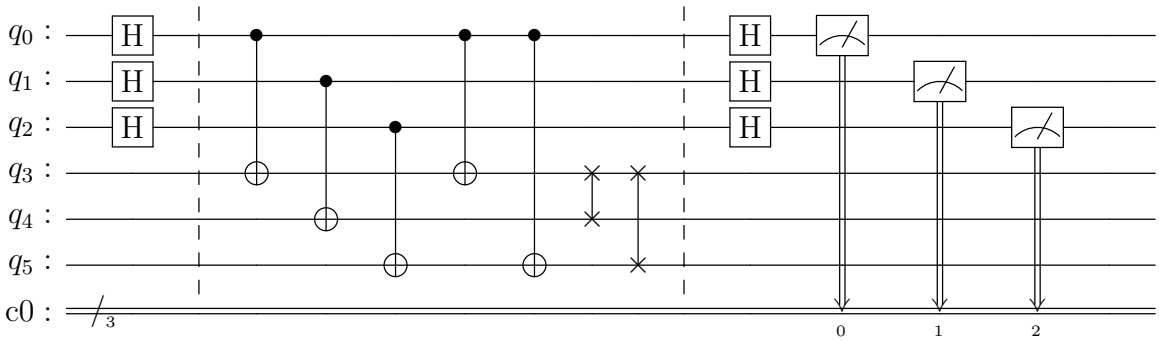


Figura 3.3: Ejecución con $b = 101$

El método elegido para la resolución de los sistemas de ecuaciones en la implementación fue la función $nullspace$ de la librería $sympy$. El método $nullspace$ se encarga de resolver el espacio nulo de una matriz A que es el conjunto de todos los vectores x que satisfacen la ecuación $Ax = 0$. Este problema es equivalente al sistema de ecuaciones Ec. (3.13) pudiendo interpretar el resultado de la siguiente manera: si el espacio nulo de una matriz es vacío, significa que no hay ningún vector x que satisfaga esta ecuación; por lo tanto, la solución será una b con todo ceros. Es decir, f se encuentra en el caso inyectiva (1:1). En caso contrario, se obtendrá el valor de b que resuelva el problema con una función f del tipo sobreyectiva (2:1).

```
    # Solve the system of equations
    # Eliminate the rows with all zeros
    aux = []
    for z in result:
      if not all(char in "0" for char in z) and z not in aux:
        aux.append(z)

    # Check if there is enough data to form a matrix
    num_rows = len(aux)
    num_cols = n
    if num_rows < num_cols:
      print(f"Not enough data to form a matrix (need at least {num_cols}
          equations)")
      return "error"

    # Create the matrix A and the vector b
    A = np.array([[int(i) for i in z] for z in aux])
    b = np.zeros(len(aux))


    print("\nSolving the system of equations...")
    try:
      b = self.get_solution(result)
    except Exception as e:
      return "Error solving the system of equations:" + str(e)

    # Return the solution of the system.
    return b if (b != None) else "".join([str(int(0)) for x in range(n)])
```

Capítulo 4

Algoritmo de Shor

El algoritmo de Shor (25) resuelve el problema de la factorización de un número entero de una manera eficiente (14). Muchos algoritmos criptográficos de clave pública como *RSA*, para su funcionamiento se basan en la multiplicación de dos números primos de gran tamaño. A continuación, se procederá a explicar de manera detallada el funcionamiento del algoritmo de cifrado de RSA para conocer de qué manera se vería afectado por el algoritmo de Shor.

1. Información privada
   a) Dos números primos de gran tamaño que se llamarán p y q.
   b) $\Phi(N) = (p-1) \cdot (q-1)$
   c) d es entero primo con $\Phi(N)$
2. Información pública
   a) $N = p \cdot q$
   b) e es el inverso de d módulo $\Phi(N)$
3. Operación de cifrado
   a) Siendo M_i el texto original
   b) $C_i = M_i^e \pmod N$
4. Operación de descifrado
   a) Siendo C_i el texto cifrado
   b) $M_i = C_i^d \pmod N$

Durante el proceso, los datos públicos son compartidos entre el emisor y el receptor para llevar a cabo el cifrado *RSA*. Entre esos datos públicos figura el valor de N, que si pudiera ser factorizado permitiría obtener los valores privados p y q.

De hecho, la seguridad del sistema *RSA* depende de que se desconozcan los valores p y q. Por este motivo, en el momento en que haya disponibilidad de ordenadores cuánticos con suficiente cúbits, dichos algoritmos de clave pública serían vulnerables y por tanto quedarían obsoletos.

4.1. Definición del problema

El problema de la factorización de un entero N consiste en encontrar enteros entre 1 y N que sean divisores enteros de N. La aproximación al problema de la factorización seguida por el algoritmo de Shor se describe brevemente a continuación. Sea r el orden de un entero a módulo N, es decir, el menor entero positivo tal que $a^r = 1 \pmod N$. En ese caso, obviamente $a^{r+1} = a \pmod N$. Esto a su vez se corresponde con que r es el período de la función $f(x) = a^x \pmod N$, pues es el menor entero r tal que $f(x + r) = f(x)$.

Por otra parte, si el orden r es par, se tiene que $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1) = 0 \pmod N$. De ahí se deduce que, como N no puede dividir a $(a^{r/2} - 1)$ ni a $(a^{r/2} + 1)$ porque r es el menor entero positivo tal que $a^r = 1 \pmod N$, entonces N debe tener un factor común no trivial con $(a^{r/2} - 1)$ y con $(a^{r/2} + 1)$, es decir, se logra factorizar N.

Así, la búsqueda de factores se realiza en el algoritmo de Shor mediante dos partes diferenciadas:

1. Subrutina cuántica para resolver el problema de encontrar el periodo de una función.
2. Subrutina clásica para resolver el problema de encontrar el orden de un entero módulo otro, es decir, el menor exponente positivo tal que al elevar el primer entero a ese exponente se obtiene 1 en módulo el segundo entero.

4.2. Desarrollo del algoritmo

La subrutina cuántica del algoritmo de Shor permite encontrar el periodo de la función que define una exponenciación modular. Dicha subrutina consta de tres etapas:

1. A los cúbits de entrada se les aplica la puerta Hadamard para ponerlos en estado de superposición.
2. Se aplica el circuito encargado de la exponenciación modular, que de manera general se describe como $a^r \pmod N$.
3. Se aplica la inversa de la transformada cuántica de Fourier $QFT^\dagger$ definida en la siguiente sección.

La función encargada de la exponenciación modular es la parte más compleja de la implementación del algoritmo de Shor. Esto se debe a que actualmente no se cuenta con una manera general factible para calcularla, porque aunque existen aproximaciones (26), las implementaciones se realizan mediante prueba y error. Uno de los factores principales que añaden dificultad a ese computo es la gran cantidad de cúbits requeridos , si se toma en cuenta la potencia y el tiempo de coherencia de los ordenadores cuánticos actuales.

En la implementación realizada para la factorización del número $N = 21$, la función de exponenciación modular utilizada fue $a^r \pmod{21}$ (27), y el circuito resultante se muestra en la Fig. 4.1.

El algoritmo correspondiente a la factorización del número $N = 21$ requiere de: 5 cúbits, un registro de trabajo conformado por dos cúbits (q_0, q_1) y un registro de control con tres cúbits (q_2, q_3 y q_4). El registro de trabajo se representa mediante tres estados base de un sistema de dos cúbits, descartándose el cuarto estado base como estado nulo. Los estados que se codifican son $|1\rangle$, $|4\rangle$ y $|16\rangle$, y se mapean siguiendo la Ec. (4.1).

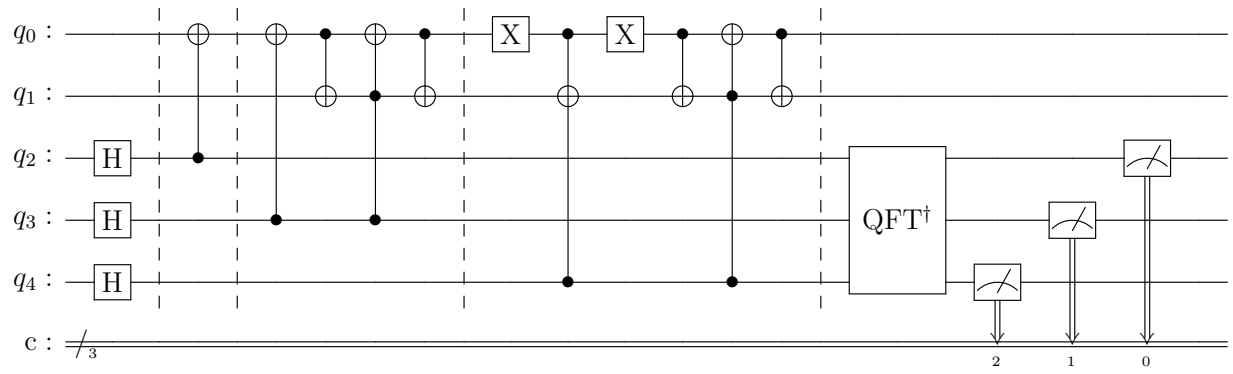


Figura 4.1: Circuito de Shor implementado para N=21

$$\begin{aligned} |1\rangle &\mapsto |log_4 1\rangle = |00\rangle \\ |4\rangle &\mapsto |log_4 4\rangle = |01\rangle \\ |16\rangle &\mapsto |log_4 16\rangle = |10\rangle \end{aligned} \tag{4.1}$$

Para calcular la exponenciación se emplean las puertas de control unitario $\hat{U}^x$ siguiendo la expresión Ec. (4.2) que reduce al intercambiar los estados $|1\rangle$, $|4\rangle$ y $|16\rangle$ en el registro de trabajo controlado por el correspondiente bit del entero x en el registro de control, que viene dado por $x = q_4 2^0 + q_3 2^1 + q_2 2^2$. Es decir, $\hat{U}^x = \hat{U}^{q_2 2^2} \hat{U}^{q_3 2^1} \hat{U}^{q_4 2^0}$ así dependiendo del cúbit de control q_i, se aplica una de las codificaciones definidas en Ec. (4.3). Las operaciones del registro de trabajo no necesitan ser puertas $SWAP(Fredkin)$ controladas.

$$|x\rangle |y\rangle \rightarrow |x\rangle \hat{U}^x |y\rangle = |x\rangle |a^x y \text{ (mod } N)\rangle \tag{4.2}$$

$$\begin{aligned} \hat{U}^1 &: \{\ |1\rangle \mapsto |4\rangle, |4\rangle \mapsto |16\rangle, |16\rangle \mapsto |1\rangle\ \} \\ \hat{U}^2 &: \{\ |1\rangle \mapsto |16\rangle, |4\rangle \mapsto |1\rangle, |16\rangle \mapsto |4\rangle\ \} \\ \hat{U}^4 &: \{\ |1\rangle \mapsto |4\rangle, |4\rangle \mapsto |16\rangle, |16\rangle \mapsto |1\rangle\ \} \end{aligned} \tag{4.3}$$

Se procede a explicar cada una de las puertas $\hat{U}$ implementadas en la Fig. 4.1:

- $\hat{U}^1$ se simplifica mediante la observación que los estados $|4\rangle$ y $|16\rangle$ tienen inicialmente amplitud cero y por lo tanto la operación $|1\rangle \mapsto |4\rangle$ se puede realizar mediante el uso de la puerta CX controlada por $|q_4\rangle$ y dirigida al cúbit $|q_1\rangle$.
- $\hat{U}^2$ se puede simplificar observando los estados $|1\rangle$ y $|4\rangle$ ya que son los únicos estados de amplitud no nula en el registro de trabajo después que se haya aplicado $\hat{U}^1$, lo que conlleva considerar únicamente $|1\rangle \mapsto |16\rangle$ y $|4\rangle \mapsto |1\rangle$. Una puerta *CX* controlada por $|q_3\rangle$ dirigida a $|q_1\rangle$ seguida de una puerta *Swap*, intercambiando $|q_0\rangle$. y $|q_1\rangle$.
- $\hat{U}^4$ se realiza descomponiendo la puerta $Fredkin$ en una puerta *CCX (Toffoli)* y dos puertas *CX*. Esta puerta no admite simplificaciones ya que todos los posibles estados en el registro de trabajo pueden tener una amplitud distinta de cero en este punto. Esta operación se implementa con una puerta *CCX* y una puerta *Swap* con puertas *X* de un cúbit.

4.3. Transformada cuántica de Fourier

La transformada cuántica de Fourier (QFT, *Quantum Fourier Transform*) (28) es la analogía cuántica de la transformada discreta de Fourier clásica. A continuación se compara el funcionamiento de ambas.

La transformada discreta de Fourier unitaria actúa sobre un vector en $\mathbb{C}^n$, $(x_0, \ldots, x_{n-1})$ y lo mapea al vector $(y_0, \ldots, y_{n-1})$ siguiendo la expresión Ec. (4.4), donde la expresión Ec. (4.5) es una raíz n-ésima de la unidad primitiva,

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \omega_N^{jk} \tag{4.4}$$

$$\omega_N^{jk} = e^{2\pi i \frac{jk}{N}} \tag{4.5}$$

De manera similar, la transformada cuántica de Fourier actúa sobre un estado cuántico $|X\rangle = \sum_{j=0}^{N-1} x_j |j\rangle$ y lo mapea al estado cuántico $|Y\rangle = \sum_{k=0}^{N-1} y_k |k\rangle$ de acuerdo con la expresión Ec. (4.6).

$$|j\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{jk} |k\rangle \tag{4.6}$$

Por otro lado, de forma equivalente, la transformada cuántica de Fourier puede verse como una matriz unitaria actuando sobre vectores estado cuántico, de acuerdo con la expresión Ec. (4.7).

$$U_{QFT} = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} \omega_N^{jk} |k\rangle |x\rangle \tag{4.7}$$

Una de las propiedades más interesantes de la transformada cuántica de Fourier reside en el hecho de que se trata de una transformación unitaria. Esto se puede verificar realizando una multiplicación de matrices y comprobando la relación $F \cdot F^\dagger = F^\dagger \cdot F = I$ donde F es la hermítica adjunta de F. Por ese motivo, la inversa de la QFT es igual a la $QFT^\dagger$.

En el caso del algoritmo de Shor se utiliza la implementación de la inversa de la transformada cuántica de Fourier ($QFT^\dagger$), cuya implementación se puede observar en la Fig. 4.2.

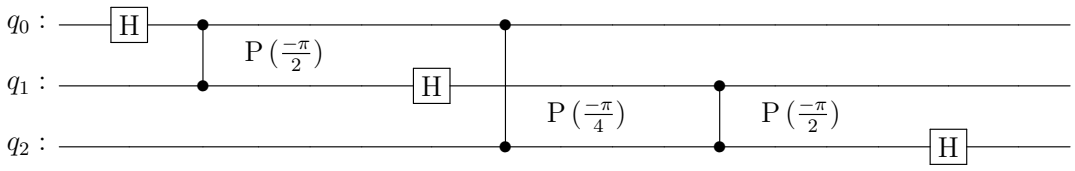


Figura 4.2: Ejemplo de circuito $QFT\dagger$

Una vez se concluye la subrutina cuántica, se procede a realizar el post-procesado clásico del resultado, para lo cual se realiza lo siguiente:

1. Se convierten los números binarios en decimal d y se obtiene la fase realizando la operación $phase = d/2^{n\_count}$.

2. Se busca la fracción más cercana al valor de la $phase$ con un denominador menor a N. De esta fracción se utiliza el valor del denominador dando con ello el valor de r.

3. Por cada valor de r se comprueba que el valor no sea impar.

4. Se realiza el cálculo $mcd(a^{r/2}-1, N)$ y $mcd(a^{r/2}+1, N)$, dando con ello los dos factores que multiplicados entre sí dan 21, que en este caso son 7 y 3.

4.4. Resultados

La ejecución del circuito cuántico representado en la Fig. 4.1 da como resultados los mostrados en la Fig. 4.3. Dichos datos concuerdan con los resultados esperados pues al ser procesados devuelven los factores 7 y 3.

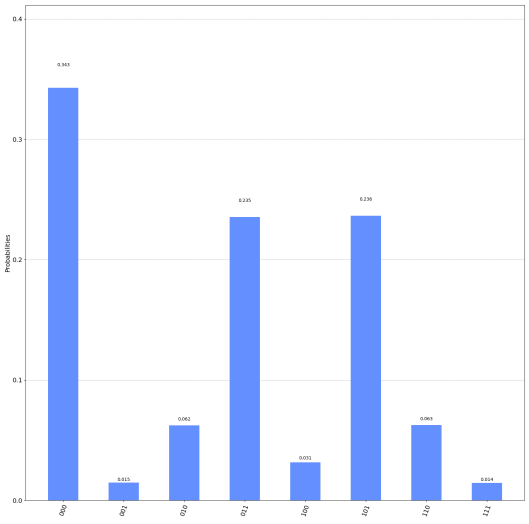


Figura 4.3: Resultados de la ejecución del algoritmo de Shor para N=21

Capítulo 5

Arquitectura hexagonal

A lo largo del desarrollo de los proyectos, es habitual que sean ampliados por diferentes programadores, de forma que cada uno de ellos aplica su propio estilo y concepción del código. Eso provoca inevitablemente que el código de la aplicación se vuelva más enrevesado e insostenible tras cada iteración.

Una arquitectura de software es un diseño a alto nivel que permite estructurar y organizar el código del proyecto de una manera más eficiente y manejable en el tiempo. También ayuda a disponer los componentes de un sistema, la forma en que se comunican entre sí y cómo se comportan en diferentes situaciones. Por ello, para evitar que el mantenimiento del proyecto de *QuantumSolver* se vuelva insostenible, se ha decidido en este trabajo realizar una propuesta de diseño de arquitectura hexagonal.

5.1. Diseño propuesto

La arquitectura hexagonal (29) (30), también conocida como arquitectura de puertos y adaptadores, es un patrón de software que se enfoca en separar la aplicación en distintas capas o regiones, cada una con su propia responsabilidad. De esta manera se consigue el objetivo de desacoplar las capas. La arquitectura hexagonal se divide en tres capas diferenciadas: infraestructura, aplicación y dominio. En la imagen de la Fig. 5.1 se plantea un esquema simplificado de la arquitectura hexagonal adaptado a las necesidades actuales y futuras de la librería *QuantumSolver*.

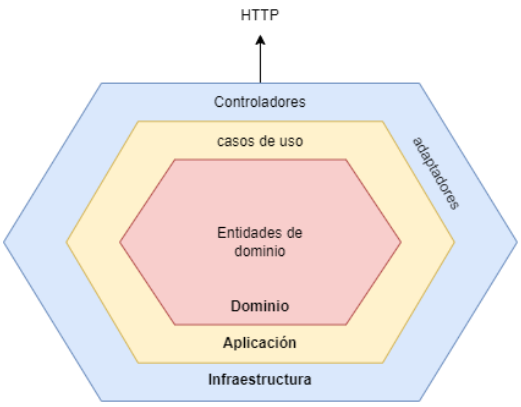


Figura 5.1: Arquitectura hexagonal

El corazón de la arquitectura hexagonal se encuentra en la capa de dominio, cuya función principal es albergar las reglas de negocio y garantizar la coherencia del estado de

los objetos del dominio. En línea con este enfoque, los algoritmos cuánticos de la biblioteca se consideran objetos de dominio, los cuales se definen siguiendo las características descritas en la Fig. 5.2. El circuito y los componentes que necesitan de alguna librería específica deben ser definidos como una interfaz a la que le llegan los datos de capas exteriores. Con esto se gana en independencia de implementación entre capas.

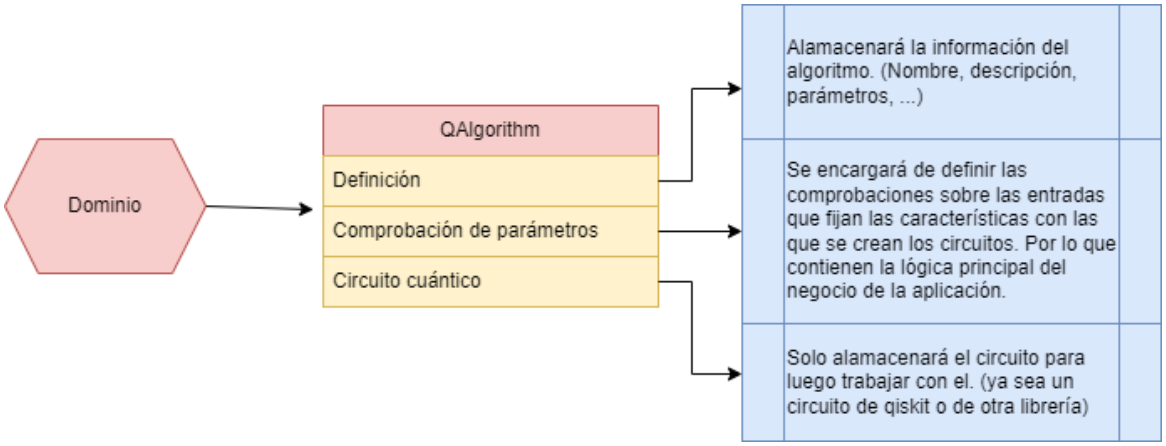


Figura 5.2: Capa de dominio de la arquitectura hexagonal

Seguidamente se define la capa de aplicación, responsable de actuar como intermediario entre la capa de dominio y la interfaz de usuario. Tiene como función orquestar las operaciones del dominio, asegurando que se ejecuten de acuerdo con las reglas de negocio y los requisitos del usuario. Como se ha explicado, esta capa es la encargada de manejar la comunicación con otros sistemas externos (veáse Fig. 5.3). Por ello, en esta capa se definen los servicios *QExecute* y *QAlgorithmManager*. Estos dos servicios servirán como manejadores para seleccionar los diferentes entornos de ejecución y algoritmos, respectivamente, llamando a sus correspondientes adaptadores.

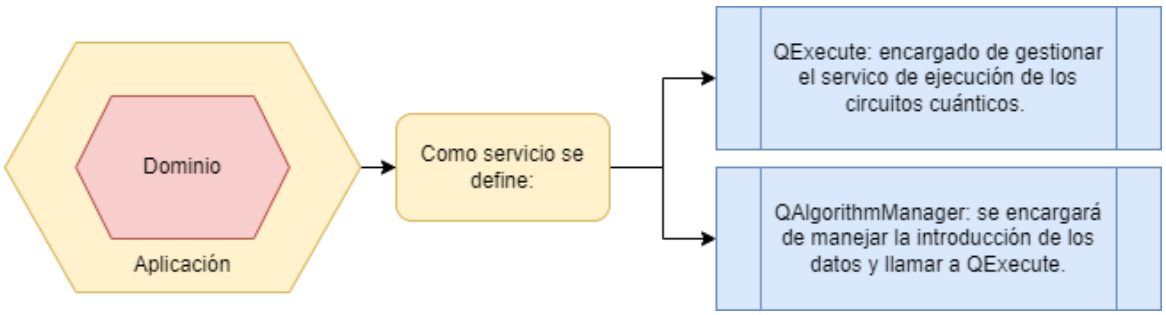


Figura 5.3: Capa de aplicación de la arquitectura hexagonal

Finalmente se encuentra la capa de Infraestructura. En ella se realizan las implementaciones de las conexiones con terceros que normalmente en el resto del proyecto se representan mediante interfaces. Con este propósito se definen los adaptadores mencionados en la capa anterior. Estos permiten en *QuantumSolver* desvincular la implementación del algoritmo y la ejecución con una librería en específico. En la Fig. 5.4 se define un ejemplo del funcionamiento de los adaptadores y de cómo permitirían la coexistencia de la implementación de los circuitos en *Qiskit* y *braket*. En esta capa también se describe el entorno *CLI* y *Framework* de la librería. Se incluye un ejemplo de la implementación con *Qiskit* y *braket*. Además en infraestructura se describe el entorno y *framework*.

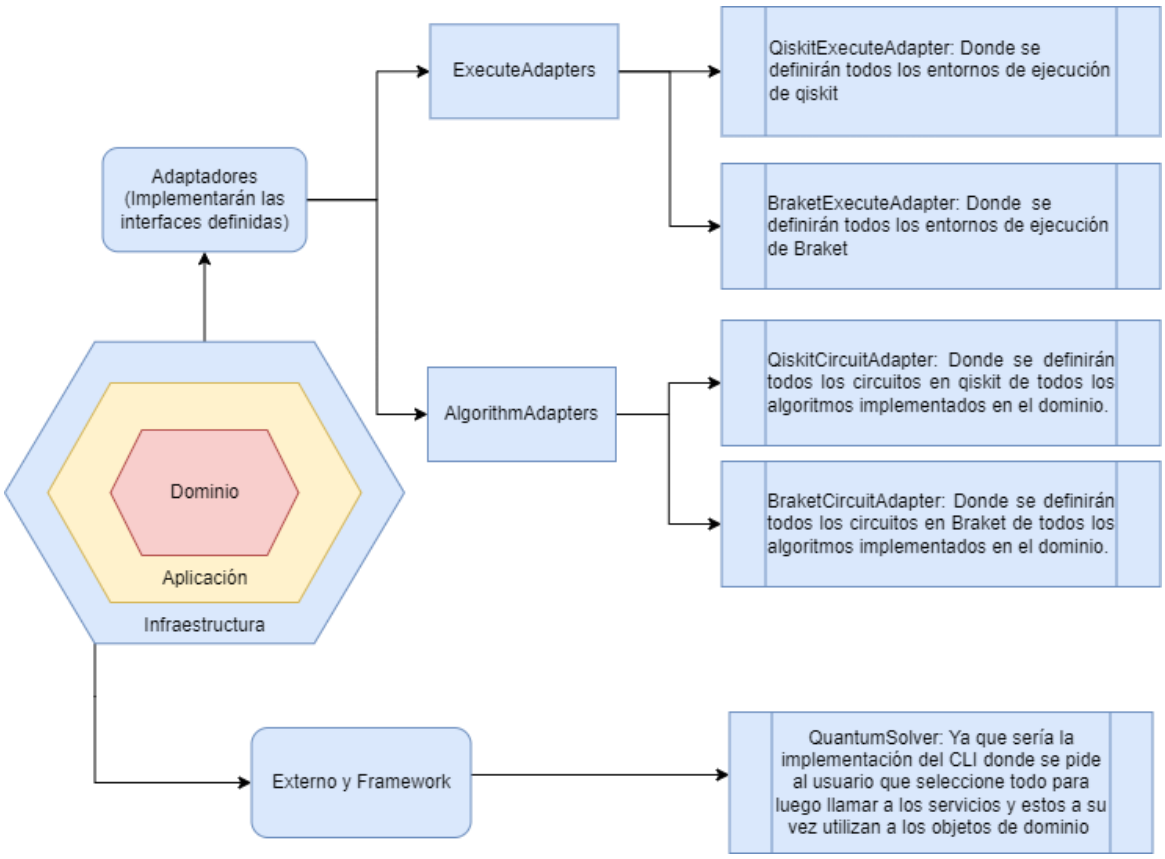


Figura 5.4: Capa de infraestructura de la arquitectura hexagonal

5.2. Funcionamiento

Siguiendo el modelo planteado, el funcionamiento quedaría de la siguiente manera:

- Seleccionar el Backend llamando al servicio *QExecute*. Este a su vez llamaría al adaptador correspondiente al tipo de Backend llamado (ya sea en el caso *Qiskit* o *Braket*), y este manager lo guardaría hasta su llamada a ejecución más adelante.
- Seleccionar el algoritmo llamando al servicio *QAlgorithmManager* encargado. Al igual que pasa en el Backend, llamando al correspondiente adaptador y este devolviendo el circuito, pero con la diferencia de que en este punto se devuelve el adaptador siguiendo el esquema planteado en la capa de dominio.

Finalmente, el repositorio de la librería seleccionada *Qiskit* o *Braket* se pasaría desde infraestructura al servicio de aplicación.

En conclusión, en la actualidad la librería *QuantumSolver* se está viendo afectada por una serie de limitaciones al no seguir una arquitectura de software bien definida. Implementando una arquitectura de proyecto como la propuesta a lo largo de este capítulo, no solo se consigue una mayor separación del código evitando dependencias innecesarias, sino que también se conseguiría evitar la dependencia que tiene en su funcionamiento con respecto a un único kit de desarrollo cuántico, *Qiskit*.

Capítulo 6

Conclusiones y líneas futuras

Durante el desarrollo de este Trabajo Final de Grado se ha desarrollado una aplicación web accesible, intuitiva y centrada en las necesidades especificas del usuario final. Estas características permitirán alcanzar una mayor difusión del proyecto, y en general mejorar la experiencia de usuario en el entorno generado. Además se han incluido en la librería cuántica objeto del trabajo, dos nuevos algoritmos cuánticos, el de Simon y el de Shor en *QuantumSolver*, dentro de un nuevo módulo llamado *QuantumSolverSubrutine*. La adición de estos dos algoritmos a la librería tiene la finalidad mostrar las ventajas que ofrece el paradigma híbrido de la computación cuántica/clásica, y cómo ese paradigma puede ayudar a mejorar las capacidades de cómputo. Por último, la arquitectura hexagonal diseñada permite eliminar la dependencia que tiene la librería con respecto a un solo kit de desarrollo cuántico. Posibles líneas futuras a desarrollar son las siguientes:

1. La refactorización del código de la librería *QuantumSolver*, siguiendo el diseño de la arquitectura hexagonal definida en el proyecto.

2. La mejora de la parametrización del algoritmo de Shor implementado, añadiendo nuevos circuitos que permitan la factorización de más números aparte de $N = 21$

3. La mejora de los resultados obtenidos gracias a la aplicación de la *QuantumSolverSubrutine*, permitiendo el análisis de los parámetros y la elección de diferentes entornos de ejecución.

4. La adición a la página web de los módulos restantes en el CLI de *QuantumSolver*, siguiendo las recomendaciones de accesibilidad marcadas en el proyecto.

Capítulo 7

Conclusions and future work

During the development of this Final Degree Project, an accessible, intuitive web application focused on the specific needs of the end user has been developed. These features will make it possible to achieve a greater diffusion of the project, and in general improve the user experience in the generated environment. In addition, two new quantum algorithms (Simon and Shor), have been included in the quantum library object of this work, *QuantumSolver*, within a new module called *QuantumSolverSubrutine*. The inclusion of these two algorithms to the library is intended to show the advantages of the hybrid paradigm of quantum/classical computing, and how that paradigm can enhance computational capabilities. Finally, the designed hexagonal architecture eliminates the dependency of the library on a single quantum development kit. Possible future directions for development are as follows:

1. The refactoring of the *QuantumSolver* library code, following the hexagonal architecture design defined in the project.

2. The improvement of the parameterization of the implemented Shor algorithm, adding new circuits that allows the factorization of more numbers apart from $N = 21$.

3. The improvement of the results obtained through the implementation of the *QuantumSolverSubrutine* algorithms, allowing the analysis of the parameters and the choice of different execution environments.

4. The addition to the web page of other modules present in the *QuantumSolver* CLI, following the accessibility recommendations marked in this project.

Capítulo 8

Presupuesto

En este apartado se realiza una estimación de los costes incurridos en el proyecto tomando en consideración tanto las horas imputadas al mismo, así como el equipo informático utilizado. También se estima el valor incurrido en elementos gráficos para el diseño de la página web.

8.1. Costes de horas imputadas

Actividad	Duración	Coste Unitario	Coste total
Estudio bibliográfico	70h	24€ / h	1680€
Planificación del proyecto	15h	24€ / h	360€
Desarrollo del proyecto	200h	28€ / h	5600€
Redacción de la memoria	50h	24€ / h	1200€
Total	335h		8840€

Tabla 8.1: Costes de horas imputadas

8.2. Costes de equipo informático

El ordenador portátil utilizado para el desarrollo del Trabajo final de grado se encuentra valorado en 700€, sus especificaciones vienen dada en la tabla 8.2

8.3. Costes elementos gráficos

Para el desarrollo de los elementos gráficos de la página web se ha realizado una estimación de precios del valor de los mismos en el mercado, viendo sus valores en la tabla 8.3

Componente	Especificación
Procesador	11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz
RAM	16,0 GB (15,7 GB usable)
Memoria SSD	475 GB
Sistema Operativo	Windows 11 pro

Tabla 8.2: Especificaciones del equipo

Componente	Especificación
Logos (Blanco y negro)	500€
Banner	250€
Total	750€

Tabla 8.3: Costes elementos gráficos

8.4. Costo total

Concepto	Precio
Costes de horas imputadas	8840€
Costes de equipo informático	700€
Costes elementos gráficos	750€
Total	10290€

Tabla 8.4: Costos totales

Apéndice A

Congreso aceptado

Implementación de los Algoritmos Cuánticos de Simon y de Shor.
Vlatko Marchan-Sekulic, Pino Caballero-Gil, Daniel Escanez-Exposito
VIII Jornadas Nacionales de Investigación en Ciberseguridad (JNIC).
Vigo. 21-23 de junio de 2023.

Bibliografía

[1] Stewart I., Ilie D., Zamyatin A., Werner S., Torshizi M.F. Knottenbelt W.J.: “Committing to quantum resistance: a slow defence for Bitcoin against a fast quantum computing attack”. Royal Society open science 5(6), 180410, 2018.

[2] Rivest, R., Shamir, A., Adleman, L.: “RSA A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”. Communications of the ACM 21(2), 120-126, 1978.

[3] Castelvecchi, D.: “Google’s quantum computer hits key milestone by reducing errors”. Nature, 2023. [online]. Available: https://doi.org/10.1038/d41586-023-00536-w [Accessed 07-May-2023]

[4] Google Quantum AI: “Suppressing quantum errors by scaling a surface code logical qubit”. Nature 614, 676–681, 2023.

[5] Organización Mundial de la Salud (OMS): “Disability”. 2023. [online]. Available: https://www.who.int/news-room/fact-sheets/detail/disability-and-health [Accessed 02-May-2023]

[6] Fundación ONCE: “Tecnología y recursos adaptados”. [online]. Available: https://www.once.es/servicios-sociales/tecnologiayrecursosadaptados [Accessed 02-May-2023]

[7] Escanez-Exposito, D.: “QuantumSolver”. [online]. Available: https://github.com/jdanielescanez/quantum-solver [Accessed 02-May-2023]

[8] “Learn React”. [online]. Available: https://react.dev/learn [Accessed 08-May-2023]

[9] “Redux Usage Guides”. [online]. Available: https://redux.js.org/usage/ [accessed 08-May-2023]

[10] “Redux Essentials”. [online]. Available: https://redux.js.org/tutorials/essentials/part-1-overview-concepts [accessed 08-May-2023]

[11] “Material UI (MUI)”. [online]. Available: https://mui.com/material-ui/getting-started/overview/ [accessed 08-May-2023]

[12] “Material Design”. [online]. Available: https://m3.material.io/ [Accessed: 14-May-2023]

[13] IBM, “Start learning in the best way for you”. [online]. Available: https://qiskit.org/learn/ [Accessed: 08-May-2023].

[14] Nielsen, M., Chuang, I.: “Quantum Computation and Quantum Information”. Cambridge Series on Information and the Natural Sciences, Cambridge University Press, 2000.

[15] “Mockups iniciales”. [online]. Available: https://www.figma.com/file/pAiUWa1CHKLhMzDGhUJuSZ/TFG?type=design&node-id=0%3A1&t=HWfcYOAaJkJckEFY-1 [Accessed: 08-May-2023]

[16] “Fork QuantumSolver”. [online]. Available: https://github.com/alu0101321141/quantum-solver.git [Accessed: 08-May-2023]

[17] “Start a New React Project”. [online]. Available: https://es.react.dev/learn/start-a-new-react-project [Accessed: 14-May-2023]

[18] “Getting Started Vite”. [online]. Available: https://vitejs.dev/guide/ [accessed 08-May-2023]

[19] “Axios”. [online]. Available: https://axios-http.com/es/docs/intro.[Accessed: 14-May-2023]

[20] “WAVE”. [online]. Available: https://wave.webaim.org/ [Accessed: 14-May-2023]

[21] “WCAG 2.1 de un vistazo”. [online]. Available: https://www.w3.org/WAI/standards-guidelines/wcag/glance/es [Accessed: 14-May-2023]

[22] “‘Informe de la auditoría”. [online]. Available: https://docs.google.com/spreadsheets/d/1eWy7F-PDL00pEfNZiI2sfkHxXRe1loZg/edit?usp=sharing&ouid=105808837602047841462&rtpof=true&sd=true [Accessed: 14-May-2023]

[23] Simon, D.R: “On the power of quantum computation”. SIAM journal on computing 26(5), 1474-1483, 1997.

[24] Cai, G., Qiu, D.: “Optimal separation in exact query complexities for Simon’s problem”. Journal of Computer and System Sciences 97, 83-93, 2018.

[25] Shor, P.W.: “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer”. SIAM J. Comput 26, 1484–1509, 1997.

[26] Ravuri, C.: “A General Implementation of Shor’s Algorithm”. [online]. Available: https://medium.com/mit-6-s089-intro-to-quantum-computing/a-general-implementation-of-shors-algorithm-da1595694430 [Accessed: 14-Apr-2023]

[27] Skosana, U., Tame, M.: “Demonstration of Shor’s factoring algorithm for N =21 on IBM quantum processors”. Scientific reports 11, 16599, 2021.

[28] Nielsen, M.A., Chuang, I.L.: “Quantum Computation and Quantum Information”. 10th Anniversary Edition, Cambridge University Press, 217-220, 2011.

[29] CloudAPPi “Introducción a la Arquitectura Hexagonal”. [online]. Available: https://cloudappi.net/introduccion-a-la-arquitectura-hexagonal/ [Accessed: 10-May-2023]

[30] Novoseltseva, E.: “Hexagonal Architecture”. [online]. Available: https://apiumhub.com/tech-blog-barcelona/hexagonal-architecture/ [Accessed: 10-May-2023]