qPad Lab: The Quantum Circuit Laboratory for iPad

Desenvolupament d'una aplicació per a iPad per dissenyar circuits quàntics

Estudiant: David Arcos Gutiérrez
Màster Universitari en Desenvolupament d'Aplicacions per a Dispositius Mòbils

Consultor: Pau Dominkovics Coll
Professor responsable de l'assignatura: Carles Garrigues Olivella

5 de juny de 2018

Títol del treball:
qPad Lab: The Quantum Circuit Laboratory for iPad

Nom de l'autor:
David Arcos Gutiérrez

Nom del consultor:
Pau Dominkovics Coll

Nom del PRA:
Carles Garrigues Olivella

Data de lliurament:
06/2018

Titulació:
Máster Universitari en Desenvolupament d'Aplicacions per a Dispositius Mòbils

Idioma del treball:
Català

Resum:
Les aplicacions mòbils cada vegada tenen més presència en tots els àmbits de la societat. Per la seva banda, la computació quàntica, una disciplina que es troba en ple desenvolupament, comença a tenir els primers simuladors i prototips accessibles remotament. L'objectiu d'aquest treball és conceptualitzar, dissenyar i desenvolupar una aplicació nativa per a iPad que sigui una interfície interactiva de disseny de circuits quàntics. Es vol, a més, que de cada circuit se'n mostrin les figures de mèrit habituals i que els circuits dissenyats es puguin exportar en formats compatibles amb els simuladors i les plataformes més utilitzats del mercat.

Per assolir aquest objectiu, primer es duu a terme un estudi del mercat actual, tant de plataformes com d'aplicacions existents, i, posteriorment, s'opta per un disseny centrat en l'usuari que permeti identificar els elements necessaris per desenvolupar una eina funcional i atractiva tant per a investigadors actius com per a usuaris que vulguin introduir-se en la computació quàntica.

Paraules clau:
Circuits quàntics, desenvolupament per a iOS.

Abstract:
Mobile development is constantly increasing its scope accessing to all the areas of the society. In addition, quantum computing – a discipline that is currently being developed – already has its first remote access prototypes and simulators. The main objective of this project is to conceptualize, to design and to develop a native app for iPad that represents an interactive interface for designing quantum circuits. Furthermore, for each designed circuit, its figures of merit should be shown and the user should be able to export such circuits in formats that are compatible with the most used simulators and platforms of the market. In order to achieve this objective, the market will be studied and the app development will be started with a user-centered design, which will help us develop an attractive and useful tool both for active researchers in the field and for beginners that want to be initiated into quantum circuits design.

Keywords:
Quantum Circuits, iOS Development.

ÍNDEX

ÍNDEX DE FIGURES

ÍNDEX DE TAULES

CAPÍTOL

1

INTRODUCCIÓ

1.1 Context i justificació del treball

1.1.1 Anàlisi de la necessitat

Els dispositius mòbils d'avui en dia disposen d'unes capacitats d'interaccio increïbles, d'un ventall de sensors molt divers i d'unes especificacions tècniques espectaculars. Això ha propiciat que el seu ús vagi molt més enllà de la comunicació a distància i de l'accés a la informació disponible a la xarxa. Tant és així, que el seu abast arriba fins a tots els àmbits de la societat: l'entreteniment, l'educació, la productivitat, la navegació, les relacions socials, la interacció home-màquina, etc. El món de la recerca no n'és una excepció. Actualment, no és estrany veure prototips i muntatges utilitzats per fer recerca que usen, d'una banda, els sensors dels mòbils (càmera, micròfon, GPS, acceleròmetre...) per tal de capturar dades de l'entorn o dels elements estudiats i, de l'altra, les pantalles interactives per mostrar i manipular la informació obtinguda. Aquesta tasca, que abans estava reservada als ordinadors convencionals, cada vegada es troba més present en el món dels dispositius mòbils, entre altres motius, perquè la diferència pel que fa a capacitat computacional i manipulació de gràfics entre els ordinadors i els dispositius mòbils és cada cop menor.

Una altra tendència actual és la d'incorporar cada cop més serveis i funcionalitats al costat del servidor. Des de les compres a través d'Internet fins a l'emmagatzematge al núvol, passant per la interacció amb els assistents personals, la part de processament que es produeix fora dels dispositius mòbils és també creixent. Això està afavorit per diversos motius com, per exemple, la millora en les infraestructures de comunicació i accés a la xarxa (fibra òptica, 5G...), la necessitat d'energia creixent (per realitzar operacions cada cop més complexes) en contraposició a la capacitat de les bateries dels dispositius mòbils, el volum de dades a tractar (bases de dades

massives, *Big Data*...) i la capitalització de les aplicacions disponibles a les botigues d'aplicacions. Aquesta tendència fa que encara que les capacitats dels dispositius mòbils siguin molt grans, aquests sovint actuen bàsicament com a interfícies de serveis i processos que s'executen en computadors remots.

Per la seva banda, la computació quàntica, una branca a mig camí entre física i enginyeria, es troba en les primeres etapes del seu desenvolupament. Aquest nou paradigma computacional es basa en l'aprofitament de les propietats de la mecànica quàntica (superposició, entrellaçament, coherència...) per aconseguir una millora important en la resolució de determinats problemes que són molt difícils de resoldre utilitzant els ordinadors convencionals. Els recursos (en temps i espai de memòria) per realitzar simulacions d'aquest tipus de computació mitjançant ordinadors convencionals creix exponencialment amb el nombre de qubits (bits quàntics) que s'utilitzen. El llindar de 49 qubits s'anomena *supremacia quàntica* i es correspon amb el punt en què el càlcul de l'ordinador quàntic no es pot simular mitjançant ordinadors clàssics (caldria emmagatzemar milers de bilions de nombres complexos només per caracteritzar l'estat inicial del registre de qubits) [13].

Els primers processadors quàntics, uns xips basats en circuits superconductors, ja són una realitat i es troben en fase de transició entre el món de la recerca i el mercat. De fet, des que grans empreses com IBM o Google han començat la carrera per aconseguir l'anomenada *supremacia quàntica*, l'atenció i els recursos invertits en la recerca en aquest àmbit s'han disparat. No obstant això, cal destacar que per tal que aquests xips superconductors presentin les propietats necessàries per realitzar computació quàntica cal aïllar-los completament i abaixar-ne la temperatura fins a uns pocs milikelvin [1]. Per tant, com en el cas dels supercomputadors, l'accés a aquests xips es realitza sempre remotament i, a més, no hi ha indicis que això deixi de ser així a curt o mitjà termini. Per aquest motiu, empreses com Rigetti (amb el *Forest* [2]) o IBM (amb el *Quantum Experience* [3]) ofereixen serveis i plataformes de *cloud* per simular, programar i testejar els seus xips quàntics. Aquesta «programació quàntica» dels xips, com en el cas de la programació clàssica a baix nivell, es pot fer via llenguatges de descripció de hardware, mitjançant instruccions en llenguatges d'assembladors o bé directament amb els esquemes de portes.

Tenint en compte el marc presentat, té sentit plantejar la possibilitat que un dispositiu mòbil com un iPad faci d'interfície per dissenyar i reprogramar els circuits quàntics. Una aplicació que permetés dissenyar els circuits utilitzant portes elementals, realitzar la caracterització dels circuits mitjançant les figures de mèrit (cost quàntic, retard i qubits brossa) i exportar els resultats en formats compatibles amb els simuladors i plataformes més utilitzats del mercat seria una eina molt útil per a la docència, la investigació i l'enginyeria en l'àmbit de la computació quàntica.

1.1.2 Anàlisi de les solucions existents

En revisar el mercat actual d'aquest tipus d'aplicacions s'ha vist que quant a aplicacions mòbils en aquest camp hi ha molt poc desenvolupat i, a més, les aplicacions generalment són més aviat

per fer petits circuits d'exemple [8, 9] que no pas per dissenyar circuits complexos. L'única excepció que s'ha trobat és un compilador quàntic per a iOS que permet executar fitxers de codi mitjançant una plataforma externa [11]. No obstant això, l'enfocament s'allunya de les possibilitats dels dispositius mòbils pel que fa d'interacció i presentació. Cal destacar, a més, que en l'àmbit de disseny i simulació de circuits clàssics sí que hi ha aplicacions amb propòsits similars a la que es planteja en aquest treball com, per exemple, el *Logic Simulator Pro* per a Android [10].

Quant a APIs i plataformes de *cloud*, hi ha opcions molt interessants —principalment, de grans empreses, com el *Quantum Experience* d'IBM [3], el *quantum playground* de Google [5] o el LIQUi|> de Microsoft [6]—, però, que estan clarament orientades a programadors que treballin directament amb codi. En aquest sentit, és més interessant veure aquestes plataformes com a software complementari del que es pretén desenvolupar, ja que permeten, a més de simular amb una potència de càlcul superior a la dels ordinadors convencionals, testejar els circuits en xips reprogramables reals. Així, tal com s'ha comentat, l'aplicació a desenvolupar a la llarga es comportaria com una interfície interactiva d'aquestes plataformes.

Pel que fa al disseny, i tot i que hi ha excepcions, les interfícies de treball que s'han vist estan poc cuidades i/o no són gaire adequades per fer presentacions, especialment, les que no disposen d'editor gràfic. Donat que aquest tipus d'eines han de servir també per a la divulgació i la representació dels circuits/esquemes, un aspecte a millorar respecte de les opcions disponibles és la representació clara i fàcil d'interpretar dels dissenys.

Finalment, un detall rellevant i, en certa manera, sorprenent és que no hi ha cap aplicació que calculi les figures de mèrit dels circuits o codis generats (retard, cost quàntic, qubits brossa, etc.). Aquest també ha de ser un dels aspectes a treballar, ja que les figures de mèrit són un element clau per a la recerca dins de l'ambit.

1.2 Objectius del treball

Tenint en compte el context descrit a l'apartat anterior, l'objectiu principal d'aquest treball és desenvolupar una aplicació nativa per a iPad (i, per tant, exclusiva per a iOS [15]) senzilla, intuïtiva i interactiva per dissenyar, simular i visualitzar els circuits i sistemes quàntics basats en portes quàntiques elementals. La idea és permetre que l'usuari dissenyi els circuits a qualsevol escala (des de circuits senzills d'uns pocs qubits fins a arquitectures completes de processadors), amb una interfície gràfica senzilla i visual (que aprofiti les capacitats d'interacció tàctil d'aquests dispositius) i que permeti exportar els dissenys en forma de llista d'instruccions de codi assemblador per tal d'executar els dissenys en simuladors i/o eines de test.

Per dur a terme aquest desenvolupament s'han d'assolir un conjunt d'objectius secundaris, que, de fet, tenen sentit per si mateixos i que es descriuen a continuació:

- **Anàlisi de mercat (estat de l'art)**: això inclou, d'una banda, la revisió i l'anàlisi de

les aplicacions i plataformes que solucionen part de la problemàtica a resoldre i, de l'altra, la identificació de les plataformes i eines complementàries més importants que es poden utilitzar per simular i/o testejar els dissenys.

- **Disseny de l'aplicació**: es pretén realitzar una aplicació interactiva, visual i intuïtiva que permeti dur a terme el disseny i exportació dels circuits de la manera més senzilla i elegant possible. Això inclou una petita anàlisi del públic objectiu i una preparació dels *mockups* i els *wireframes* de l'aplicació adaptats a aquest públic.

- **Desenvolupament de la vista principal**: l'objectiu és resoldre la part principal de l'aplicació, és a dir, la vista de disseny de circuits des de la qual es poden arrossegar les portes quàntiques per tal de fer els esquemes.

- **Generació de fitxers de codi**: desenvolupar l'algorisme que tradueix els esquemes de portes a fitxers de codi QASM que es puguin executar en una de les plataformes disponibles. Si s'assoleix aquest objectiu per a una de les plataformes/llenguatges, adaptar el codi per fer la traducció a altres plataformes/llenguatges hauria de ser senzill. Es deixa fora de l'abast del projecte la comunicació directa amb el *back-end* de les plataformes.

- **Mètode sistemàtic per caracteritzar els circuits**: descriure un procediment per calcular les figures de mèrit habituals (cost quàntic, retard i qubits brossa) dels circuits dissenyats mitjançant l'aplicació. Aquest objectiu està relacionat amb la necessitat d'avaluar i comparar els circuits dissenyats amb l'aplicació.

- **Implementar persistència**: desenvolupar un sistema que permeti emmagatzemar (carregar) circuits a (de) la memòria de l'iPad i, addicionalment, estudiar la possibilitat d'utilitzar algun tipus de servei d'emmagatzematge extern.

- **Avaluació de l'aplicació**: l'objectiu final és testejar l'aplicació en un dispositiu real. Per fer-ho, cal dissenyar un circuit senzill conegut mitjançant un iPad físic, obtenir-ne les figures de mèrit, generar el fitxer de codi i provar-lo en un plataforma externa.

1.3 Enfocament i mètode seguit

L'objectiu principal del projecte és el desenvolupament de l'aplicació. L'estratègia a seguir per assolir-lo és crear una aplicació nativa des de zero. L'elecció de fer-la nativa està motivada per dues característiques principals: d'una banda, el gran pes que es vol donar a la interacció i al disseny, una tasca que es pot assolir amb més facilitat si s'utilitza un desenvolupament natiu, i, de l'altra, pel fet que es vol realitzar únicament per a iPad (si es volgués fer una aplicació multiplataforma, l'opció d'un desenvolupament híbrid guanyaria més pes). Pel que fa a l'elecció de desenvolupar des de zero, està justificat en el sentit que l'enfocament plantejat (dissenyar els circuits i generar els fitxers de codi per simular) s'allunya del plantejament que presenten les plataformes disponibles de codi obert (escriure els codis i, en tot cas, generar els circuits després). Més encara, per la naturalesa del projecte, té sentit construir tota l'arquitectura de l'aplicació partint de la definició dels elements centrals, és a dir, les portes i els circuits, i que totes les funcionalitats addicionals (persistència, càlcul de figures, traducció, etc.) s'adaptin al

model d'aquests objectes. Això sí, un cop generats els fitxers de codi, sí que té sentit aprofitar desenvolupaments externs com, per exemple, l'SDK d'IBM.

1.4 Planificació del treball

Aquest projecte de desenvolupament té una duració aproximada de 15 setmanes (incloent-hi les tasques de disseny, de desenvolupament i de redacció de la memòria) amb una dedicació de 15 hores per setmana (una tercera part de les quals en caps de setmana o festius).

	Inici	**Final**	**Temps aprox.**	**Dedicació**
PAC1 - Pla de treball	21-02-2018	14-03-2018	3 setmanes	40 h
PAC2 - Disseny	15-03-2018	04-04-2018	3 setmanes	45 h
PAC3 - Implementació	05-04-2018	16-05-2018	6 setmanes	100 h
PAC4 - Entrega final	17-05-2018	06-06-2018	3 setmanes	40 h

Taula 1.1: Taula resum de la planificació.

En els apartats següents, es detallen les tasques i els terminis per assolir els objectius plantejats. Les tasques s'han classificat segons la tipologia: les tasques d'anàlisi (A) s'han marcat en blau, les tasques de disseny (D) en groc, les tasques de programació i desenvolupament (P) en verd, les tasques de test i control de qualitat (T) en rosa i les tasques de redacció dels diferents capítols de la memòria (M) en marró.

1.4.1 Tasques corresponents al pla de treball

S'han definit un total de quatre tasques a realitzar durant la primera etapa del projecte. Es tracta bàsicament de tasques de recerca i planificació.

A.1 - Planificació

En aquesta tasca s'han d'assolir els objectius següents:

- Definir els objectius i determinar abast del treball.
- Segmentar i valorar les tasques a realitzar.

Inici: 21-02-18 **Finalització:** 13-03-18 **Entrega:** PAC1 **Dedicació:** 10 hores

A.2 - Anàlisi del mercat

En aquesta tasca s'han d'assolir els objectius següents:

- Recerca d'aplicacions existents al mercat que cobreixin part de les necessitats.
- Anàlisi dels punts febles i forts.
- Definició de l'estratègia diferenciadora i de les característiques de l'aplicació.

Inici: 21-02-18 **Finalització:** 6-03-18 **Entrega:** PAC1 **Dedicació:** 10 hores

A.3 - Estudi de les plataformes disponibles

En aquesta tasca s'han d'assolir els objectius següents:

- Recerca de plataformes de simulació i/o testeig de circuits quàntics.
- Anàlisi de les capacitats i els formats utilitzats per cada plataforma.
- Valorar la compatibilitat i portabilitat dels formats.
- Seleccionar una plataforma per realitzar els testos finals.

Inici: 28-02-18 **Finalització:** 13-03-18 **Entrega:** PAC1 **Dedicació:** 10 hores

M.1 - Redacció dels capítols 1 i 2 de la memòria

En aquesta tasca s'han d'assolir els objectius següents:

- Preparar la plantilla de la memòria.
- Redactar els dos primers capítols (planificació i mercat).
- Elaborar el diagrama de la planificació.

Inici: 28-02-18 **Finalització:** 13-03-18 **Entrega:** PAC1 **Dedicació:** 10 hores

La distribució temporal de les quatre tasques anteriors es veu reflectida en el diagrama següent:

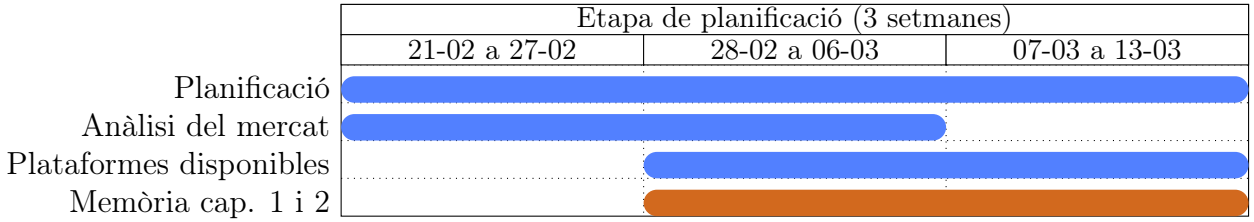

Figura 1.1: Diagrama Gantt de l'etapa de planificació (PAC1).

1.4.2 Tasques corresponents al disseny

S'han definit un total de set tasques a realitzar durant l'etapa de disseny. Es tracta de tasques relacionades amb l'aspecte visual de l'aplicació, la navegació i l'adaptació al públic objectiu. Cal garantir que l'usuari pot utilitzar l'aplicació amb comoditat i que té fàcil accés a les funcions que ha d'utilitzar amb més freqüència.

A.4 - Definició del públic objectiu

En aquesta tasca s'han d'assolir els objectius següents:

- Resumir les característiques dels usuaris principals de l'aplicació (Perfil A).
- Resumir les característiques dels usuaris secundaris de l'aplicació (Perfil B).

Inici: 14-03-18 **Finalització:** 20-03-18 **Entrega:** PAC2 **Dedicació:** 2 hores

A.5 - Avaluació dels casos d'ús

En aquesta tasca s'han d'assolir els objectius següents:

- Definir les històries d'usuari.
- Elaborar el diagrama de casos d'ús.

Inici: 14-03-18 **Finalització:** 20-03-18 **Entrega:** PAC2 **Dedicació:** 5 hores

D.1 - Definició dels fluxos principals de l'aplicació

En aquesta tasca s'han d'assolir els objectius següents:

- Elaborar els diagrames UML dels fluxos principals.

Inici: 14-03-18 **Finalització:** 20-03-18 **Entrega:** PAC2 **Dedicació:** 5 hores

D.2 - Disseny del prototip de l'aplicació en baixa definició

En aquesta tasca s'han d'assolir els objectius següents:

- Definir els elements d'interacció.
- Elaborar els *mockups* de les vistes de l'aplicació.

Inici: 21-03-18 **Finalització:** 27-03-18 **Entrega:** PAC2 **Dedicació:** 8 hores

D.3 - Definició de l'arbre de navegació

En aquesta tasca s'han d'assolir els objectius següents:

- Elaborar l'arbre de navegació de l'aplicació.

Inici: 21-03-18 **Finalització:** 27-03-18 **Entrega:** PAC2 **Dedicació:** 5 hores

D.4 - Disseny de la vista principal en alta definició

En aquesta tasca s'han d'assolir els objectius següents:

- Dissenyar la vista principal de l'aplicació.
- Seleccionar la paleta de colors, la font i els estils de l'aplicació.
- Generar els recursos gràfics necessaris per utilitzar en el projecte XCode.

Inici: 28-03-18 **Finalització:** 03-04-18 **Entrega:** PAC2 **Dedicació:** 10 hores

M.2 - Redacció del capítol 3 de la memòria

En aquesta tasca s'han d'assolir els objectius següents:

- Redactar el tercer capítol de la memòria (disseny).

Inici: 14-03-18 **Finalització:** 03-04-18 **Entrega:** PAC2 **Dedicació:** 10 hores

La distribució temporal de les set tasques anteriors es veu reflectida en el diagrama següent:

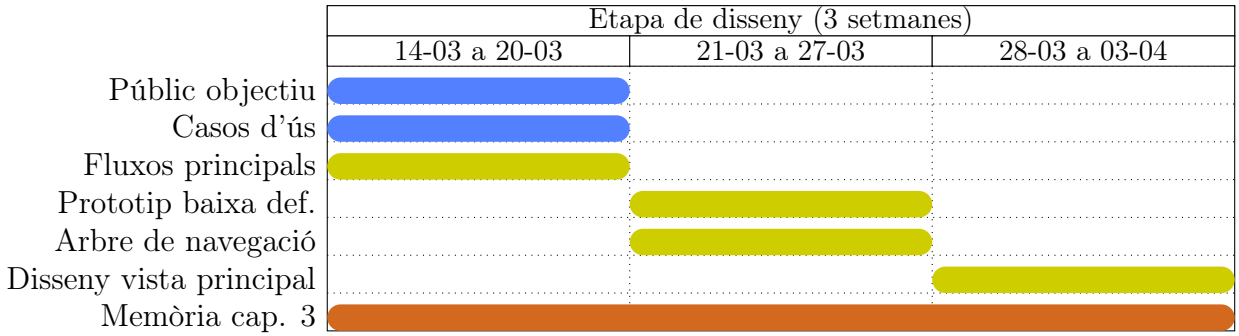

Figura 1.2: Diagrama Gantt de l'etapa de disseny (PAC2).

1.4.3 Tasques corresponents a la implementació

S'han definit un total de nou tasques a realitzar durant l'etapa principal del projecte. Són pròpiament tasques de desenvolupament i definició de l'arquitectura de l'aplicació. Les tasques de test no fan referència als testos inherents al desenvolupament sinó més aviat a proves d'usabilitat de l'aplicació.

P.1 - Desenvolupament del controlador de la vista principal

En aquesta tasca s'han d'assolir els objectius següents:

- Programar l'arquitectura de l'aplicació.
- Desenvolupar el controlador de la vista principal.
- Gestionar la interacció tàctil amb les portes i el llenç (zona de disseny de circuits).

Inici: 04-04-18 **Finalització:** 01-05-18 **Entrega:** PAC3 **Dedicació:** 25 hores

P.2 - Implementació dels models de les portes i els circuits

En aquesta tasca s'han d'assolir els objectius següents:

- Elaborar els models dels objectes més rellevants.
- Programar les classes de les portes elementals i dels circuits.

Inici: 04-04-18 **Finalització:** 24-04-18 **Entrega:** PAC3 **Dedicació:** 15 hores

T.1 - Test d'usabilitat en dispositiu real

En aquesta tasca s'han d'assolir els objectius següents:

- Test de l'aplicació en un dispositiu real (en relació amb la interacció).
- Valoració de l'experiència d'usuari en la tasca de disseny de circuits.

Inici: 02-05-18 **Finalització:** 08-05-18 **Entrega:** PAC3 **Dedicació:** 2 hores

P.3 - Desenvolupament d'un gestor de traduccions

En aquesta tasca s'han d'assolir els objectius següents:

- Programar una classe encarregada de centralitzar i gestionar les conversions.
- Classificar i parametritzar les portes elementals.
- Establir un sistema per emmagatzemar els fitxers de traduccions.
- Implementar el sistema per presentar/emmagatzemar/exportar els resultats.

Inici: 18-04-18 **Finalització:** 01-05-18 **Entrega:** PAC3 **Dedicació:** 20 hores

P.4 - Implementació de la generació de fitxers QASM

En aquesta tasca s'han d'assolir els objectius següents:

- Programar la conversió de diagrames de circuits a fitxers QASM.
- Afegir les definicions QASM de les portes elementals.
- Desenvolupar el mètode per sol·licitar la conversió a QASM.

Inici: 25-04-18 **Finalització:** 08-05-18 **Entrega:** PAC3 **Dedicació:** 12 hores

T.2 - Test del generador de fitxers

En aquesta tasca s'han d'assolir els objectius següents:

- Realitzar proves de generació de fitxers i executar-los en la plataforma corresponent.

Inici: 09-05-18 **Finalització:** 15-05-18 **Entrega:** PAC3 **Dedicació:** 2 hores

A.6 - Descripció d'algorismes per calcular les figures de mèrit

En aquesta tasca s'han d'assolir els objectius següents:

- Definir les figures de mèrit pel cas d'un circuit genèric basat en portes.
- Descriure mètodes recursius per calcular figures de mèrit de circuits compostos.
- Plantejar la possibilitat de realitzar els càlculs en temps real.
- Valorar estratègies d'optimització automàtica dels circuits.

Inici: 25-04-18 **Finalització:** 08-05-18 **Entrega:** PAC3 **Dedicació:** 12 hores

P.5 - Implementació d'algorismes per calcular les figures de mèrit

En aquesta tasca s'han d'assolir els objectius següents:

- Programar els algorismes per calcular les figures de mèrit.
- Desenvolupar el mètode per sol·licitar i presentar els resultats.

Inici: 02-05-18 **Finalització:** 15-05-18 **Entrega:** PAC3 **Dedicació:** 10 hores

M.3 - Redacció dels capítols 4 i 5 de la memòria

En aquesta tasca s'han d'assolir els objectius següents:

- Redactar els capítols quart i cinquè (modelització i desenvolupament).

Inici: 25-04-18 **Finalització:** 15-05-18 **Entrega:** PAC3 **Dedicació:** 10 hores

La distribució temporal de les nou tasques anteriors es veu reflectida en el diagrama següent:

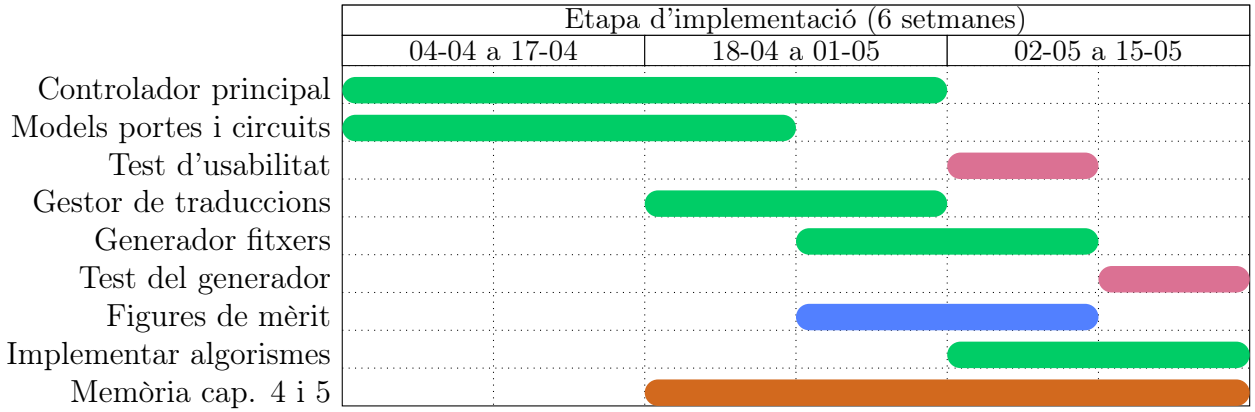

Figura 1.3: Diagrama Gantt de l'etapa d'implementació (PAC3).

1.4.4 Tasques finals

S'han definit un total de cinc tasques a realitzar durant la darrera etapa del projecte. Com que la peristència és una característica secundària pels objectius del treball s'ha deixat per aquesta etapa final, de manera que es dona prioritat al desenvolupament de l'etapa anterior. La resta de tasques són de test i redacció de memòria.

P.6 - Desenvolupament del gestor de persistència de dades

En aquesta tasca s'han d'assolir els objectius següents:

- Programar una classe encarregada de centralitzar i gestionar la persistència.
- Implementar un sistema per carregar/guardar els circuits en la memòria de l'iPad.
- Investigar la possibilitat d'incloure persistència externa dels circuits.

Inici: 16-05-18 **Finalització:** 22-05-18 **Entrega:** PAC4 **Dedicació:** 13 hores

T.3 - Test de la persistència

En aquesta tasca s'han d'assolir els objectius següents:

- Provar de carregar i guardar diferents circuits.

Inici: 30-05-18 **Finalització:** 05-06-18 **Entrega:** PAC4 **Dedicació:** 2 hores

T.4 - Test de l'aplicació en un cas pràctic

En aquesta tasca s'han d'assolir els objectius següents:

- Escollir un circuit senzill per simular-lo.
- Dibuixar el circuit amb l'aplicació.
- Calcular les figures de mèrit.
- Exportar el fitxer en format QASM.
- Simular el codi QASM amb una plataforma externa.

Inici: 30-05-18 **Finalització:** 05-06-18 **Entrega:** PAC4 **Dedicació:** 2 hores

M.4 - Redacció dels capítols 6 i 7 de la memòria

En aquesta tasca s'han d'assolir els objectius següents:

- Redactar el capítol sisè de la memòria (resultat i testeig).
- Redactar les conclusions i l'avaluació crítica del projecte.
- Revisar els capítols anteriors.
- Redactar el resum del projecte.

Inici: 16-05-18 **Finalització:** 05-06-18 **Entrega:** PAC4 **Dedicació:** 10 hores

M.5 - Preparació dels vídeos demostratius

En aquesta tasca s'han d'assolir els objectius següents:

- Realitzar un circuit senzill mitjançant l'aplicació.
- Elaborar un vídeo demostratiu del funcionament de l'aplicació.
- Elaborar el vídeo de presentació del projecte.

Inici: 30-05-18 **Finalització:** 05-06-18 **Entrega:** PAC4 **Dedicació:** 4 hores

La distribució temporal de les cinc tasques anteriors es veu reflectida en el diagrama següent:

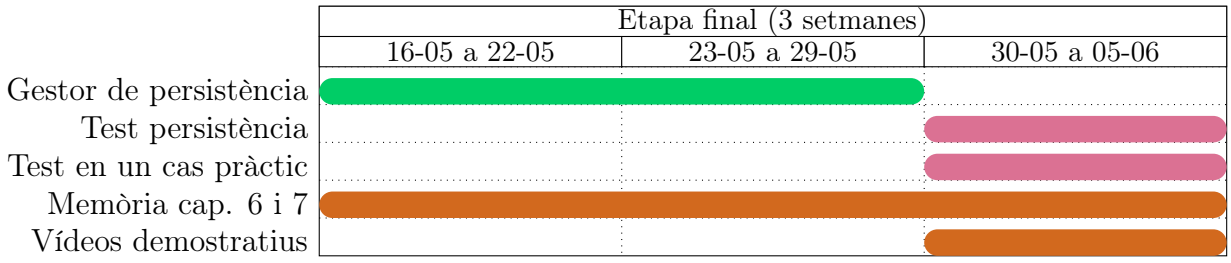

Figura 1.4: Diagrama Gantt de l'etapa final (PAC4).

1.4.5 Diagrama Gantt de la planificació del projecte

El resum de totes les tasques a realitzar es pot veure al diagrama Gantt següent:

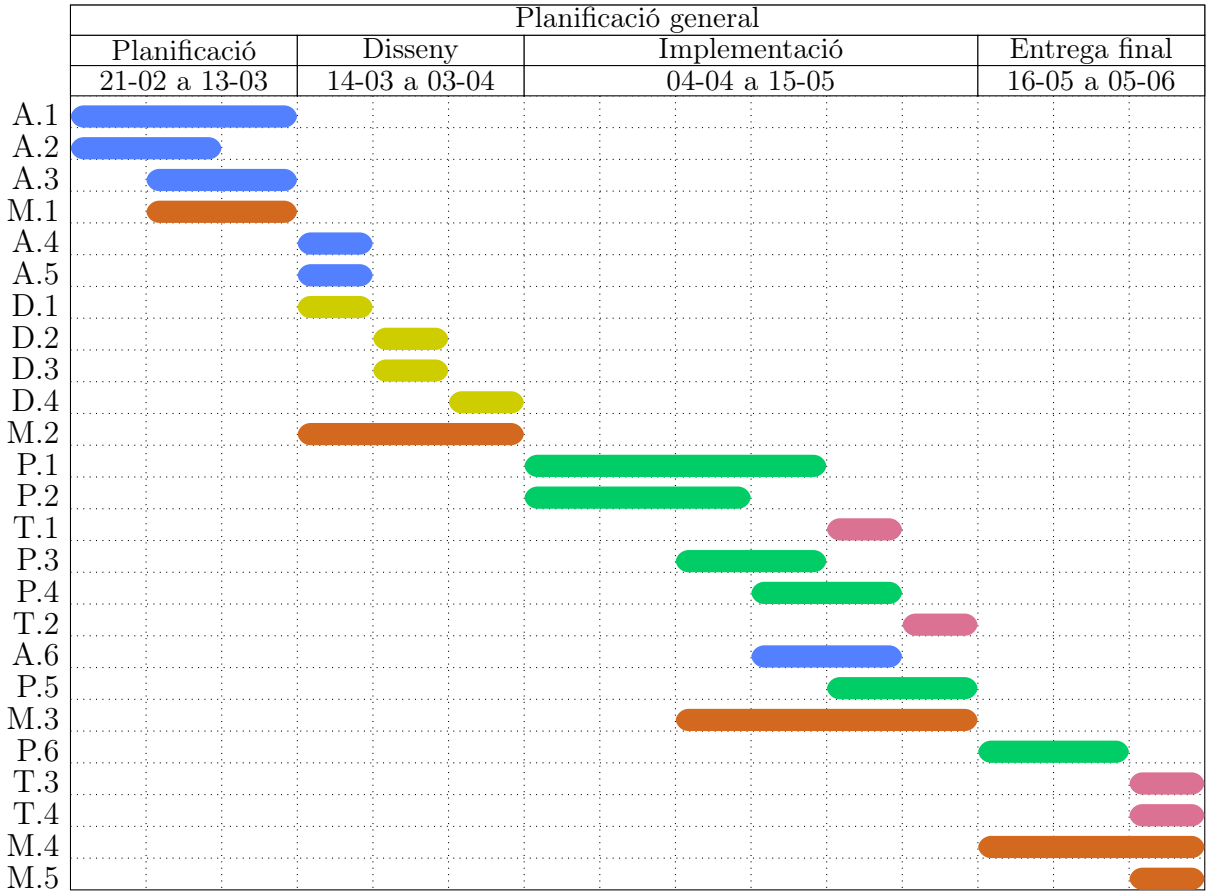

Figura 1.5: Diagrama Gantt del projecte complet.

1.5 Breu sumari de productes obtinguts

A partir de la llista d'objectius es pot extreure la relació de productes que s'han d'obtenir a partir d'aquest treball:

- **Anàlisi de mercat**: informe de les aplicacions disponibles al mercat, inclòs al segon capítol de la memòria.
- **Disseny de l'aplicació**: conjunt de *mockups* i *wireframes* de l'aplicació, disponibles al tercer capítol de la memòria.
- **Desenvolupament de la vista principal**: projecte Xcode amb el codi Swift i breu explicació al cinquè capítol de la memòria.
- **Generació de fitxers de codi**: codi Swift i breu explicació al quart capítol de la memòria.
- **Mètode sistemàtic per caracteritzar els circuits**: codi Swift i descripció del mètode al quart capítol de la memòria.

- **Implementació de la persistència**: codi Swift i breu explicació en el cinquè capítol de la memòria.
- **Avaluació de l'aplicació**: cas d'exemple en el sisè capítol de la memòria i vídeo demostratiu del funcionament.

A continuació, s'inclou la llista amb els entregables del treball:

1. Projecte Xcode amb el codi Swift de l'aplicació.
2. Memòria del projecte.
3. Vídeo demostratiu del funcionament de l'aplicació.

1.6 Breu descripció dels altres capítols de la memòria

La memòria d'aquest projecte es divideix en set capítols, incloent-hi la introducció i les conclusions finals. En el primer capítol s'ha justificat la necessitat de l'aplicació, s'han establert els objectius i s'ha marcat un pla de treball. Els capítols centrals (del 2 al 6) descriuen de manera cronològica les diferents parts del procés de creació de l'aplicació a desenvolupar fent especial èmfasi en tot allò que és singular i innovador respecte d'altres desenvolupaments similars. A continuació, es descriuen els continguts dels capítols centrals.

En el segon capítol s'analitzen el públic objectiu, el mercat actual d'aquests tipus d'aplicacions mòbils i les plataformes de simulació disponibles. Una anàlisi exhaustiva de l'estat de l'art és la primera etapa de qualsevol treball de desenvolupament o de recerca. Aquest capítol amplia i categoritza la informació resumida que s'ha presentat en l'apartat de context i justificació del treball. A més, representa el punt de partida per a la següent etapa de procés de desenvolupament: el disseny.

En el tercer capítol es presenta el disseny de l'aplicació adaptat al perfil d'usuari i a les especificacions funcionals obtingudes en l'anàlisi del segon capítol. Això inclou els prototips en baixa definició, els diagrames UML dels fluxos principals d'usuari, l'arbre de navegació i alguns dissenys en alta definició.

En el quart capítol es descriuen els models conceptuals dels objectes a representar (portes i circuits), els mètodes de càlcul de les figures de mèrit i les estratègies emprades pel que fa a l'arquitectura de la solució. Per tant, aquest capítol recull la informació teòrica necessària per començar el desenvolupament de l'aplicació.

En el cinquè capítol es descriuen els aspectes més rellevants de la implementació de la vista principal i de la persistència de dades. També s'hi inclouen els models de les classes emprades i la relació entre elles.

El sisè capítol recull els resultats del desenvolupament de l'aplicació i els testos realitzats en un cas pràctic concret: la simulació d'una cel·la multiplicadora quàntica.

Finalment, en el darrer capítol s'han inclòs les conclusions del treball realitzat, una anàlisi crítica del projecte i propostes per continuar-hi treballant.

CAPÍTOL

2

ANÀLISI DEL MERCAT

En aquest apartat s'analitzen algunes de les eines i aplicacions existents al mercat que ja cobreixen part o la totalitat de les funcionalitats/característiques de la proposta d'aplicació que es pretén desenvolupar. Per a cada eina, s'analitzen els punts forts i febles per tal d'obtenir-ne una estretègia diferenciadora.

Tot i això, algunes d'aquestes eines són, de fet, plataformes complementàries a l'aplicació que es vol desenvolupar i, per tant, es tindran en compte a l'hora de programar l'aplicació (per exemple, a l'hora d'escollir els formats d'exportació dels dissenys).

2.1 Plataformes

2.1.1 IBM Q Experience

IBM disposa d'una plataforma en línia per simular i testejar circuits quàntics [3]. Aquesta plataforma permet, entre altres coses, dissenyar petits circuits que es poden simular en els servidors d'IBM o bé executar directament en un xip quàntic de 5 o 16 qubits basat en tecnologies superconductores (figura 2.1).

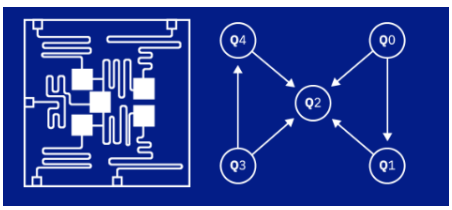

Figura 2.1: Xip quàntic d'IBM.

Aquesta plataforma presenta dues modalitats per dissenyar els circuits: d'una banda, l'editor gràfic, el qual permet dissenyar circuits senzills arrossegant les portes disponibles a les línies de registre (figura 2.2), i, de l'altra, l'editor de codi QASM (*quantum assembly language*), que permet descriure la successió de portes quàntiques sobre el registre mitjançant un llenguatge d'assemblador (figura 2.3).

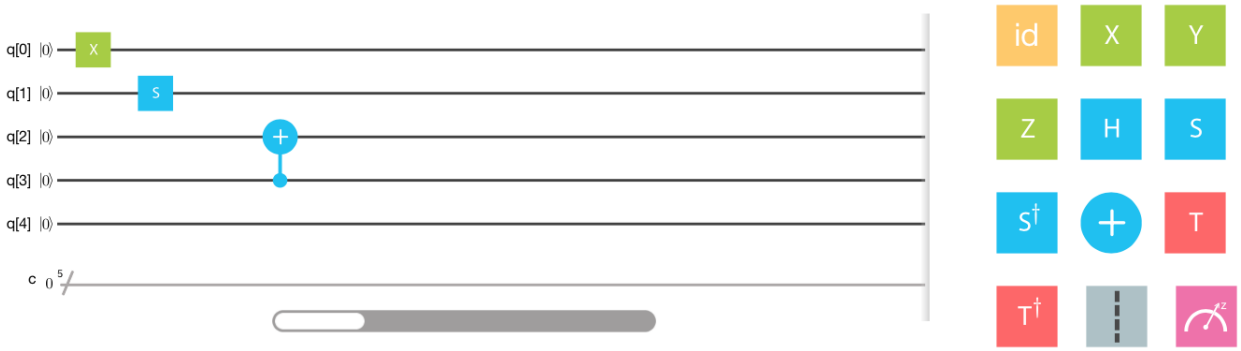

Figura 2.2: Editor gràfic de la plataforma d'IBM.

```
1  include "qelib1.inc";
2  qreg q[5];
3  creg c[5];
4
5
```

Figura 2.3: Editor QASM de la plataforma d'IBM.

A més, aquesta plataforma és accessible remotament mitjançant el QISKit[4] un *kit* de desenvolupament lliure desenvolupat per IBM amb el suport d'altres col·laboradors.

Punts forts:

- Permet testejar els circuits en un xip real de 5 o 16 qubits.
- És una eina senzilla d'utilitzar (intuïtiva).
- Es tracta d'una eina gratuïta i multiplataforma (accessible en línia).
- Inclou material i eines addicionals: tutorials, fòrum, exemples...

Punts febles:

- L'editor gràfic està molt limitat.
- No es permet el disseny modular per blocs.
- No explota les capacitats d'interacció tàctil dels dispositius mòbils.

2.1.2 Quantum Playground de Google

Google disposa d'una plataforma en línia per simular circuits quàntics de fins a 22 qubits mitjançant scripts (QScripts) [5]. La solució es mostra mitjançant un histograma (figura 2.4) que representa la probabilitat de cada estat possible del registre a la sortida.

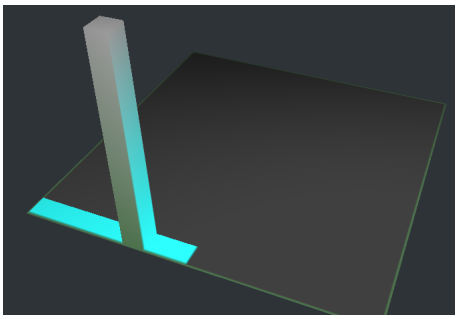

Figura 2.4: Resultat de l'algorisme de Grover al Quantum Playground.

Punts forts:

- Es pot executar el codi pas a pas per veure el balanç de probabilitat de cada iteració.
- Conté exemples de *scripts* dels circuits més importants.
- És una eina gratuïta i multiplataforma (accessible en línia).

Punts febles:

- No disposa d'editor gràfic.
- No explota les capacitats d'interacció tàctil dels dispositius mòbils.

2.1.3 LIQUi|> de Microsoft

El LIQUi|> de Microsoft és una arquitectura de software lliure i multiplataforma per a computació quàntica desenvolupada per l'equip QuArC (Quantum Architectures and Computation Group). Aquesta eina permet simular algorismes quàntics, optimitzar sistemes quàntics o traduir instruccions d'alt nivell escrites en #F a instruccions màquina, entre d'altres [7].

```
let CNOT (qs:Qubits) =
    let gate =
        Gate.Build("CNOT",fun () ->
            new Gate(
                Name    = "CNOT",
                Help    = "Controlled NOT",
                Mat     = (CSMat(4,[(0,0,1.,0.); (1,1,1.,0.);
                                    (2,3,1.,0.); (3,2,1.,0.)])),
                Draw    = "\\ctrl{#1}\\go[#1]\\targ"
        ))
    gate.Run qs
```

Figura 2.5: Exemple de porta definida utilitzant LIQUi|>.

A més, permet treballar amb circuits quàntics de fins a 30 qubits i inclou mecanismes per exportar les llistes d'instruccions a esquemes circuitals (és a dir, el pas invers de l'aplicació que es pretén desenvolupar).

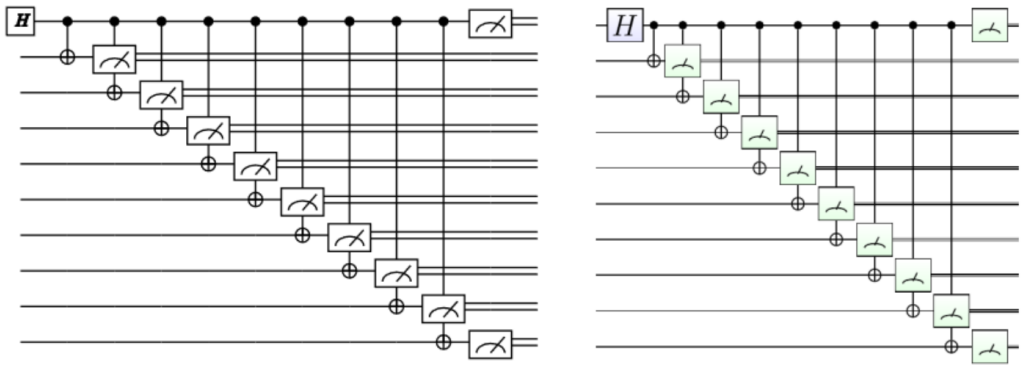

Figura 2.6: Exemples de circuits generats amb LIQUi|>.

Punts forts:

- És una eina molt completa i versàtil.
- Gratuïta i multiplataforma.
- Permet programar en alt nivell amb #F.

Punts febles:

- Calen ferms coneixements de programació.
- No disposa d'editor gràfic.
- No explota les capacitats d'interacció tàctil dels dispositius mòbils.

2.1.4 Forest de Rigetti

Es tracta d'una API de computació quàntica al núvol mitjançant el llenguatge d'instruccions Quil i el pyQuil, una llibreria de Python per executar programes quàntics (figura 2.7).

```
1  from pyquil.quil import Program
2  from pyquil.gates import H, CNOT
3  from pyquil.api import SyncConnection
4  # construct a Bell State program
5  p = Program()
6  p.inst(H(0))
7  p.inst(CNOT(0, 1))
8  # run the program on a QVM
9  qvm = SyncConnection()
10 result = qvm.wavefunction(p)
```

Figura 2.7: Exemple de programa escrit en pyQuil.

El Forest és, de fet, una plataforma en línia per simular i testejar circuits quàntics de fins a 26 qubits mitjançant un entorn de test virtual (Quantum Virtual Machine) [2]. A més, com en el cas d'IBM, aquesta plataforma permet dissenyar circuits que es poden executar directament en un xip quàntic real (en aquest cas, de 19 qubits) basat en tecnologies superconductores.

Punts forts:

- Permet testejar els circuits en un xip real de 19 qubits (amb força limitacions).
- És una eina gratuïta i multiplataforma (accessible en línia).
- Disposa d'un entorn de desenvolupament complet.
- Integra el paradigma de processador híbrid (clàssic-quàntic).

Punts febles:

- Calen coneixements avançats de programació.
- És una eina d'accés limitat.
- No disposa d'editor gràfic.
- No explota les capacitats d'interacció tàctil dels dispositius mòbils.

2.2 Aplicacions

2.2.1 Quirk

Aplicació web que és, de fet, un simulador de navegador, interactiu i en temps real de circuits quàntics amb registres de fins a 16 qubits [12]. A la figura 2.8 es mostra l'aspecte del simulador i les diferents accions que es poden aplicar sobre les línies de registre.

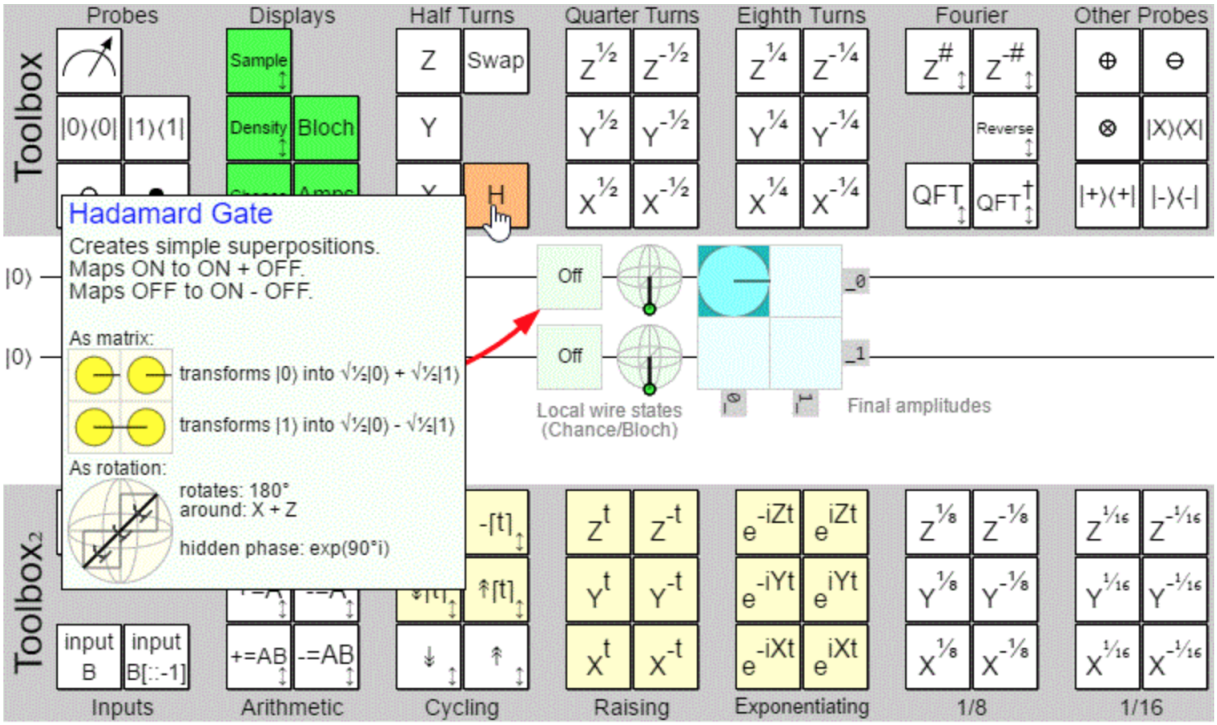

Figura 2.8: Simulador Quirk en linia.

Punts forts:

- Eina visual i interactiva (de tipus *drag-and-drop*).
- Eina gratuïta i multiplataforma (accessible en línia).
- Permet el disseny modular per blocs i la creació de portes personalitzades.

Punts febles:

- Capacitat de simulació baixa.
- Opcions d'exportació limitades (no permet exportar els dissenys en llenguatges de descripció de hardware).
- Disseny poc cuidat i amb baixa adaptació als dispositius mòbils.

2.2.2 Quantum Circuit Simulator

Aplicació per a Android de simulació de circuits quàntics. Es tracta d'una eina molt limitada, però que permet fer ràpidament alguns exemples senzills de circuits i simular-los directament amb el mòbil.

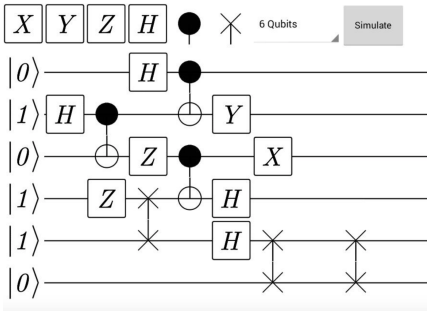

Figura 2.9: Simulador de circuits quàntics disponible per a Android.

Punts forts:

- Eina visual i interactiva.
- Eina intuïtiva i fàcil d'utilitzar

Punts febles:

- Molt limitada en tots els aspectes.
- Disseny poc cuidat.

2.2.3 Quantum Compiler

Es tracta d'un compilador de programes quàntics escrits amb Python que utilitza l'SDK QISKit per compilar, executar i mostrar els resultats mitjançant la plataforma d'IBM. Aquesta aplicació és la que més s'ajusta als objectius del projecte. De fet, molts dels aspectes que s'han valorat a l'hora de seleccionar les característiques que ha de tenir l'aplicació es troben d'una manera o altra en l'aplicació (ús dels dispositius mòbils, simulació en remot, persistència al dispositiu,

resultats visuals...). No obstant això, la descripció dels circuits se segueix fent per codi, és a dir, sense editor gràfic de circuits.

File Run

```
qp = QuantumProgram()

# Change as required
numQubits=numBits=2

# Create registers and circuit
qr = qp.create_quantum_register('qr', numQubits)
cr = qp.create_classical_register('cr', numBits)
qc = qp.create_circuit('my_circuit', [qr], [cr])

# Put first qubit in superposition and entangle both qubits
qc.h(qr[0])
qc.cx(qr[0], qr[1])

# Measure qubits
qc.measure(qr[0], cr[0])
qc.measure(qr[1], cr[1])

# Run and get results
result = qp.execute('my_circuit', shots=1024)
stats = result.get_counts('my_circuit')
print(stats)
sandbox_show(plot_histogram(stats))
```

Figura 2.10: Compilador quàntic per a iOS.

Els circuits es poden generar a partir del codi. Per tant, en certa manera fa el procés invers al de l'aplicació que es vol desenvolupar en aquest projecte.

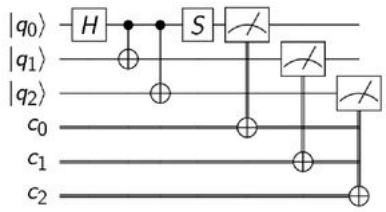

Figura 2.11: Circuit generat mitjançant el *Quantum Compiler*.

Punts forts:

- Adaptació a dispositius mòbils (teclat personalitzat, disseny amb *tabbar*, etc.).
- Simulació en remot per aprofitar la potència dels servidors d'IBM.
- Inclou mecanismes de persistència.
- Permet generar els circuits a partir del codi.

Punts febles:

- No presenta editor gràfic.
- Té limitacions heretades de la plataforma de simulació (nombre de bits, temps de simulació, etc.).
- Molt limitada si no hi ha connexió a Internet.

2.3 Estratègia diferenciadora

Després de revisar i avaluar les diferents alternatives que hi ha al mercat se'n poden extreure les següents conclusions sobre els aspectes a millorar:

En primer lloc, cal destacar que el públic objectiu al qual es dirigeixen condiciona clarament les solucions que han proporcionat. En el cas de les plataformes és clar que van dirigides principalment a usuaris amb ferms coneixements de programació, és a dir, bàsicament, a enginyers informàtics. En el cas de les aplicacions mòbils, en canvi, sembla que els perfils són més variats i s'acosten més al de les disciplines afins (electrònica, física, matemàtiques...) però no són prou útils per a la recerca en aquest camp.

En general, el disseny i l'aspecte visual de les eines que hi ha disponibles no és gaire bo. Sovint es treballa directament amb codi (a vegades, amb el terminal) i en els casos que hi ha editor gràfic, els botons i la resta d'elements visuals amb què l'usuari interactua no presenten un aspecte elaborat o modern. Aquesta tendència és probablement conseqüència de la poca prioritat que se li dona al disseny en aquest àmbit. No obstant això, tot i que és totalment comprensible prioritzar la funcionalitat i/o la capacitat computacional de l'eina davant de l'aspecte quan es programen els circuits, a l'hora de presentar els resultats l'aspecte visual i la claredat dels dissenys pren protagonisme. A més, un bon disseny millora l'experiència d'usuari, la corba d'aprenentatge i l'eficiència en l'ús de l'aplicació.

Un altre aspecte negatiu de les eines que s'han analitzat és el desaprofitament de les capacitats d'interacció dels dispositius mòbils. Més encara, en alguns casos, l'ús a través de mòbil o la tauleta resulta impracticable. De nou, tot i que no se li sol donar importància dins de l'àmbit, un disseny interactiu és essencial per millorar la productivitat dels usuaris (i més, si es té en consideració l'ús creixent dels dispositius mòbils en la recerca).

Pel que fa al mètode de treball, en la major part de les eines es considera que la descripció dels circuits s'ha de fer mitjançant codi (i després es generen els circuits). Aquest punt de vista és vàlid però no representa l'única manera de dissenyar els circuits basats en portes. De fet, en l'àmbit del disseny de processadors és habitual treballar indistintament amb llenguatges de descripció de hardware i amb editors esquemàtics dels circuits (mitjançant els símbols dels elements i les connexions entre ells). El disseny esquemàtic és més senzill d'entendre i dona una visió més adequada de la implementació pràctica del circuit.

S'ha trobat a faltar l'avaluació de les figures de mèrit habituals en totes les eines disponibles. Algunes de les eines donen informació molt interessant com el temps de coherència, la probabilitat d'error del sistema o els histogrames amb les probabilitats dels estats, però no s'ha trobat cap d'elles que doni els valors del retard, el cost quàntic o el nombre de qubits brossa.

Finalment, pel que fa a les plataformes disponibles de simulació i test, el fet que estiguin recolzades per grans empreses i que permeten una comunicació amb hardware real les converteixen en un recurs que val la pena aprofitar.

Tenint en compte els punts anteriors, i tal com s'ha comentat al primer capítol, l'estratègia diferenciadora per abordar la conceptualització i el desenvolupament de l'aplicació es pot resumir en els següents punts:

- Adaptar l'aplicació al públic objectiu.
- Realitzar un disseny senzill i modern de la interfície de l'aplicació.
- Assegurar que la interacció s'ajusta al terminal des del qual s'utilitzarà l'aplicació.
- Generar els fitxers de codi a partir dels esquemes circuitals (i no al revés).
- Capacitat de calcular les figures de mèrit dels circuits.
- Compatibilitat de formats amb les plataformes existents.

CAPÍTOL

3

DISSENY

3.1 Públic objectiu

En un disseny centrat en l'usuari cal identificar i caracteritzar correctament els perfils dels usuaris objectiu, així com les seves motivacions i els escenaris en què utilitzaran l'aplicació. D'aquesta manera, es poden ajustar les funcionalitats, el disseny i la navegació per garantir una navegació fluïda i intuïtiva. Per això s'utilitzarà el mètode Persones [17] amb tres arquetips diferents: en Pau, la Minna i en Dave. Aquests usuaris no es corresponen amb usuaris reals concrets, però modelen adequadament els perfils típics d'usuaris de l'aplicació, és a dir, són representats del públic objectiu.

Aquesta aplicació és una eina que va dirigida principalment a un públic tècnic, especialista en el tema (circuits quàntics) i que presenti unes nocions bàsiques de programació en llenguatges de descripció de *hardware*. Aquest serà el perfil d'en Pau (perfil A). Per definir aquest perfil s'han utilitzat caracterísitques de l'autor del treball (ja que, de fet, serà un dels primers usuaris de l'aplicació) i característiques de companys investigadors (tant del mateix àmbit com d'àmbits propers).

S'ha de preveure, però, la possibilitat que l'aplicació sigui una eina d'introducció a aquest àmbit i, per tant, ha de ser útil per a estudiants, enginyers i graduats de qualsevol de les disciplines afins (física, matemàtiques, electrònica, informàtica, etc.) que vulguin experimentar amb aquest tipus de circuits. Aquest serà el perfil de la Minna (perfil B) i en Dave (perfil C). Per definir aquests perfils s'ha recollit informació d'alguns dels membres de les comunitats que poden estar-hi interessades. Més concretament, de la Societat Catalana de Física [18] i del *think tank* barcelonaqbit [19], ja que són col·lectius que poden mostrar interès en aplicacions com la que es pretén desenvolupar.

3.1.1 Perfil A: Pau Freixa

En Pau Freixa és enginyer en electrònica i fa poc que ha acabat el màster en física computacional. Actualment realitza la tesi en l'àmbit del disseny de circuits reversibles i està molt interessat en els avenços en computació quàntica que s'han produït en els darrers anys. Treballa com a professor associat a temps parcial a la universitat donant classes d'electrònica de primer curs. És un noi jove, entusiasta i proactiu que adora la ciència i la tecnologia des que era ben petit. Els seus pares li van regalar el seu primer iPod quan estava a la universitat i des de llavors s'ha anat comprant diversos dispositius d'Apple. El dispositiu que més utilitza és l'iPad, ja que el fa servir habitualment per veure sèries per Internet i també per les presentacions que fa a la universitat (tant del doctorat com de les classes).

Pau Freixa

Edat: 26 anys.
Professió: estudiant de doctorat (amb beca) i professor associat.
Residència: Barcelona, Espanya.
Estat civil: solter.
Nivell d'ingressos: modestos (< 20.000 €/any).
Aficions/interessos: tecnologia, cinema i videojocs.
Dispositius mòbils: iPhone 6s (iOS 11.0) i iPad Pro 12.9 (iOS 11.2),
Motiu pel qual utilitza iOS/Apple: per influència familiar.
Motiu per utilitzar l'aplicació: recerca i docència.
Coneixements sobre computació quàntica: elevats (ja que estan relacionats directament amb la seva tesi).

En Pau és un representant del conjunt de persones que investiguen sobre el tema i que pretenen utilitzar l'aplicació com una eina de recerca. Aquests usuaris tenen bons coneixements en aquest àmbit i han utilitzat software, aplicacions o plataformes similars per simular circuits quàntics. Per tant, l'aplicació els ha de permetre dissenyar les cel·les i els circuits d'una manera àgil i còmoda. A més, aquest tipus d'usuari valora la capacitat d'exportació dels dissenys en un format compatible amb les plataformes que coneix i que es calculin automàticament els valors de les figures de mèrit dels circuits.

Alguns exemples d'escenaris en què en Pau utilitzarà l'aplicació són els següents:

- Dissenyar un circuit/experiment de violació de les desigualtats de Bell.
- Proposar i comparar diferents alternatives de circuits comparadors d'estats quàntics.
- Comparar propostes de sumadors quàntics de més de 2 qubits.
- Preparar un exemple de circuit que implementi la part de l'emissor o el receptor d'un sistema de criptografía quàntica (protocol BB84).
- Avaluar la viabilitat de simular i/o testejar la cel·la d'un multiplicador quàntic escalable mitjançant el xip quàntic d'IBM.

A continuació, se'n descriu un cas d'ús més concret. En Pau es troba immers en la seva recerca intentant trobar un circuit que generi una superposició de tots els nombres de la forma $2^n P$, amb P un nombre primer. S'ha adonat que això ho pot aconseguir sempre que sigui capaç de generar una superposició de tots els nombres primers. Es fa la pregunta següent: és possible fer una cel·la que, a partir d'un registre inicialitzat a l'estat $|0\rangle$, generi una superposició dels nombres primers? Per a un registre de mida controlada creu que ha de ser força senzill, ja que hi ha relativament pocs nombres que siguin primers. Agafa el qPad Lab i es disposa a fer les primeres proves d'un circuit que generi la superposició dels nombres primers menors que 15 (registre de quatre qubits). Un cop hagi enllestit el circuit, en realitzar una simulació d'un miler d'iteracions a la plataforma d'IBM, espera veure que l'histograma obtingut presenta pics als nombres 2, 3, 5, 7, 11 i 13.

3.1.2 Perfil B: Minna Zimmermann

La Minna Zimmermann és una llicenciada en matemàtiques que treballa com a analista en una empresa alemanya de software. És una persona seriosa i molt responsable que s'ha guanyat la confiança i el respecte dels seus companys i amics. Abans de parlar sobre un tema li agrada documentar-se i revisar-ne l'estat de l'art. Per aquest motiu, tothom té en compte la seva opinió i, fins i tot, algun cop ha col·laborat com a revisora en revistes científiques en els àmbits de la computació distribuïda i el *machine learning*. Està casada amb un empresari del sector del motor i té dos fills adolescents. Utilitza dispositius mòbils de gamma alta i, a més, els renova habitualment.

Minna Zimmermann

Edat: 48 anys.
Professió: analista (software).
Residència: Heidelberg, Alemanya.
Estat civil: casada (amb dos fills).
Nivell d'ingressos: elevats ($>$ 80.000 €/any).
Aficions/interessos: la ciència, l'economia i el benestar.
Dispositius mòbils: iPhone 8 Plus (iOS 11.2) i iPad Pro 9.7 (iOS 11.2).
Motiu pel qual utilitza iOS/Apple: per la seguretat i el *caché*.
Motiu per utilitzar l'aplicació: curiositat/entreteniment.
Coneixements sobre computació quàntica: baixos. Però ha llegit algunes notícies que parlen sobre el tema i n'ha parlat amb persones que hi treballen.

La prioritat per aquest perfil és que l'aplicació sigui intuïtiva i senzilla d'utilitzar. Es tracta de perfils autodidactes que volen l'aplicació per aprendre i experimentar amb aquest tipus de circuits. Com que és una aplicació molt específica, si aquest usuaris se l'han descarregada és perquè han llegit, vist o sentit alguna cosa relacionada amb la matèria. Per tant, han de poder identificar els elements que han vist en les altres fonts (símbols, nomenclatura, esquemes) de manera directa i sense ambigüitats. A més, poden requerir algun tipus de tutorial incial o accés a informació complementària des de la mateixa aplicació.

Alguns exemples d'escenaris en què la Minna podria utilitzar l'aplicació són els següents:

- Experimentar amb el disseny de circuits quàntics sense cap objectiu concret.
- Dissenyar alguns exemples d'algorismes quàntics famosos que ha trobat a la literatura: Grover, Bernstein...
- Preparar estats estranys (entrellaçats, en superposició, etc.) i provar-los en un simulador o un xip real disponible en línia.
- Familiaritzar-se amb les portes bàsiques elementals.

A continuació, se'n descriu un cas d'ús més concret. Avui la Minna, parlant amb un client sobre un experiment de cotxes autònoms en què han participat, ha sentit que Google ha adquirit un nou model de DWave (un ordinador quàntic d'ús específic). Un company de feina ha comentat que la computació quàntica adiabàtica no permet resoldre problemes de caire general. Quan la Minna ha arribat a casa, ha buscat informació sobre computació quàntica a Internet i ha topat amb una aplicació per a iPad amb uns esquemes que li recorden els circuits digitals que coneix. Ha fet un petit circuit entrellaçador i ha vist que el resultat obtingut al simulador és molt contraintuïtiu. Demà ho ensenyarà als companys i seguirà documentant-se.

3.1.3 Perfil C: Dave Thomas

En Dave Thomas és professor de física i astronomia a l'escola de ciència del MIT (MIT School of Science) des de fa més de 10 anys. És un home molt creatiu a qui li agrada innovar i experimentar amb noves metodologies docents. El seu departament està involucrat en diversos projectes de recerca i col·labora amb empreses de disseny i fabricació de peces per a satèl·lits i radiotelescopis. És un gran jugador d'escacs i li encanten els jocs d'enginy en general, una afició que comparteix amb el seu fill de 22 anys. De fet, tenen una col·lecció de més de 1000 trencaclosques de la família del cub de Rubik. Utilitza l'iPad habitualment per llegir articles, per a les classes a la universitat i per resoldre sudokus interactius.

Dave Thomas

Edat: 56 anys.
Professió: professor universitari (física i astronomia).
Residència: Cambridge, Massachusetts (USA).
Estat civil: casat (amb un fill).
Nivell d'ingressos: elevats ($>$ 100.000 \$/any).
Aficions/interessos: la ciència i els jocs d'enginy.
Dispositius mòbils: iPhone SE (iOS 11.2) i iPad edició 2018 (iOS 11.2).
Motiu pel qual utilitza iOS/Apple: és la marca més popular a Estats Units i, a més, va tenir una rebaixa important pel fet de pertànyer al sector educatiu.
Motiu per utilitzar l'aplicació: complement per a les classes que imparteix a la universitat (teoria de la informació quàntica).
Coneixements sobre computació quàntica: elevats. Imparteix classes sobre computació quàntica.

En Dave Thomas és un representant del col·lectiu docent que vol utilitzar l'aplicació com una eina interactiva complementària per a les explicacions teòriques sobre el tema. Aquesta aplicació li ha de permetre explicar els conceptes de registre i operador quàntic, definir les figures de mèrit, introduir els llenguatges de descripció de *hardware* quàntic, etc. A més, ha de ser una aplicació senzilla d'utilitzar perquè els alumnes que vulguin aprendre i experimentar amb ella ho puguin fer sense problemes.

Alguns exemples d'escenaris en què el senyor Thomas podria utilitzar l'aplicació són:

- Aprendre i/o ensenyar el paral·lelisme entre els circuits lògics clàssics i els circuits quàntics basats en portes d'una manera visual i dinàmica.
- Explicar les portes bàsiques elementals i el seu comportament sobre diferents estats quàntics d'entrada.
- Preparar problemes de simplificacions de circuits per resoldre a classe.
- Preparar exemples per explicar fenòmens genuïnament quàntics que s'utilitzen en aplicacions pràctiques.
- Motivar els alumnes a provar les plataformes de simulació disponibles com, per exemple, el *Quantum Experience* d'IBM.
- Presentar esquemes circuitals rellevants i veure'n la traducció a instruccions de codi màquina en QASM.

A continuació, tal com s'ha fet en els dos perfils anteriors, es descriu un cas d'ús més específic. Aquesta setmana, a l'assignatura que imparteix el senyor Thomas, toca començar el tema dedicat a la teleportació quàntica d'informació i presentar els protocols BB84 i Ekert91. Abans, però, el senyor Thomas vol realitzar un exemple pràctic per introduir el tema a classe mitjançant una simualció d'una comunicació quàntica de 10 bits. Com que disposa del qPad Lab, pot preparar quatre cel·les opaques que representen les configuracions que poden tenir els qubits de l'emissor i dues cel·les etiquetades (el contingut de les quals mostra als alumnes) que representen les accions que es poden fer al receptor. Per a cada configuració, genera un *script* que pot córrer al simulador de Google i així demostrar que emissor i receptor comparteixen els mateixos bits quan es compleixen unes determinades condicions. Amb un petit circuit pot simular l'acció d'un intrús que vol accedir a la informació que viatja pel canal i, en aquest cas, el receptor ho pot detectar. Ha aconseguit explicar el protocol BB84 sense fer servir equacions.

Per acabar, tot i que aquests perfils s'ajusten als possibles usuaris reals de l'aplicació, no s'han d'oblidar els usuaris esporàdics que descarreguen l'aplicació per simple curiositat. Per a aquests usuaris (i també, en menor mesura, per a la Minna i els alumnes d'en Dave) pot ser interessant que l'aplicació final disposi d'un tutorial inicial sobre comuptació quàntica que permeti als usuaris posar-se en context, i que en presenti la funció principal (disseny de circuits quàntics) i en descrigui les zones de treball (cal recordar que es tracta d'una aplicació molt tècnica i específica).

3.2 Històries d'usuari

Un cop establerts els perfils dels usuaris i plantejats alguns escenaris d'ús de l'aplicació es poden definir les històries d'usuari (US, de l'anglès *user story*) a les quals s'ha de donar solució. Aquestes històries es corresponen amb tasques o accions específiques que els usuaris volen dur a terme mitjançant l'aplicació per tal d'assolir els seus objectius. A partir d'aquestes històries es poden obtenir tant les subtasques de desenvolupament principals que cal implementar com les vistes necessàries perquè l'usuari pugui realitzar les accions.

3.2.1 Dissenyar un circuit

Com a usuari, vull poder dissenyar un circuit quàntic mitjançant un editor gràfic interactiu. Vull que l'aplicació em permeti arrossegar els símbols de les portes i interconectar-los d'una manera senzilla i eficient.

Aquesta és la funció principal i, per tant, es pot considerar el nucli del disseny i del desenvolupament de l'aplicació. La major part de les funcionalitats de l'aplicació formen part o estan estretament relacionades amb aquesta funció. Per dissenyar un circuit cal un espai de treball al qual poder afegir, modificar i eliminar portes i/o cel·les. Això implica, d'una banda, la necessitat d'una vista específica per a aquesta funcionalitat i, de l'altra, un conjunt de subtasques que s'han de desenvolupar.

3.2.2 Dissenyar una cel·la

Com a usuari, vull poder dissenyar una cel·la quàntica mitjançant un editor gràfic interactiu. Vull que l'aplicació em permeti arrossegar els símbols de les portes elementals a les línies de registre per tal de generar cel·les reaprofitables.

Es tracta d'un cas particular del disseny de circuits i es diferencia per dos aspectes fonamentals. D'una banda, el disseny d'una cel·la només podrà incloure les portes elementals i, de l'altra, aquestes portes només es podran connectar a les línies de registre horitzontals (com succeeix en la major part de les eines existents). A més, cal afegir que l'objectiu principal per fer una cel·la és generar «peces» per al disseny d'un circuit més complex i, per tant, la manera de guardar i carregar les cel·les també és una funció rellevant per a l'usuari.

3.2.3 Avaluar un circuit

Com a usuari, vull poder avaluar les característiques d'un circuit en termes de les figures de mèrit habituals.

Donat un disseny de circuit, cal poder obtenir-ne informació rellevant per caracteritzar-lo i avaluar-ne la viabilitat de simular-lo, testejar-lo i/o implementar-lo físicament. Com que un circuit està compost per portes i cel·les, aquesta tasca inclou l'avaluació d'una cel·la i, en última instància, d'una porta elemental.

3.2.4 Avaluar una cel·la

Com a usuari, vull poder avaluar les característiques d'una cel·la en termes de les figures de mèrit habituals.

Aquesta història és paral·lela a l'anterior. En aquest cas, però, les figures de mèrit són també útils per escollir les «peces» més adients en la tasca de disseny de circuits. Per aquest motiu pot ser interessant mostrar aquesta informació no només des de la vista de disseny de cel·les, sinó també en una vista que actuï com a biblioteca de cel·les.

3.2.5 Simular un circuit

Com a usuari, vull poder simular un circuit quàntic mitjançant una plataforma externa.

Aquesta funcionalitat no forma part de l'aplicació i s'ha de realitzar mitjançant una plataforma externa. No obstant això, per poder assolir aquest objectiu és necessari poder exportar els dissenys en un format compatible amb la plataforma de simulació i indicar-ho d'alguna manera a l'usuari.

3.2.6 Testejar un circuit

Com a usuari, vull poder testejar un circuit quàntic mitjançant el hardware d'una plataforma externa.

De nou, aquesta funcionalitat no forma part de l'aplicació i s'ha de realitzar mitjançant una plataforma externa. En aquest cas, a més, cal que la plataforma disposi del hardware necessari per poder executar el codi i això no és trivial, ja que moltes vegades hi ha restriccions geomètriques o de capacitat d'entrellaçament del sistema físic.

Tot i que seria una opció molt interesant, queda totalement fora de l'abast del projecte l'avaluació automàtica del disseny per determinar si és possible el test en una determinada plataforma i/o l'opció d'adaptar el disseny per forçar-ne la compatibilitat.

3.2.7 Exportar un fitxer de codi

Com a usuari, vull poder generar un fitxer de codi d'assemblador a partir d'un disseny de portes. Vull poder utilitzar els resultats dels meus dissenys en altres plataformes.

Tal com s'ha comentat, per a la simulació i el testeig dels dissenys cal convertir els esquemes circuitals en una llista d'instruccions en codi d'assemblador. Això implica la necessitat d'una vista d'exportació de codi i el desenvolupament d'un sistema gestor de conversió de formats. Aquesta US presenta també una forta dependència respecte de les plataformes de simulació i testeig disponible i requereix una anàlisi i una selecció dels formats més utilitzats.

3.2.8 Carregar una cel·la

Com a usuari, vull poder carregar una cel·la que he creat per utilitzar-la en el disseny d'un circuit.

Aquesta tasca està relacionada amb l'edició de la llista de «peces» disponibles per dissenyar el circuit i no pas amb la modificació d'una cel·la, que estaria inclòs dins de les tasques *dissenyar una cel·la* i *guardar una cel·la*. Aquesta US reafirma la necessitat d'una vista de tipus biblioteca de cel·les.

3.2.9 Guardar una cel·la

Com a usuari, vull poder guardar els canvis d'una cel·la.

Es tracta d'una funcionalitat relacionada tant amb el disseny de cel·les com amb el de circuits (generació d'una «peça»). Així, d'una banda, es voldrà assegurar la persistència de la cel·la en curs i, de l'altra, indicar una sèrie de paràmetres (nom, autor, entrades...) per identificar-la de cara a dissenys més complexos.

3.2.10 Guardar un circuit

Com a usuari, vull poder guardar els canvis del disseny actual.

Aquesta història està relacionada amb la persistència dels dissenys de circuits realitzats. Cal, doncs, una vista per mostrar els dissenys guardats i un mecanisme per carregar un disseny prèviament guardat.

3.3 Diagrames de casos

Un cop definides les històries d'usuari principals es poden representar les accions que realitzen els usuaris actors, mitjançant diagrames UML de casos. A més de les tasques que s'han descrit en l'apartat anterior i que estan directament relacionades amb allò que els usuaris volen aconseguir, s'han inclòs algunes funcions més que indirectament han de fer els usuaris (o el software) per tal d'assolir els objectius de les US. Aquests diagrames inclouen, també, les relacions de dependència entre funcionalitats o tasques. D'una banda, les inclusions, que indiquen que una tasca en requereix una altra per poder-se completar, i, de l'altra, les extensions, que indiquen que es tracta de casos particulars de la tasca pare.

3.3.1 Diagrama general

En el diagrama que es mostra en la figura 3.1 s'han representat les accions principals que l'usuari realitza sobre l'aplicació, així com les que realitza sobre les plataformes externes gràcies a l'ús que fa de l'aplicació.

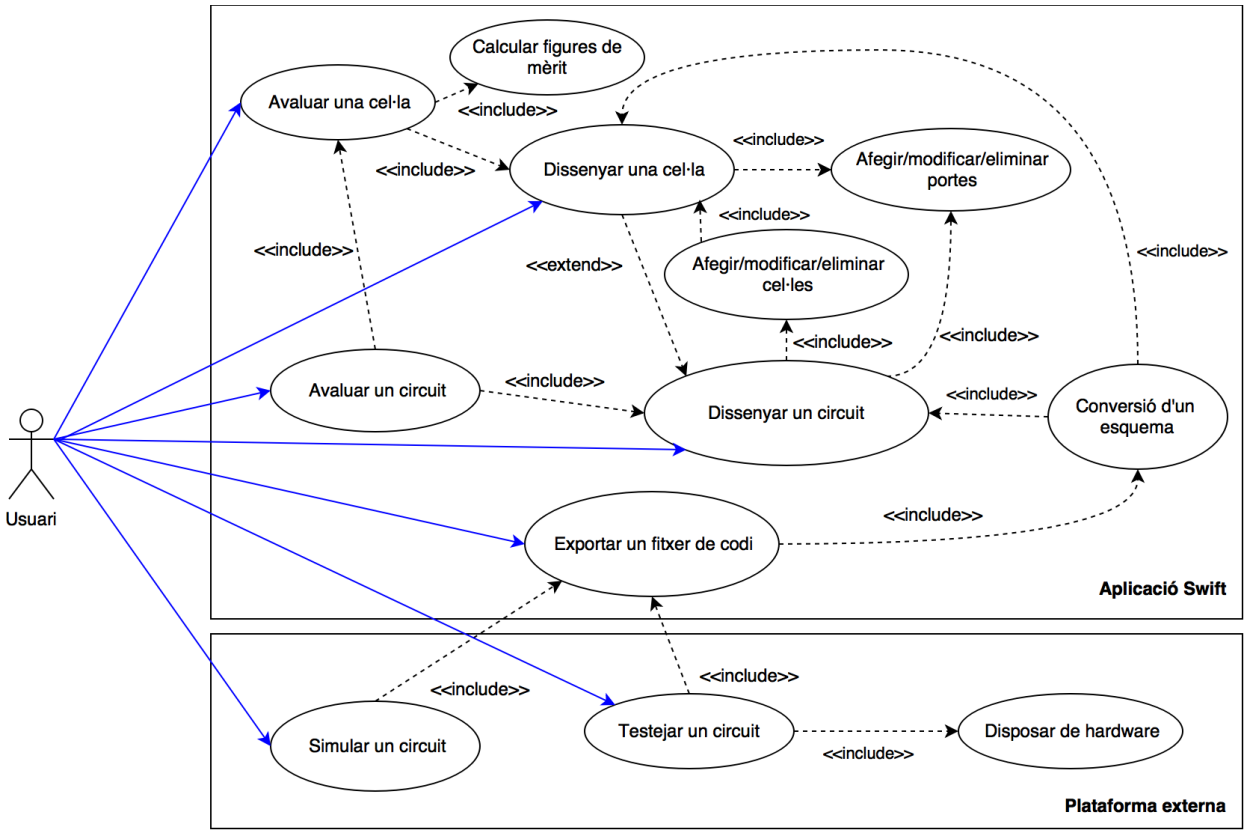

Figura 3.1: Diagrama de casos de l'aplicació.

3.3.2 Diagrama del disseny d'un circuit

Per a la tasca central, el disseny d'un circuit, s'ha elaborat un diagrama de casos més específic que conté les US secundàries, que no s'han inclòs en el diagrama anterior. Cal recordar que, com que aquesta tasca és l'eix central de l'aplicació, està relacionada amb la major part de les funcions.

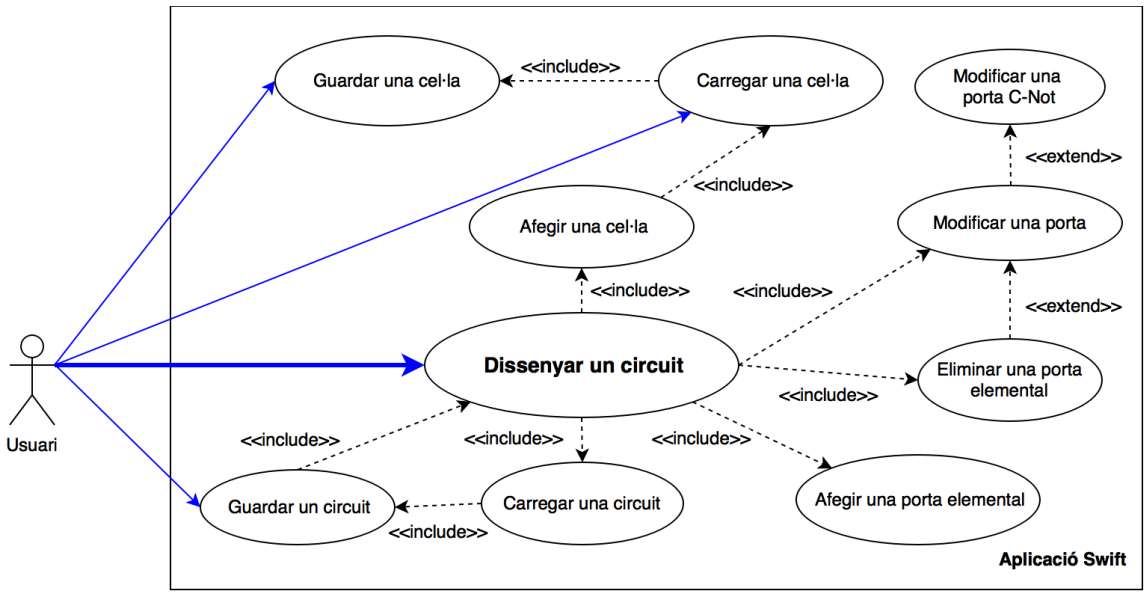

Figura 3.2: Diagrama de casos específic del disseny d'un circuit.

És important destacar que en aquest diagrama s'han començat a separar algunes funcionalitats en subtasques que, tot i que formen part de la mateixa US, s'han de tractar de manera independent. Per exemple, el cas de la modificació de la porta control-not (C-Not) requereix un tractament especial respecte de la resta de portes (això es veurà reflectit amb més detall en l'apartat de fluxos principals).

3.4 Fluxos principals de l'aplicació

En els diagrames de casos que s'han presentat en l'apartat anterior es respon a la pregunta: què pot fer l'usuari amb l'aplicació? Per continuar el procés de definició es pot intentar donar resposta a la pregunta següent que sorgeix de manera natural: com ho ha de fer? Els fluxos que es presenten a continuació responen a aquesta qüestió. Aquests esquemes tracten de descriure amb molt més detall la successió d'accions que fa l'usuari i les respostes de l'aplicació davant d'aquestes accions. Aquests fluxos, a més, estan molt relacionats amb l'experiència d'usuari, ja que indiquen de quina manera s'interacciona amb la interfície de l'aplicació.

De la mateixa manera que en l'apartat anterior, l'atenció també se centrarà en els fluxos relacionats amb la funció principal de l'aplicació, és a dir, el disseny de circuits. D'aquesta tasca se n'encarregarà la vista principal de l'aplicació que apareixerà immediatament en obrir l'aplicació i que es pot dividir en les següents seccions:

- ***Grid*:** quadrícula de disseny del circuit o de la cel·la en curs. En aquest espai s'hi poden col·locar i arrossegar les portes disponibles al *gatebox*. Quan un element es troba en el *grid*, la seva posició no és arbitrària. Està tabulada d'una manera semblant a les peces en un tauler d'escacs. Per tant, hi ha un conjunt finit de posicions disponibles en què posar les portes. Aquesta discretització també es manifesta a l'hora de desplaçar les portes, ja que el moviment de les peces es produeix en salts discrets entre posicions vàlides (de nou, com en el cas dels escacs).

- ***Gatebox*:** espai que conté el conjunt de portes («peces») que es poden utilitzar per dissenyar el circuit. També inclou els accessos a la llibreria de cel·les i a la vista de creació de cel·la.

- ***Garbage*:** zona reservada per a la funció d'eliminar portes. Quan un element s'arrossega des del *grid* fins a la zona de *garbage*, aquest s'esborra del circuit en curs.

- ***Menu/Tools*:** barra d'eines superior que conté les funcions i els enllaços que afecten globalment el circuit. Per exemple, iniciar un nou disseny, guardar els canvis, exportar els fitxers de codi, etc.

- ***Cell information*:** espai reservat per mostrar informació del circuit o la cel·la (nom del circuit, autor, data de creació, etc.). En aquesta zona també ha d'aparèixer la informació corresponent a les figures de mèrit. Els valors de les figures s'actualitzaran en temps real quan es realitzin canvis en els elements del *grid*.

En la figura 3.3 es mostra la distribució aproximada d'aquests espais en la interfície.

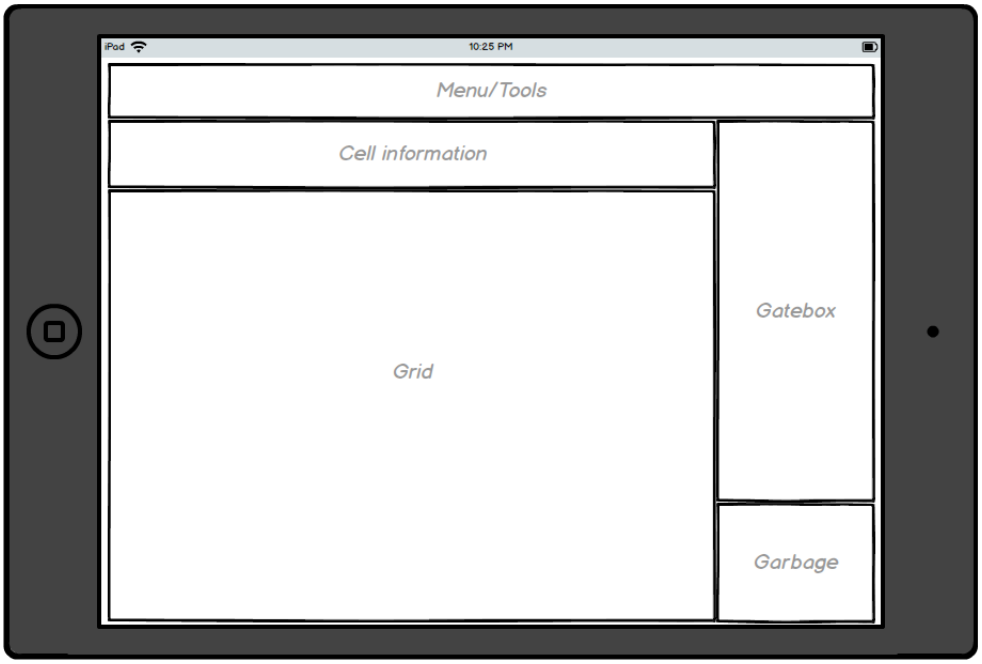

Figura 3.3: Esquema de la vista principal de l'aplicació.

Amb aquest esquema i definicions en ment ja es pot procedir a descriure els fluxos d'interacció principals. Els fluxos més senzills són els del cicle de vida de l'aplicació (figura 3.4).

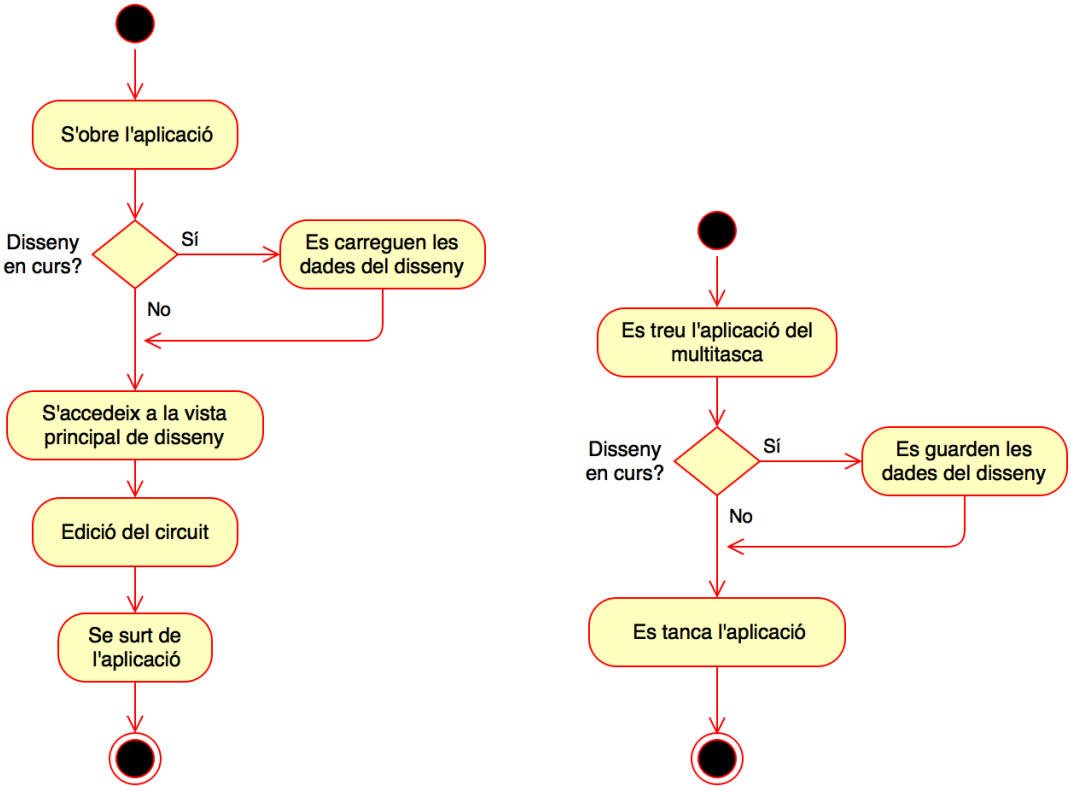

Figura 3.4: Fluxos del cicle de vida de l'aplicació.

El més destacat d'aquesta parella de fluxos és que s'ha suposat que l'usuari vol que les dades del circuit o la cel·la en curs es mantinguin quan l'aplicació sigui eliminada del multitasca. Això implica que s'ha d'incloure un mínim de gestió de la persistència local.

A continuació, es mostren els fluxos corresponents a les següents accions: carregar un circuit prèviament guardat (figura 3.5), crear un circuit nou o guardar l'actual (figura 3.6) i exportar el circuit en format de codi d'assemblador (figura 3.7). Totes aquestes tasques afecten el disseny de manera global i, per tant, són opcions que s'han d'ubicar a la barra d'eines superior. El més interessant és que, segons l'estat del disseny en curs (que pot ser: nou, modificat o guardat), l'aplicació reacciona mostrant alertes i/o *popups* amb formulari per tal de garantir que les dades dels dissenys no es perden. Això també té impacte en l'experiència d'usuari, ja que tot i que en alguns casos pot semblar un procés tediós, proporciona un punt de confiabilitat i seguretat a l'aplicació.

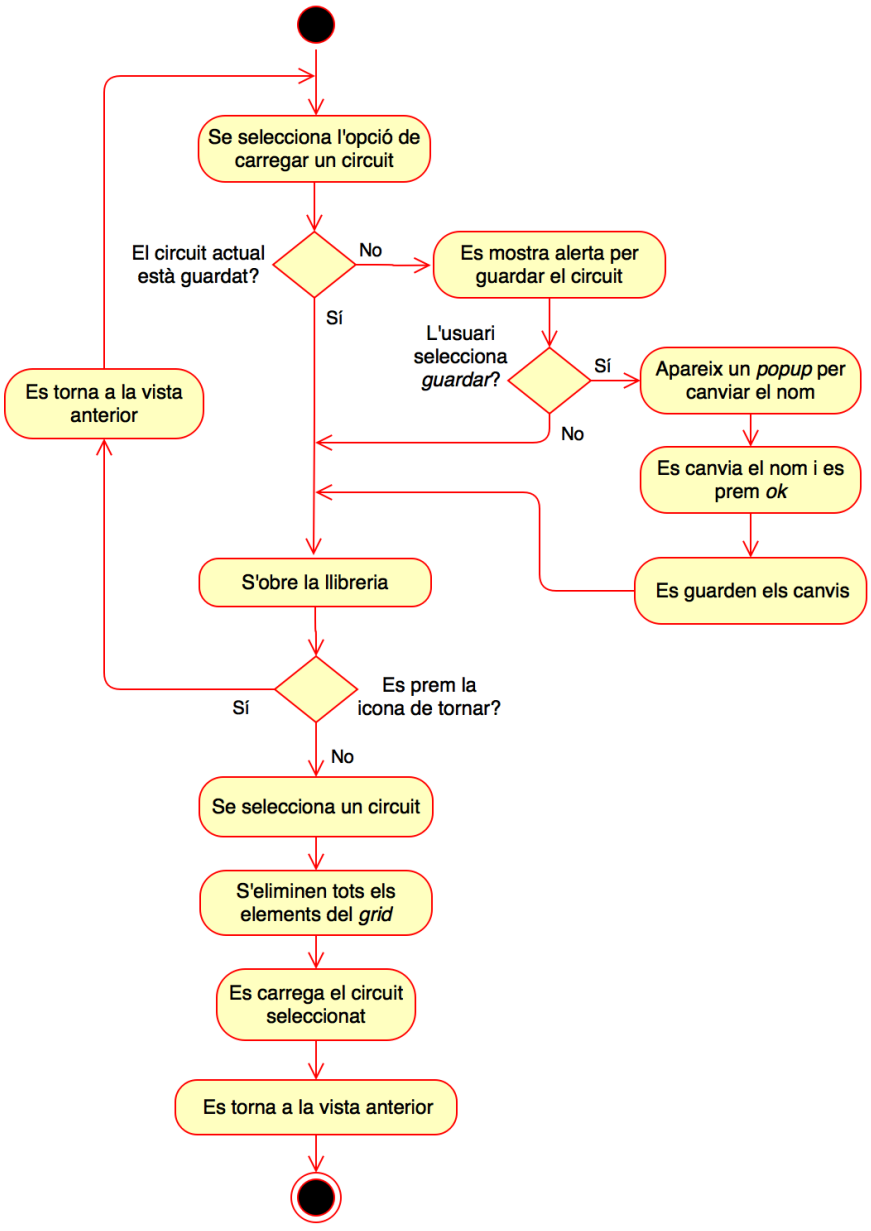

Figura 3.5: Flux per carregar un circuit.

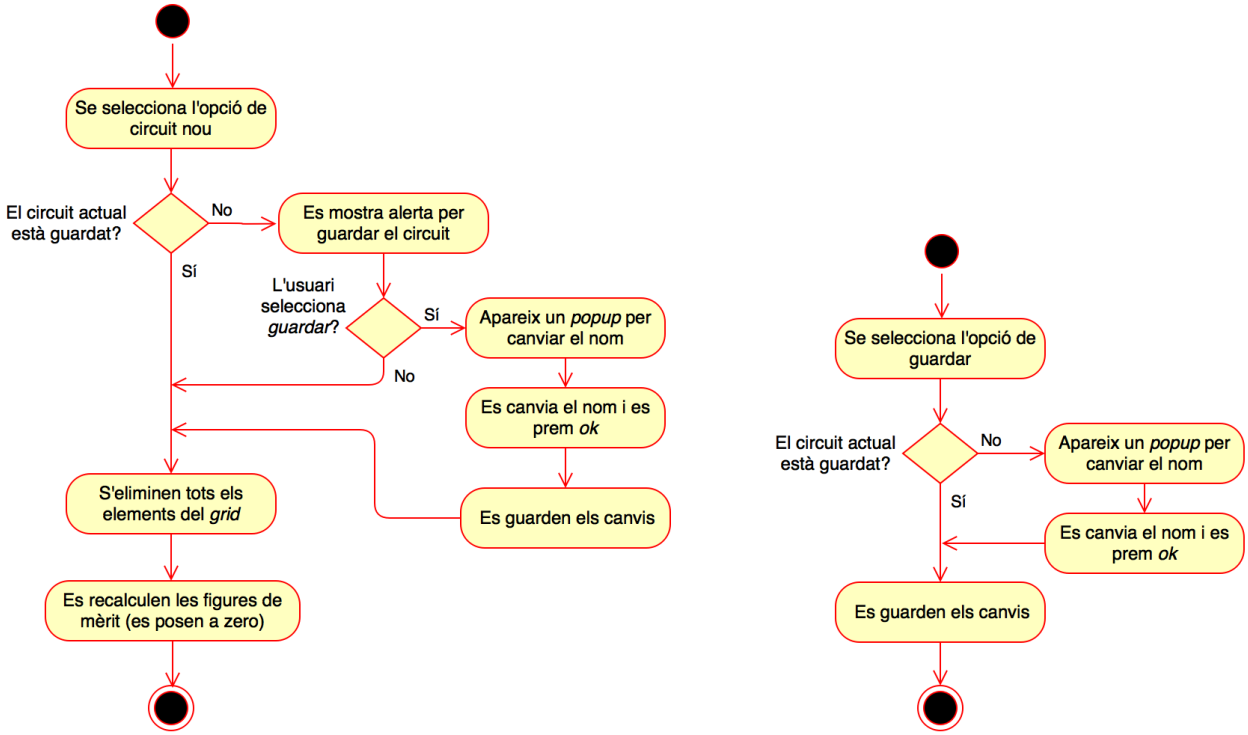

Figura 3.6: Fluxos per crear/guardar un circuit.

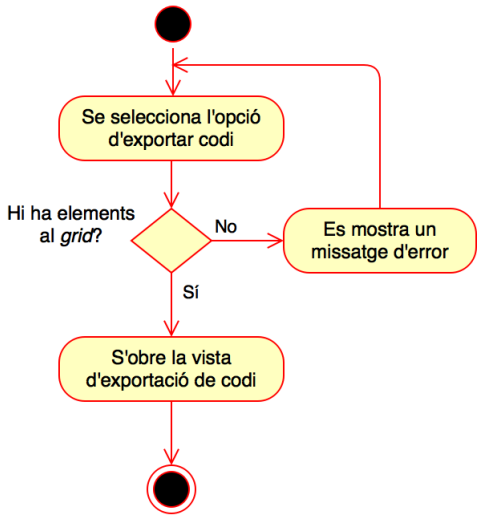

Figura 3.7: Flux d'exportació de codi.

D'aquest darrer flux es dedueix que, per tal d'exportar el codi, és necessari que com a mínim hi hagi un element al *grid*. Una condició equivalent és que la mida del registre d'entrada (que pot formar part de la informació del circuit) sigui diferent de zero.

Finalment, es mostren els fluxos de manipulació dels elements del *grid*. Més concretament, els fluxos corresponents a les accions d'afegir una porta al circuit (figura 3.8), d'eliminar i/o de moure una porta (figura 3.9) i el cas particular d'editar el *target* d'una porta C-Not (figura 3.10). Com que es tracta dels fluxos més importants del projecte s'han descrit amb més detall.

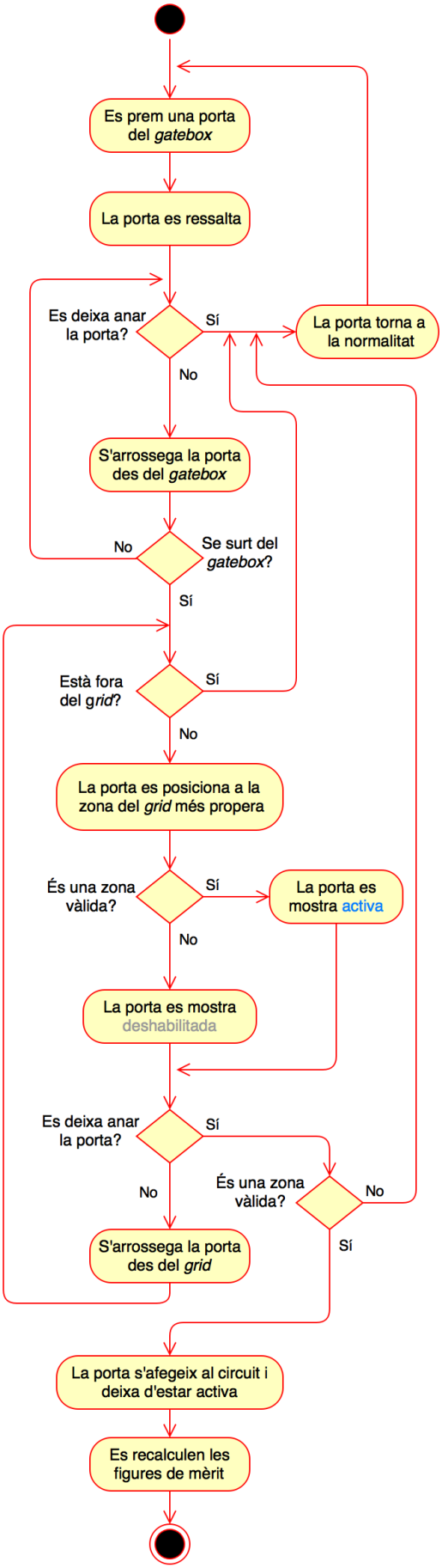

Figura 3.8: Flux d'afegir una porta al *grid*.

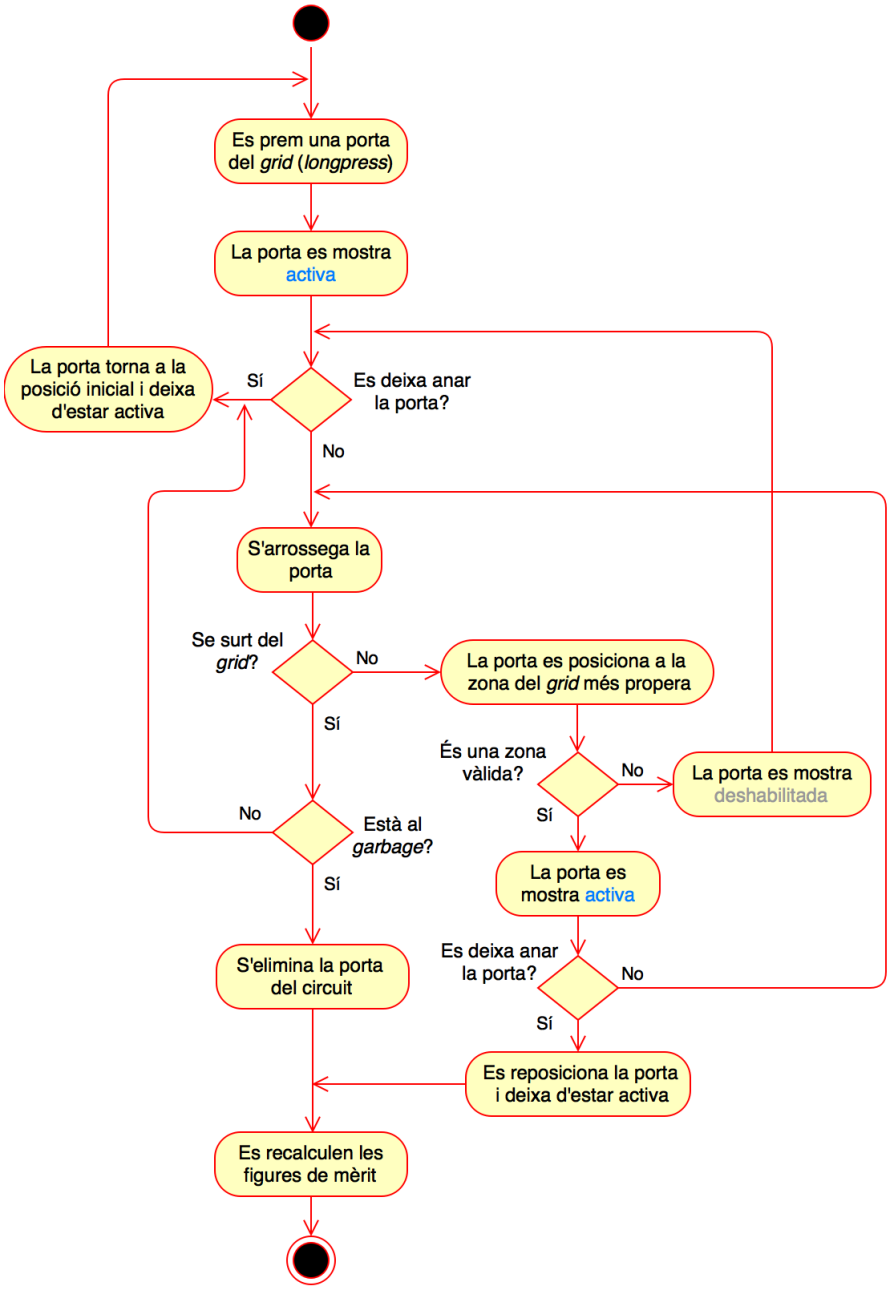

Figura 3.9: Flux d'eliminar/moure una porta del *grid*.

El flux d'eliminar o moure una porta del *grid* és vàlid per a qualsevol porta o cel·la que hi hagi a la zona de disseny. No obstant això, per al cas de la porta C-Not quan es dissenya amb línies de registre, s'ha d'incloure una interacció addicional per modificar-ne l'alçada. Aquesta modificació de l'alçada es deu a la particularitat que aquesta porta actua entre dos qubits del registre —el control (*control*) i el controlat (*target*)— i la distància entre ells pot ser, en principi, arbitrària. Per tant, la interacció amb la C-Not funcionarà de la següent manera: en primer lloc, es posicionarà la porta situant el *control* a la posició desitjada i el *target* quedarà situat al qubit immediatament inferior. A continuació, si es prem sobre el símbol del *target* ($\oplus$), es podrà modificar la posició del símbol assignant-lo a altres qubits del registre (però el qubit de control es mantindrà inalterat).

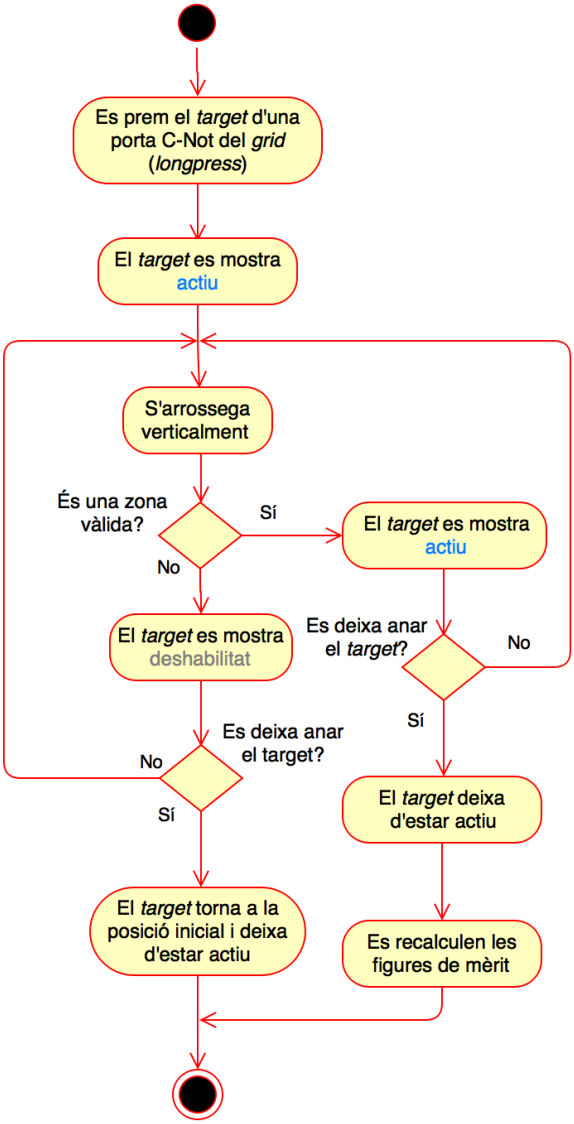

Figura 3.10: Flux d'edició del *target* d'una porta C-Not.

3.5 Prototips en baixa definició

Seguint amb la resposta a la pregunta «com ho fa l'usuari?» i amb l'objectiu de visualitzar millor l'aspecte de l'aplicació se segueix el procés de disseny presentant els prototips en baixa definició (també anomenats *mockups*). Aquests prototips s'han realitzat primer en paper i després s'han redibuixat amb el software Balsamiq [20].

En els apartats de definició d'històries d'usuari i de diagrames de casos ja ha aparegut la necessitat d'algunes vistes amb objectius ben concrets (disseny de circuit, disseny de cel·la, biblioteca de cel·les...). Aquestes són les que donen identitat a l'aplicació i ofereixen les funcionalitats que esperen els usuaris. Calen, però, algunes vistes més per donar consistència a l'aplicació i oferir un context d'ús als usuaris. Més concretament, s'han realitzat els prototips de les vistes complementàries de configuració i de suport.

En la figura 3.3 ja s'ha presentat la distribució d'espai de la vista principal. A continuació, se'n mostra el prototip.

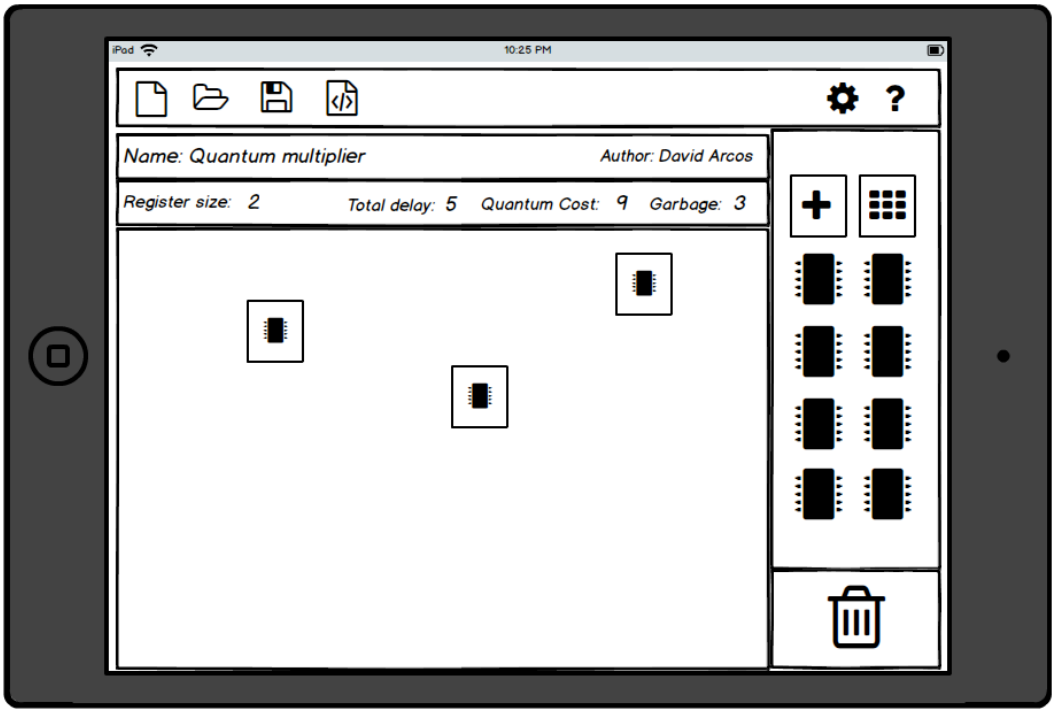

Figura 3.11: Vista disseny de circuits (*Main View*).

Aquesta és la primera vista que es mostra quan s'obre l'aplicació i, per tant, constitueix l'arrel de l'arbre de navegació. A la barra d'eines hi ha, d'esquerra a dreta, els accessos a creació d'un disseny, llibreria de circuits, guardar el disseny actual, exportació de codi, configuració de l'aplicació i, per últim, informació i suport. El panell d'informació que hi ha sota la barra d'eines mostra les dades bàsiques del disseny i els valors de les figures de mèrit. A la part superior de la zona del *gatebox* hi ha dues opcions, representades amb una creu i una col·lecció d'elements, que donen accés, respectivament, a creació d'una cel·la nova (figura 3.12) i a la biblioteca de cel·les (figura 3.13). Aquestes opcions són sempre visibles ja que es troben fixes a la part superior. La resta d'elements del *gatebox*, en canvi, es troben en un espai no delimitat, és a dir, amb scroll, de manera que s'hi poden anar afegint cel·les en funció de les necessitats de l'usuari. Finalment, a la zona del *garbage* s'hi ha afegit el símbol d'una paperera per representar-ne la funció. Aquest símbol no es correspon amb cap element clicable i, per tant, cal afegir-hi alguna diferència visual respecte de la resta de botons (per exemple, amb un color o una textura diferents).

Molt relacionada amb la vista anterior, hi ha la vista de disseny d'una cel·la, que es mostra en la figura 3.12. Cal destacar les diferències principals respecte de la vista de disseny d'un circuit que ja s'han descrit anteriorment. D'una banda, la presència de les línies de registre al *grid*, que representen l'evolució temporal dels qubits i que delimiten la mida del registre, i, de l'altra, que el *gatebox* no es pot editar i només presenta les portes elementals per defecte (la més important de les quals, la C-Not, s'ha indiciat específicament). Pel que fa a la barra d'eines superior, es pot veure que hi ha menys opcions que en la vista principal, ja que el disseny d'una cel·la és un pas intermedi i no té sentit posar-hi la resta d'opcions.

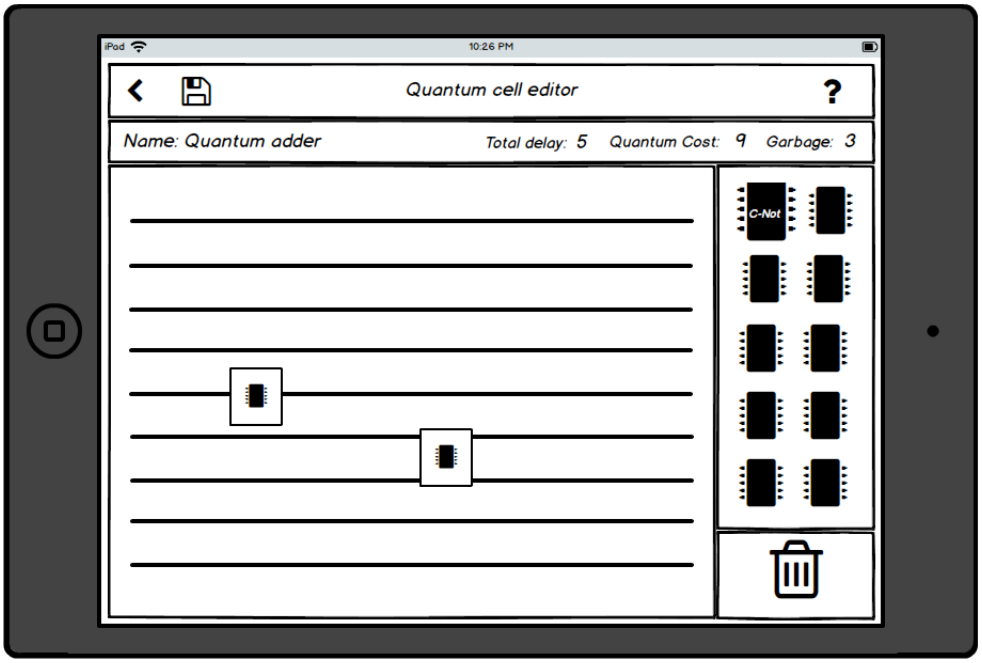

Figura 3.12: Vista de disseny d'una cel·la (*Cell View*).

La següent vista a analitzar és la biblioteca de cel·les. La funció principal d'aquesta vista és configurar les portes i cel·les disponibles al *gatebox* de la vista principal. A la part dreta hi ha les portes i cel·les seleccionades i, a l'esquerra, les cel·les disponibles a l'aplicació (la biblioteca). Per sobre de la biblioteca hi ha una zona de filtratge i un accés directe a creació d'una cel·la nova. Finalment, a la part inferior s'ha deixat indicat un espai reservat per descarregar cel·les d'un possible repositori extern.

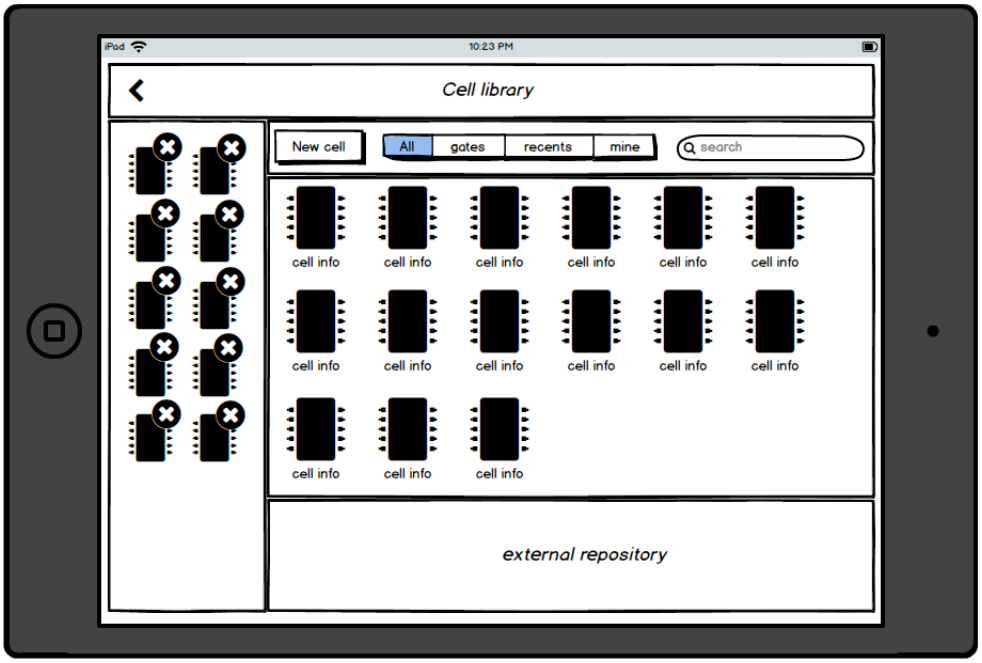

Figura 3.13: Vista de la biblioteca de cel·les (*Cell Library View*).

Pel que fa a la bilbioteca de dissenys, l'espai per carregar circuits, la distribució és molt semblant (però sense incloure el *gatebox* ni l'accés a crear un nou circuit). A la part superior dreta s'ha afegit, a més, un botó d'accés a un menú d'opcions extres, com poden ser, per exemple, el canvi al mode edició, la sincronització amb icloud, l'exportació de dades, etc.

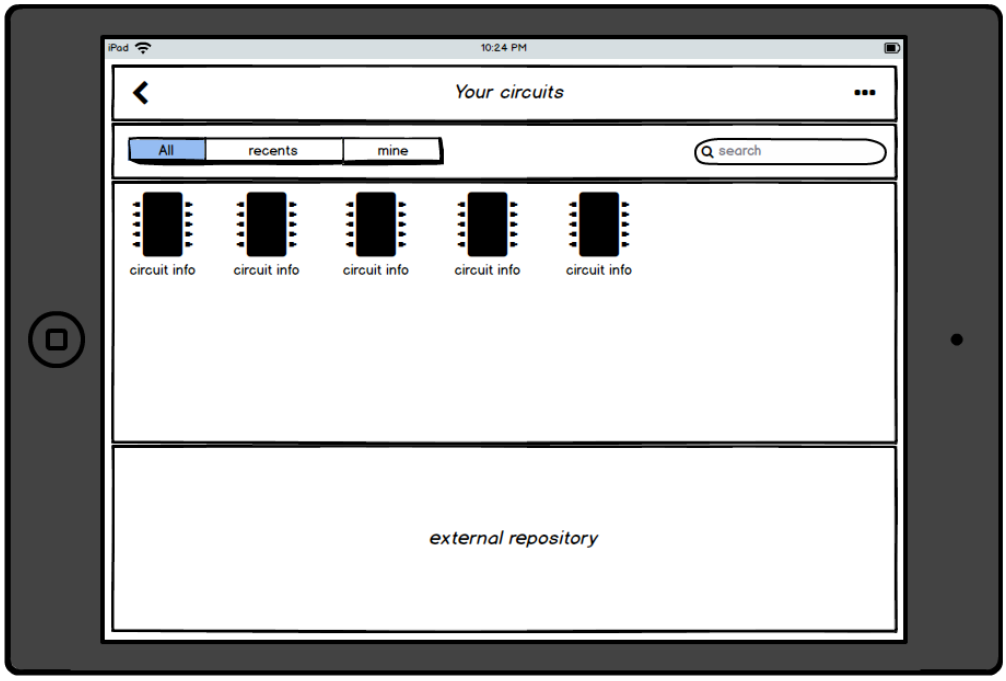

Figura 3.14: Vista de la biblioteca de dissenys (*Library View*).

A continuació, es mostra la distribució aproximada de la vista d'exportació de codi completa.

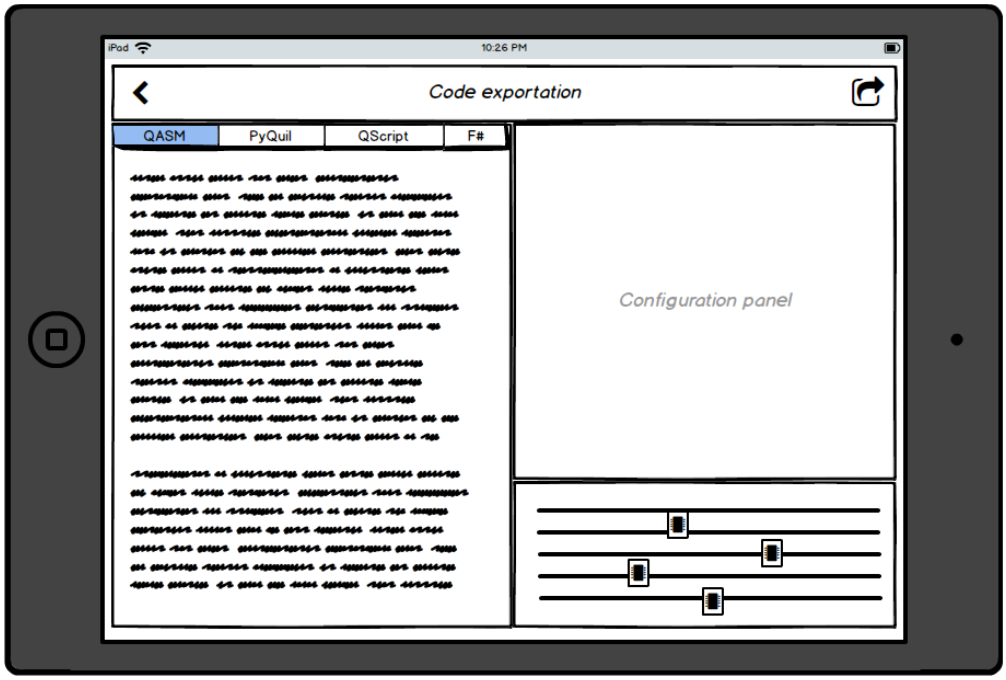

Figura 3.15: Vista d'exportació de codi (*Exportation View*).

A l'esquerra hi ha la previsualització del codi generat (amb la possiblitat de canviar entre diferents llenguatges/formats). A la dreta hi ha un espai reservat per configurar alguns aspectes de l'exportació comuns als diferents formats (extensió del fitxer de sortida, canviar el nom dels qubits d'entrada/sortida, habilitar/deshabilitar comentaris, opcions de compactificació, etc.) i una previsualització gràfica del circuit complet a nivell de portes. A la part superior dreta hi ha el botó d'exportació que obre un menú amb les diferents opcions disponibles (guardar en disc, enviar per correu, obrir amb altres aplicacions, etc.).

Finalment, es presenten les vistes complementàries de configuració de l'aplicació (figura 3.16) i d'informació i suport (figura 3.17).

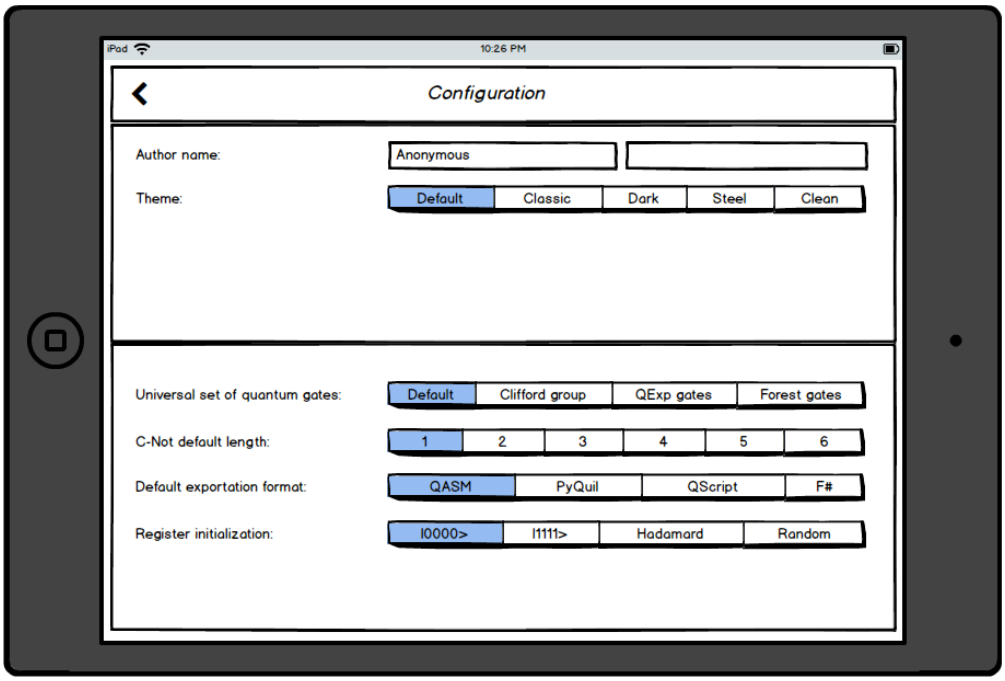

Figura 3.16: Vista de configuració (*Configuration View*).

Les opcions que es mostren a la vista de configuració són provisionals (a mode d'exemple) i s'han agrupat segons si afecten l'aplicació general (com la informació de l'usuari o l'aspecte de l'aplicació) o les preferències de disseny (quin conjunt de portes elementals s'usa, en quin format s'exporten els dissenys per defecte, etc.).

A la vista d'informació i suport apareix de manera esquemàtica l'agrupació de la informació que pot ser d'interès per als usuaris (dades de contacte, tutorials, accessos directes a les plataformes, etc.). S'ha representat utilitzant tres subvistes horitzontals iguals i amb scroll, però segons el contingut concret a mostrar se'n pot modificar la distribució (per exemple, en columnes) per tal de facilitar l'accés de l'usuari a la informació rellevant. Per escollir la distribució final (i els continguts exactes) caldria recollir informació addicional dels usuaris objectiu per ajustar-se al màxim a les seves necessitats.

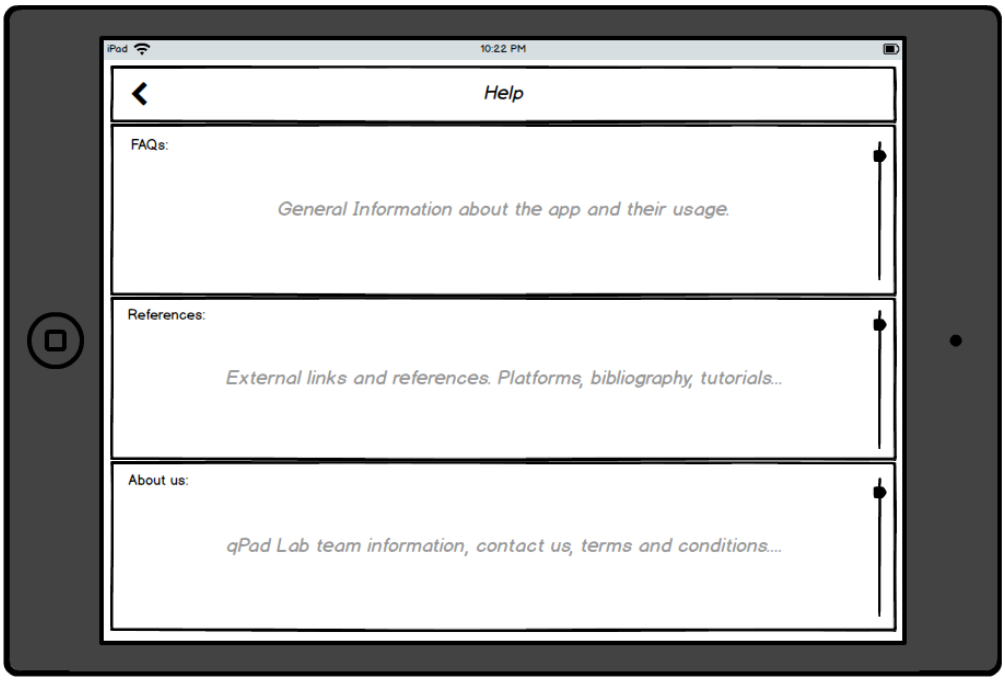

Figura 3.17: Vista d'informació i suport (*Help View*).

3.6 Arbre de navegació

Un cop definides les vistes es pot procedir a dissenyar l'arbre de navegació de l'aplicació. Es tracta d'un esquema que indica com es relacionen les vistes entre elles i com navega l'usuari a través de l'aplicació. En la figura 3.18 es mostren aquestes relacions.

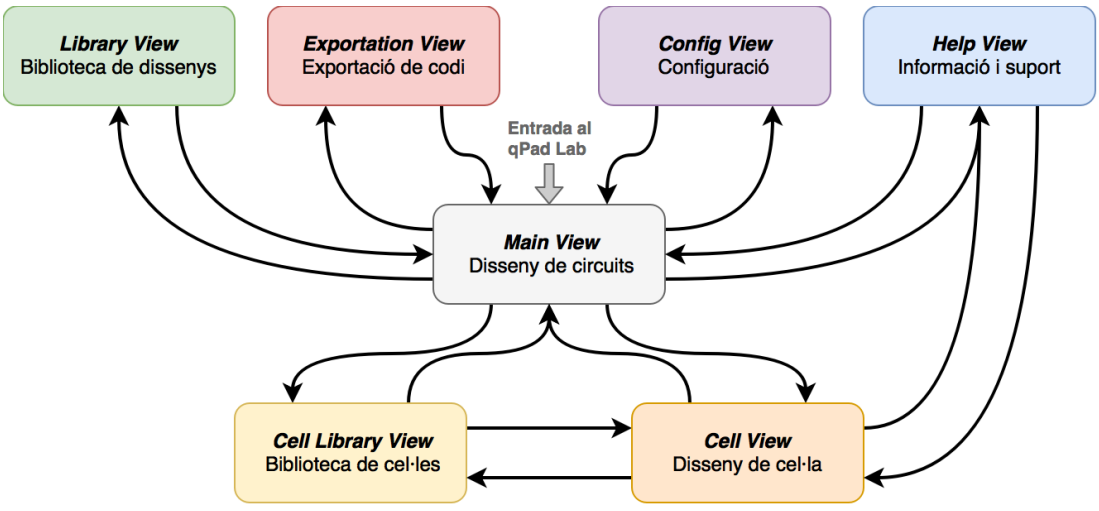

Figura 3.18: Arbre de navegació de l'aplicació.

Tal com s'ha comentat en l'apartat anterior, el punt d'entrada a l'aplicació i l'arrel de l'arbre de navegació és la vista de disseny de circuits (*Main View*). Es pot veure que el nombre d'accessos (tant d'entrada com de sortida) per aquesta vista és molt superior al de la resta. En la figura 3.19 es mostra un esquema que ajuda a entendre com es realitza la navegació d'entrada des de

la vista principal a la resta de vistes mitjançant els prototips de l'apartat anterior.

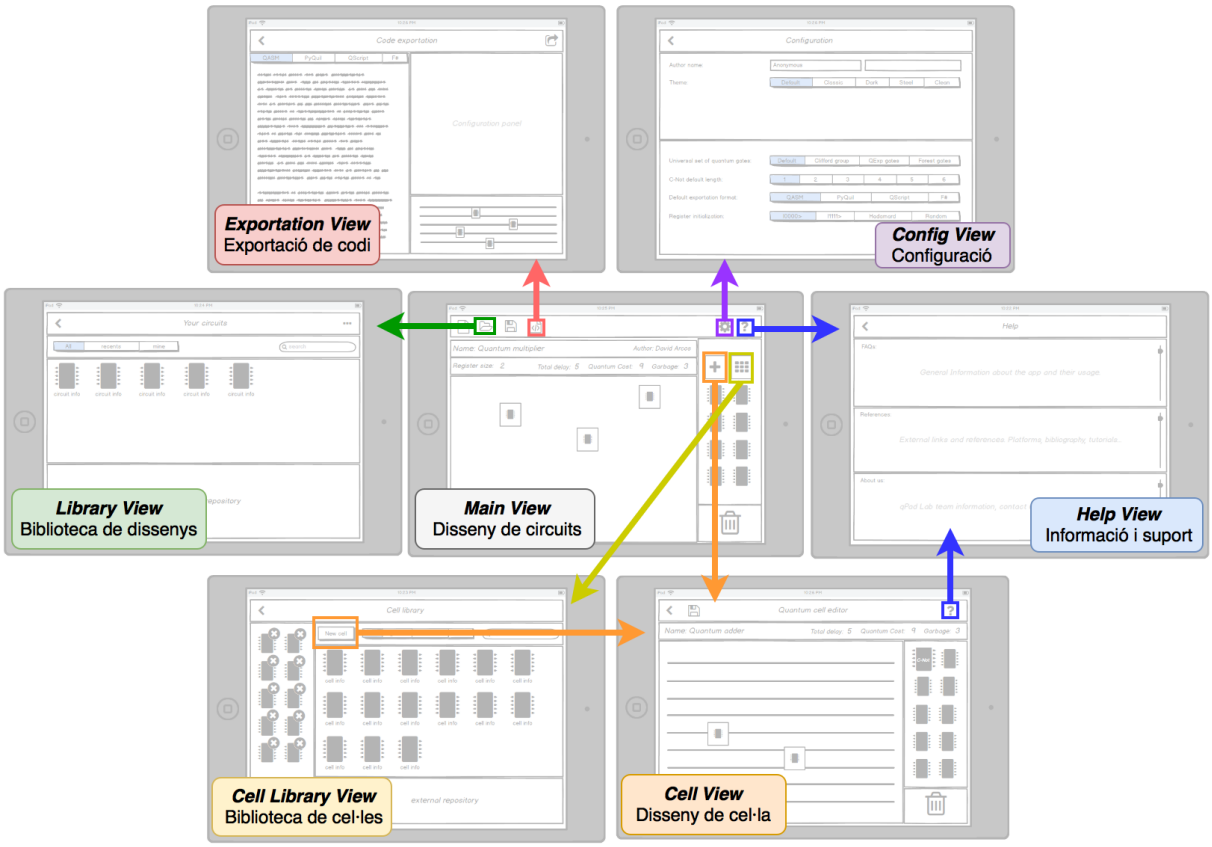

Figura 3.19: Esquema de navegació d'entrada.

La navegació inversa (de sortida) es realitza gairebé en tots els casos mitjançant els botons de tornada (*back*) de les barres de navegació. Com que hi ha algunes vistes accessibles des de diferents punts, la vista a la qual porta el *back* és condicional, ja que sempre respon al comportament de «tornar a la vista anterior». Aquest comportament és intuïtiu per a l'usuari i no introdueix problemes de navegació cíclica.

3.7 Disseny de la vista principal en alta definció

El darrer pas del procés de disseny és donar estil a l'aplicació i generar els recursos necessaris per utilitzar-los en el desenvolupament. En aquest punt s'han de prendre decisions com la filosofia de disseny que s'adoptarà, el tipus de font, la paleta de colors, etc. Aquestes decisions han d'estar justificades principlament pel tipus d'aplicació, el públic objectiu o la plataforma en què es desenvolupa, i porten a la generació dels prototips en alta definició, no només de les vistes sinó de totes les alertes, menús i icones que apareixen a l'aplicació (i que idealment s'han de validar abans de començar el desenvolupament). Aquest és un procés iteratiu que, en general, pot durar diverses setmanes. En aquest projecte, però, se simplificarà aquest procés dissenyant únicament la vista principal de l'aplicació, ja que és prou representativa per donar una imatge de l'aspecte final de l'aplicació completa i, a més, serà la part que es programarà en l'etapa de desenvolupament. En la figura 3.20 es mostra aquest disseny.

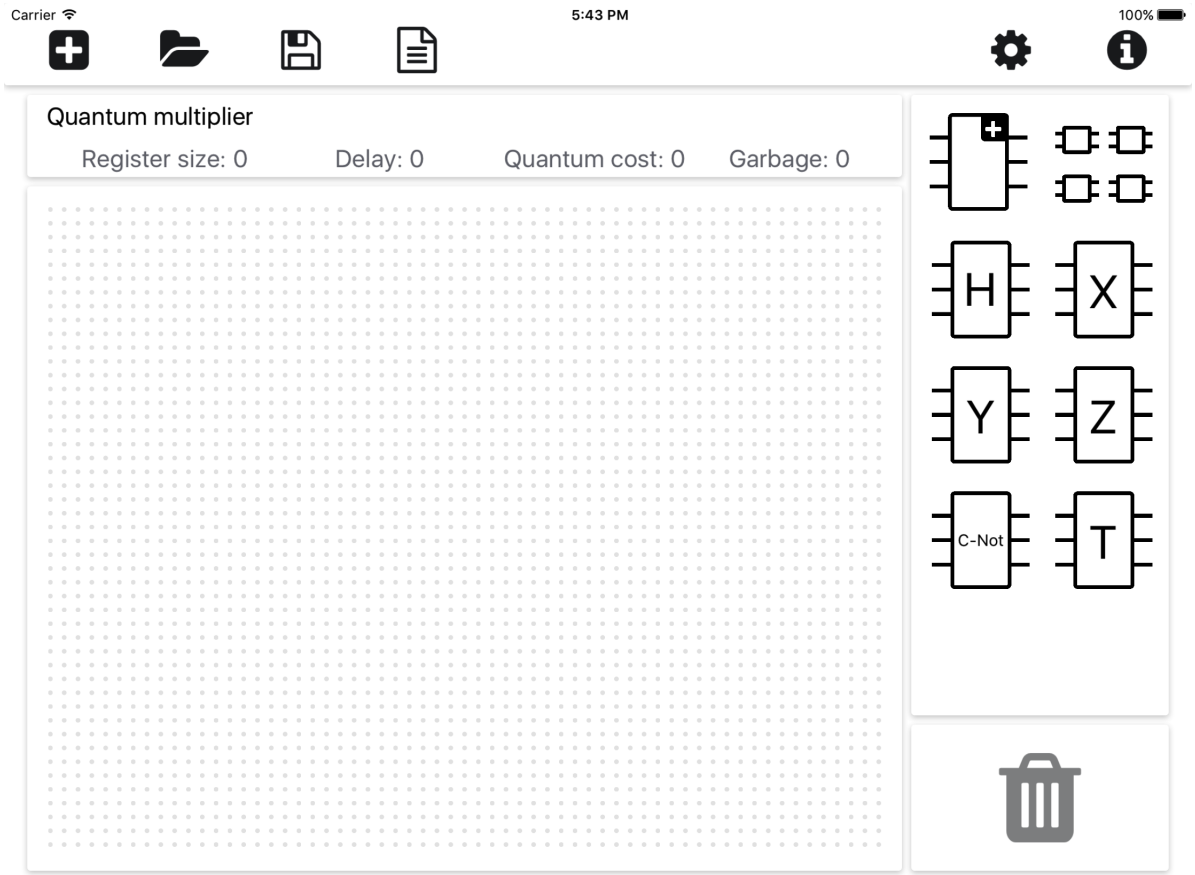

Figura 3.20: Disseny de la vista principal en alta definició.

Pel que fa a la filosofia de disseny, s'ha optat per introduir algunes pautes i elements propis del *material design* [21], ja que és un estil modern i atractiu que s'ajusta perfectament a la funcionalitat principal de l'aplicació. Per les icones de la barra d'eines s'ha utilitzat la plataforma *Fontastic* [22], ja que permet generar fonts a mida amb elements comuns a la major part de les aplicacions i, per tant, són intuïtius. En canvi, per al cas de les icones de les portes del *gatebox*, s'han generat expressament des de zero, ja que es tracta d'elements molt específics d'aquesta aplicació. Finalment, pel que fa a la paleta de colors, s'ha optat per una escala de grisos, ja que es pot associar a la implementació física dels circuits superconductors (alumini) i, a més, proporciona un aspecte formal que s'ajusta als perfils d'usuari i als escenaris d'ús de l'aplicació.

CAPÍTOL

4

MODELITZACIÓ

4.1 Registres, circuits i cel·les quàntiques

Abans de començar el procés de desenvolupament de l'aplicació cal definir i modelar els elements protagonistes de l'aplicació —és a dir, els circuits, les cel·les i les portes quàntiques—, per tal d'identificar les limitacions computacionals, així com determinar les variables i mètodes que s'han d'incloure als objectes Swift que els representen. Aquesta definició prèvia al desenvolupament és necessària a causa de la naturalesa tècnica del contingut de l'aplicació i, en certa manera, forma part del procés de disseny de l'aplicació.

La unitat bàsica d'informació en computació quàntica és el qubit. Es tracta d'un sistema físic amb un espai de Hilbert associat de dimensió 2 i, per tant, en fer una mesura sobre aquest sistema, es poden obtenir dos valors possibles, els elements de la base de l'espai, que se solen representar amb $|0\rangle$ i $|1\rangle$ per analogia amb la computació clàssica binària. La física quàntica diu, a més, que el qubit es pot trobar en un estat de superposició dels dos valors, de manera que és possible realitzar operacions en paral·lel sobre els dos valors possibles amb una única acció física. D'això se'n diu *paral·lelisme quàntic* i permet plantejar una manera de computar diferent de la clàssica. Aquests estats de superposició es representen de manera genèrica amb una parella de nombres complexos com es mostra a continuació.

$$|\phi\rangle = \alpha|0\rangle + \beta|1\rangle, \qquad \alpha, \beta \in \mathbb{C}$$

Més encara, quan es considera una col·lecció de n qubits, s'obté un registre quàntic l'estat del qual es representa amb un vector de dimensió 2^n. Per tant, per representar l'estat d'un registre quàntic en un ordinador convencional calen, en general, 2^n nombres complexos. Per aquest motiu, la simulació dels sistemes quàntics és tan costosa (creix exponencialment amb la mida

del registre). D'altra banda, en els esquemes circuitals se solen representar els registres amb un conjunt de n línies horitzontals (una per cada qubit) que representen l'evolució dels qubits al llarg del temps. Des d'un punt de vista pràctic, com que la part de simulació no es realitzarà en el qPadLab (aquesta tasca es deixa per a les plataformes externes), al codi de l'aplicació no és necessari codificar un estat genèric, només cal identificar els qubits del registre i etiquetar-los. Per tant, el model del registre es pot reduir a una llista ordenada.

Les diferents operacions que es van realitzant sobre els qubits del registre s'anomenen *portes quàntiques* i matemàticament es representen amb matrius unitàries de dimensió $2^n \times 2^n$. L'evolució temporal de l'estat del registre es pot obtenir mitjançant els productes dels vectors d'estat per les matrius que representen les portes. Per tant, de nou torna a sortir el problema del creixement exponencial dels recursos a l'hora de simular amb un ordinador clàssic. En els esquemes circuitals, les portes es representen amb caixes que actuen sobre una o més línies del registre.

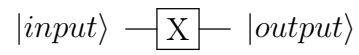

Figura 4.1: Representació d'una porta quàntica.

L'objecte central de l'aplicació és el circuit quàntic que es pot veure, en última instància, com una col·lecció ordenada de portes quàntiques que actuen sobre el registre de qubits en un moment determinat del procés de computació. Cal recordar, però, que en aquesta aplicació es pretén que els usuaris puguin dissenyar els circuits amb un nivell d'abstracció i de llibertat més gran mitjançant el reaprofitament d'estructures en forma de cel·les quàntiques i, per aquest motiu, el model del circuit quàntic passa a ser més proper al concepte de circuit clàssic, és a dir, un conjunt de cel·les amb entrades i sortides interconnectades entre elles. A continuació, es mostra un exemple de l'aspecte d'una cel·la quàntica.

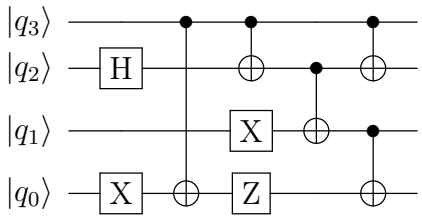

Figura 4.2: Exemple de cel·la quàntica.

Per tant, el model per a les cel·les quàntiques es pot reduir a una col·lecció ordenada de portes quàntiques aplicades als qubits del registre. Per tal d'identificar l'ordre de les portes es planteja un model gràfic basat en la posició relativa de la porta respecte dels punts del *grid*. A cada porta quàntica se li associarà un parell de coordenades enteres que codifiquen la posició de l'entrada i un altre parell que codifiquen l'amplada i l'alçada (entera) que ocupa la porta en el *grid*. Per assignar l'ordre de cada porta s'utilitzarà en primer lloc la coordenada x i en segon lloc la coordenada y. Aquestes cel·les es compactaran en noves caixes amb n entrades i n sortides (figura 4.3), que es podran utilitzar en el disseny de circuits més grans.

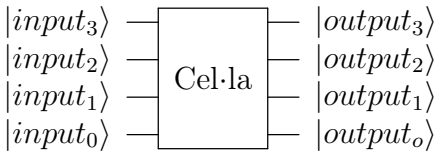

Figura 4.3: Representació d'una cel·la quàntica.

Per tant, en el disseny de circuits ja no s'usaran les línies de registre com a tal, sinó que s'utilitzaran qubits com a punts d'entrada i de sortida del circuit i uns cables quàntics que uniran les cel·les (figura 4.4).

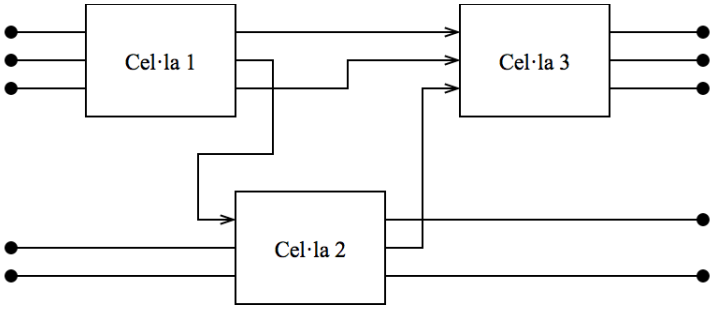

Figura 4.4: Exemple de circuit quàntic.

Així, el model del circuit quàntic es pot veure com un graf dirigit, en el qual els nodes seran els qubits dels registres i les arestes, els cables quàntics que els relacionen en un cert sentit. La figura 4.5 mostra aquesta representació.

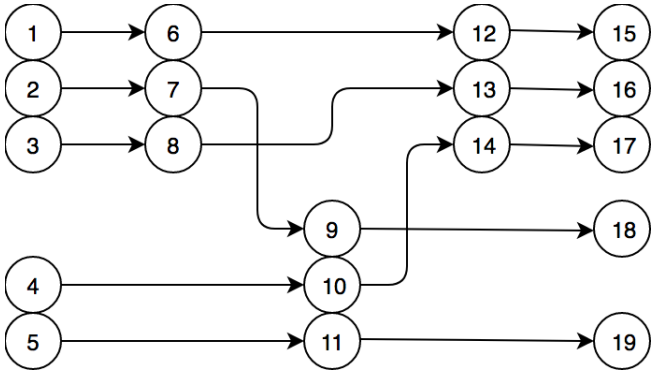

Figura 4.5: Exemple del graf d'un circuit quàntic.

Aquesta representació és útil de cara als usuaris de l'aplicació, però cal tenir present que, de fet, tots els circuits són cel·les com les de la figura 4.2. Per tant, a l'hora de generar el codi d'un circuit, n'hi ha prou de reordenar les línies de registre i aplicar les portes de cada cel·la als qubits del registre global en l'ordre corresponent, és a dir, generar una macrocel·la quàntica abans de realitzar la traducció final al codi màquina. Així doncs, pel que fa al desenvolupament, només cal implementar un algorisme de traducció de cel·la a codi i un conversor de circuit a cel·la.

4.2 Figures de mèrit

Per tal de caracteritzar les cel·les i els circuits s'utilitzaran les figures de mèrit habituals, les quals es defineixen a continuació:

- Retard o *Quantum delay* (Δ): es calcula com $N \cdot t_0$, on N es correspon amb el nombre d'operacions unitàries elementals (i.e. que involucren dos o menys qubits) que cal realitzar en una mateixa línia per tal d'obtenir el resultat de sortida, i t_0 es correspon amb el temps màxim de retard d'una d'aquestes operacions. En aquesta definició se suposa que l'entrada de la línia es troba disponible en el moment de realitzar la computació. Com que cada línia de registre té el seu propi retard, el retard total del circuit o de la cel·la en estudi es correspon amb el màxim dels retards.

- Cost quàntic o *Quantum cost* (QC): el nombre d'operacions unitàries elementals (i.e. que involucren dos o menys qubits) que cal realitzar en qualsevol de les línies per obtenir el resultat. Per al cas de la cel·la, és el nombre de portes elementals que es posen al *grid* i, per al cas del circuit, és la suma dels costs quàntics de les cel·les involucrades.

- Qubits brossa o *Garbage* (G): que és el nombre de qubits auxiliars que cal introduir al circuit per tal d'obtenir una operació reversible. No es tenen en compte les línies que es corresponen amb el qubits del resultat o les entrades si es mantenen intactes. Aquesta figura requereix d'una interpretació de l'usuari dels resultats, ja que cal determinar quins dels qubits de sortida són útils i quins no ho són. Per tant, a diferència de les anteriors, no es pot determinar de manera automàtica quan es modifica el registre.

Addicionalment, s'inclourà la mida del registre d'entrada, *register size*, que comptarà el nombre de qubits totals que calen per executar el circuit (això inclou els qubits brossa).

4.3 Portes elementals

En aquest apartat es presenten les portes bàsiques elementals (d'un o dos qubits) que es troben disponibles en totes les plataformes de simulació i testeig de circuits que s'han analitzat i que s'inclouran en el desenvolupament de l'aplicació. Pel fet de ser portes elementals tenen, per definició, les figures de mèrit següents:

Δ	QC	G
t_0	1	0

Taula 4.1: Figures de mèrit de la portes

Se'n presentarà la definició matricial i la representació circuital més habitual per tal de poder-les utilitzar com a peces en la construcció de les cel·les. Cal tenir en compte, però, que aquestes portes no formen un conjunt universal de portes quàntiques i, per tant, no permeten implementar tots els circuits possibles. Tot i això, sí que permeten realitzar algunes configuracions molt interessants (i genuïnament quàntiques) i representen un punt de partida més que acceptable per començar a treballar en el disseny de circuits quàntics.

4.3.1 Identitat

La més senzilla i elemental de les portes és la identitat. Representa la no acció sobre el qubit d'entrada durant un interval de temps t_0. També es correspon amb el model del cable quàntic que uneix les cel·les dels circuits.

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{4.1}$$

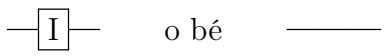

Figura 4.6: Porta Identitat.

4.3.2 Hadamard

Una de les portes més important és l'anomenada *porta Hadamard*. Permet transformar qubits en «estats clàssics» (0 o 1) en qubits en estats de superposició equilibrada. Per tant, és la clau del paralel·lisme quàntic.

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{4.2}$$

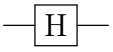

Figura 4.7: Porta Hadamard.

4.3.3 Porta de Pauli X (porta Not)

La primera de les portes de Pauli. Es correspon amb l'equivalent quàntic de la porta clàssica Not (inversió lògica del bit).

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{4.3}$$

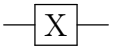

Figura 4.8: Porta de Pauli X.

4.3.4 Porta de Pauli Y

La segona de les portes de Pauli. Realitza una inversió i, a més, introdueix una fase relativa entre els estats.

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{4.4}$$

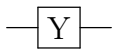

Figura 4.9: Porta de Pauli Y.

4.3.5 Porta de Pauli Z

La tercera de les portes de Pauli. Es tracta d'un cas particular de la porta desplaçament de fase (amb $\theta = \pi$).

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{4.5}$$

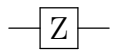

Figura 4.10: Porta de Pauli Z.

4.3.6 Porta control-not (Porta Feynman)

Una altra de les portes més importants en computació quàntica. Es tracta d'una porta de dos qubits (qubit de control i qubit controlat) que realitza una inversió del qubit controlat (li aplica una Not) quan el qubit de control presenta el valor $|1\rangle$ i no fa res en cas contrari (li aplica la identitat). Cal tenir en compte que si el qubit de control es troba en un estat de superposició, el qubit controlat també es veurà afectat per la superposició amb la mateixa ponderació (és la base de l'entrellaçament).

$$CNot = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{4.6}$$

Figura 4.11: Porta C-Not.

CAPÍTOL

5

DESENVOLUPAMENT

5.1 Aspectes generals de l'implementació

Un cop finalitzats el disseny i la modelització dels elements principals es pot començar la part pràctica del projecte, és a dir, el procés de desenvolupament de l'aplicació. Cal recordar que l'objectiu és desenvolupar una versió reduïda però funcional que permeti, com a mínim, dissenyar circuits petits, mostrar-ne les figures de mèrit i exportar el codi en QASM per usar-lo en una de les plataformes disponibles. Per tant, i com ja s'ha comentat en capítols anteriors, se centrarà l'atenció en el desenvolupament de la vista principal. Ara bé, cal tenir en compte que sense la presència de cel·les (ni la vista d'edició de cel·la) no es pot realitzar una versió útil de l'aplicació. Per aquest motiu, en comptes d'incloure el *grid* i el *gatebox* de la vista principal s'utilitzaran el *grid* i el *gatebox* de la vista d'edició de cel·la (els quals contenen només les portes elementals). Com a conclusió, el desenvolupament que es realitzarà és, de fet, un híbrid entre la vista principal (de disseny de circuits) i la vista d'edició de cel·la. Això permet obtenir un producte limitat però funcional, ja que, al cap i a la fi, tot circuit —per gran que sigui— es pot veure com una cel·la quàntica (vegeu el primer apartat del capítol anterior). A més, cal afegir que aquesta simplificació no és crítica, ja que els punts clau del desenvolupament (interacció entre zones, manipulació de col·leccions de portes...) són comuns a les dues vistes i és relativament fàcil readaptar-los de cara a futures actualitzacions de l'aplicació (per passar a una versió completa).

Pel que fa al projecte, cal recordar que l'aplicació només és compatible amb iPad (no és universal) i només es permet l'orientació horitzontal del dispositiu (*landscape*). D'altra banda, s'ha deixat el *deployment target* per defecte (11.3) i s'ha creat un projecte en blanc (sense cap plantilla predissenyada).

5.2 Hardware i software utilitzat

Per desenvolupar aplicacions per a iOS és necessari utilitzar *hardware* d'Apple i descarregar el *software* XCode per a Mac. Addicionalment, per testejar les aplicacions en un dispositiu físic cal registrar-se com a desenvolupador d'aplicacions a iTunes Connect. Pel que fa al *hardware*, en aquest projecte s'ha utilitzat, d'una banda, un iMac (8 GB de RAM i processador Intel Core i5 de 3,2 GHz) amb el sistema operatiu macOS High Sierra versió 10.13.4 i, de l'altra, un terminal de prova iPad Pro de 12,9 polzades amb iOS 11.3. Pel que fa al *software*, s'ha utilitzat la versió 9.3 de Xcode i diversos simuladors d'iPad amb mides de pantalla i capacitats tècniques diferents. Cal dir, però, que en general el testeig s'ha de realitzar amb diversos dispositius físics i que, a més, els dispositius més interessants per detectar problemes amb l'experiència d'usuari (per exemple, la fluïdesa i els temps de càrrega) són els més limitats en capacitats tècniques i, per tant, el dispositiu de prova físic que s'ha utilitzat no és el més adequat per a aquest fi.

Pel que fa a la plataforma externa de simulació s'ha utilitzat l'IBM Quantum Experience amb el processador IBM Q 5.1 (ibmqx4). Tot i que aquest processador té una forta dependència de la topologia, la possibilitat que té aquesta plataforma de carregar els codis en QASM i generar els circuits corresponents (és a dir, el camí invers al que realitza el qPad Lab) permet avaluar la validesa dels codis generats.

5.3 Classes Swift

En aquest apartat es presenten alguns dels objectes Swift que s'han desenvolupat per donar solució a les funcionalitats derivades de les històries d'usuari. Aquests objectes s'han classificat en tres categories: controladors, gestors i entitats. Els primers són els controladors de vista que segueixen el patró MVC (model, vista i controlador) habitual d'Apple i que s'associen a les vistes que s'han definit en l'apartat de disseny. Els segons (els gestors) són *singletons*, objectes únics i centralitzats que s'encarreguen de gestionar aspectes clau de l'aplicació i que són accessibles des de qualsevol punt de l'aplicació. Els darrers (les entitats) són els objectes que representen les cel·les i les portes que s'han definit en l'apartat anterior. Per tant, aquesta classificació s'ha fet segons la tasca que desenvolupen en el context general de l'aplicació.

5.3.1 Controladors

Els controladors de vista, de fet, es componen de dos fitxers. D'una banda, la classe Swift, amb la lògica i les propietats, i, de l'altra, els fitxers .xib o .storyboard, que contenen la maquetació dels objectes interactius i les regles d'adaptació a les diverses mides de pantalla (*constraints* i *autoresizing*). Es tracta de classes que hereten del *UIViewController* proporcionat per Apple i, per tant, tenen un conjunt de mètodes i propietats per defecte que s'han utilitzat a l'hora d'implementar la lògica de les vistes.

El controlador més important és el *MainViewController*, que es correspon amb el controlador de la vista principal de l'aplicació (disseny de circuits). Per aquest motiu, també és la classe amb més propietats i mètodes de l'aplicació (figura 5.1).

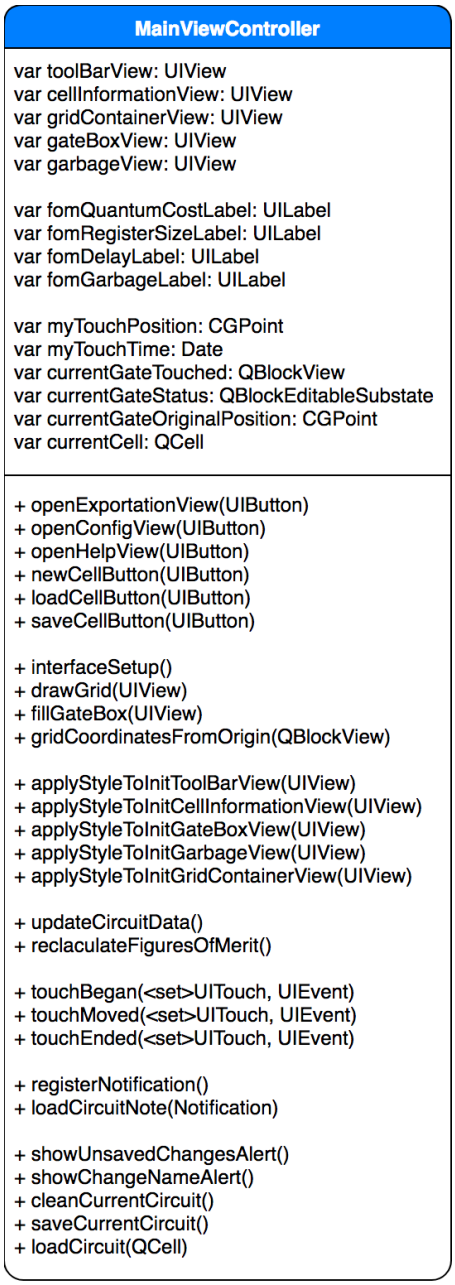

Figura 5.1: Model del controlador de la vista principal.

Entre les seves responsabilitats hi ha:

- Actuar com a contenidor de les diferents zones de treball (*grid*, *gatebox*, *garbage*, panell d'informació de cel·la i menú principal).
- Mostrar, en temps real, les figures de mèrit de la cel·la en curs.
- Gestionar la interacció amb les portes tenint en compte la zona en la qual es troben (tal com s'ha descrit als fluxos principals).
- Emmagatzemar l'estat de la cel·la en curs.
- Inicialitzar els estils dels elements amb l'ajuda del gestor d'estils.
- Controlar l'accés als altres controladors.

Segurament, la més rellevant d'aquestes tasques és la interacció tàctil amb les portes, que s'ha implementat mitjançant els mètodes *touchBegan*, *touchMoved* i *touchEnded* propis de la classe *UIView*. S'ha optat per utilitzar aquests mètodes en comptes dels *gestureRecognizers* perquè la interacció amb aquests elements depèn fortament de la zona en la qual es troba l'element (i també del seu estat) i, a més, perquè el moviment en el *grid* és discret.

Un altre controlador que cal mencionar és el ViewController que crea per defecte XCode quan es genera un projecte nou (figura 5.2). Aquest controlador s'ha deixat com un possible punt d'entrada alternatiu a l'aplicació mitjançant lògica a l'*appDelegate*. Podria ser útil, per exemple, per mostrar una pantalla de càrrega personalitzada posterior al *launch screen* o bé per un possible tutorial inicial. També s'ha utilitzat com a contenidor dels possibles mètodes d'extensió de *UIViewController*. En particular, s'ha implementat un mètode per mostrar missatges temporals (semblants als *toasts* d'Android).

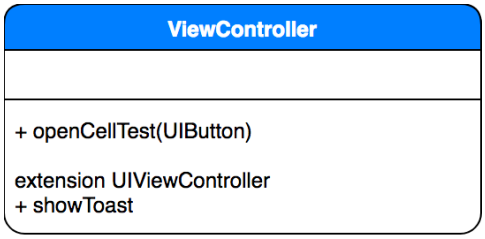

Figura 5.2: Model del controlador de vista incial.

A banda d'aquests controladors, se n'han creat alguns més per realitzar proves de navegació i començar a preparar l'arquitectura de l'aplicació completa. Més concretament, s'han desenvolupat els controladors *ExportationViewController*, *HelpViewController*, *ConfigViewController* i *LibraryViewController*, que es corresponen, respectivament, amb els controladors de les vistes d'exportació de codi, d'informació, de configuració general de l'aplicació i llibreria de circuits.

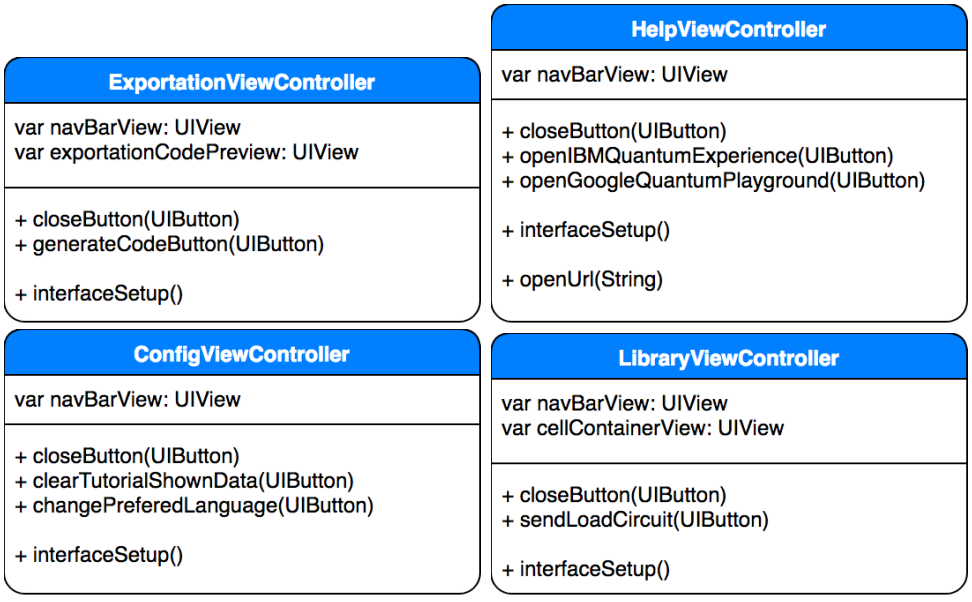

Figura 5.3: Model dels controlador secundaris.

5.3.2 Gestors

Els gestors s'han diferenciat de la resta d'objectes per la seva naturalesa transversal i unificada, i perquè duen a terme una tasca molt concreta. Com ja s'ha comentat, es tracta de *singletons* que s'instancien un sol cop (el primer cop que són cridats) i que després són accessibles mitjançant l'atribut *shared*. No cal incloure punters a aquests objectes, ja que els seus atributs i mètodes són accessibles des de qualsevol punt del codi.

El primer gestor es correspon amb l'avaluador de circuits (figura 5.4). Aquest gestor, de fet, no és *singleton* sinó una classe pública que conté mètodes accessibles per a la resta d'objectes (com si fos una llibreria) i s'encarrega d'avaluar les figures de mèrit del circuit o cel·la que se li passa com a paràmetre. Es tracta d'una tasca essencial que s'ha d'ajustar, controlar i optimitzar a mesura que s'avanci en el desenvolupament de l'aplicació. Aquest mètodes s'executen constantment cada cop que es modifiquen els elements del circuit o la cel·la en curs.

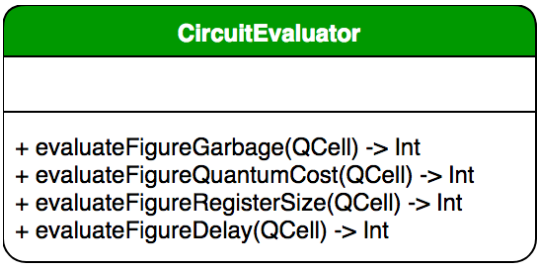

Figura 5.4: Model del avaluador de circuits.

El següent gestor (figura 5.5) és el traductor de dissenys de portes (esquemes) a codis quàntics (descripció de *hardware*). La seva missió principal és convertir els objectes Swift que representen les cel·les i els circuits en cadenes de text en llenguatges compatibles amb les plataformes externes. Addicionalment, s'han inclòs alguns mètodes per crear i gestionar fitxers generats en aquests llenguatges. De moment, però, només s'han inclòs els llenguatges QASM (compatible amb el *Quantum Experience* d'IBM) i QScript (compatible amb el *Quantum Computing Playground* de Google).

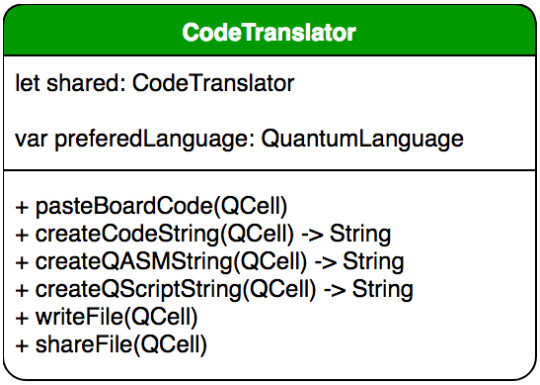

Figura 5.5: Model del gestor de traduccions.

Un altre gestor que s'ha desenvolupat és l'anomenat *StyleManager*, el qual s'encarrega de centralitzar alguns dels estils dels elements centrals de l'aplicació (figura 5.6). Conté, per exemple, alguns paràmetres configurables del *grid*, un mètode per aplicar efectes propis del *material design* i la definició dels efectes visuals de les portes quàntiques.

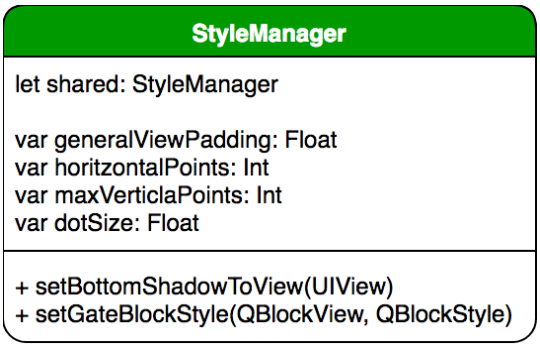

Figura 5.6: Model del gestor d'estils.

Pel control de la gestió de les dades de l'aplicació (incoent-hi la persistència local) s'ha desenvolupat un gestor anomenat *DataManager*, el qual s'encarrega d'emmagatzemar els circuits i gestionar-ne la càrrega i descarrega de la memòria del dispositiu (figura 5.7). Com que la persistència s'ha implementat utilitzant el protocol *NSCoding*, part de la feina d'aquest controlador s'ha traslladat a les entitats que representen a les portes i les cel·les (mitjançant els mètodes *code* i *encode* del protocol).

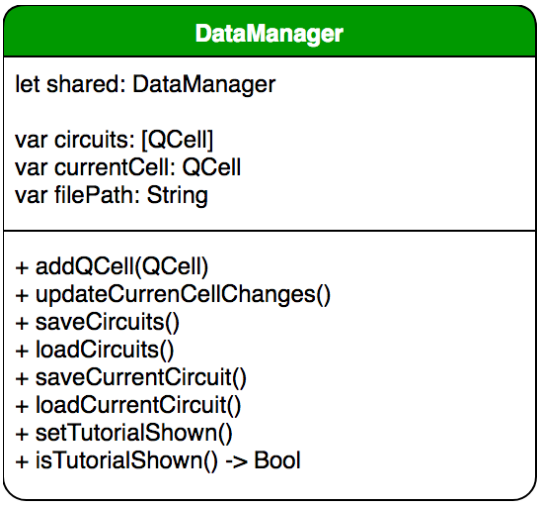

Figura 5.7: Model del gestor de Dades.

El darrer gestor (figura 5.8), tampoc és *singleton* sinó una classe pública que conté mètodes accessibles per a la resta d'objectes relacionats amb càlculs matemàtics específics de l'aplicació. De moment, però, només conté mètodes de generació de nombres aleatoris que s'han utilitzat per realitzar testos d'eficència i d'ús de memòria.

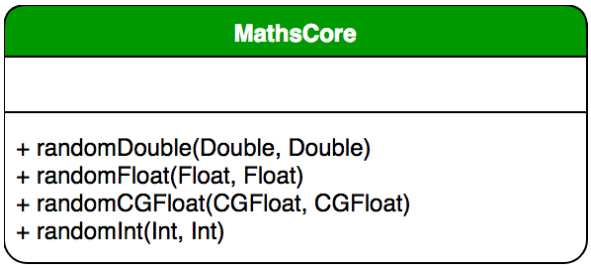

Figura 5.8: Model del gestor de mètodes matemàtics.

A més d'aquests gestors també s'ha plantejat la possibilitat d'incloure un gestor de navegació per construir una arquitectura flexible davant de possibles canvis en l'arbre de navegació. Aquest gestor s'ha descartat per la senzillesa de l'arbre actual, però pot ser interessant incloure'l si l'aplicació creix i/o si es preveuen canvis importants en la navegació.

5.3.3 Entitats

Les entitats representen els objectes Swift dels elements centrals de l'aplicació, és a dir, les cel·les i les vistes de les portes elementals. Aquests objectes emmagatzemen la informació del treball que fa l'usuari amb l'aplicació i, per tant, són els primers elements que s'han de tenir en compte a l'hora de treballar en la persistència.

L'element de jerarquia superior és, de moment, la cel·la quàntica (figura 5.9) que, tal com s'ha modelat, conté una col·lecció de portes i una codificació del registre d'entrada. D'altra banda, aquest objecte conté mètodes per afegir, treure i ordenar les portes, de manera que la llista estigui sempre preparada per al gestor de traduccions.

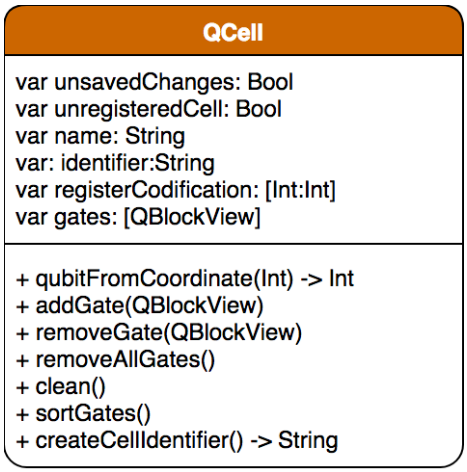

Figura 5.9: Model de la cel·la quàntica.

El següent objecte, el *QBlockView*, es correspon amb la vista de la porta quàntica elemental (figura 5.10). Es tracta d'una classe genèrica que hereta de *UIView*, que agrupa les propietats abstractes comunes de les portes i que defineix les estructures de dades necessàries per codificar-ne la informació.

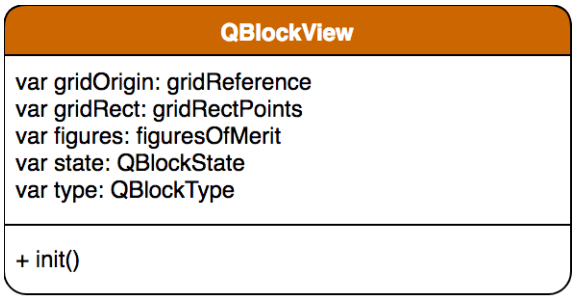

Figura 5.10: Model de la vista d'una porta.

Per a cada porta elemental s'ha fet una extensió de *QBlockView*, on s'han definit els paràmetres inicials i s'ha sobreescrit el mètode *draw* per, mitjançant *CoreGraphics*, dibuixar els diagrames vectorials dels símbols de les portes. A la figura 5.11 es mostren les tres classes que s'han desenvolupat i es pot veure que les tres portes de Pauli s'han agrupat en una sola classe (es diferencien per la propietat *type*).

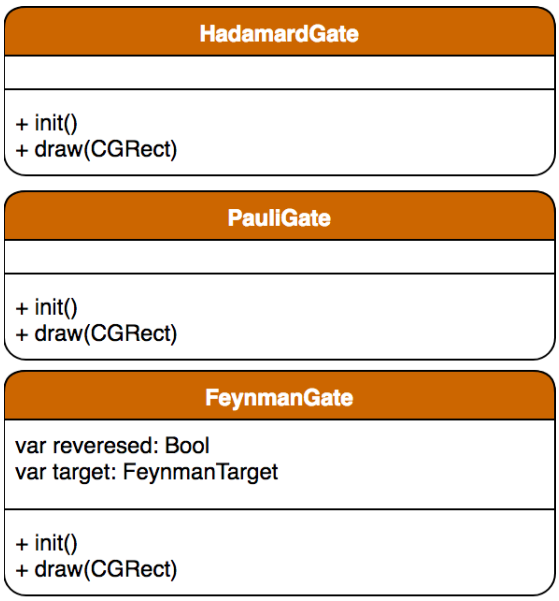

Figura 5.11: Model de les portes bàsiques.

Cal destacar que la control-not (*FeynmanGate*) té dues propietats addicionals. La primera és una propietat booleana que indica si el qubit de control i el controlat estan invertits o no. La segona és un punter a un objecte de classe *FeynmanTarget* (figura 5.12), que és una subclasse de *UIView* que representa la zona clicable del *target* de la porta.

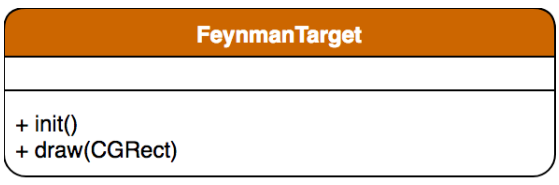

Figura 5.12: Model de la subvista del *target* de la control-not.

5.4 Diagrama de classes

Un cop presentades les principals classes del projecte és interessant veure'n les relacions en un diagrama de classes. Això permet obtenir una imatge global del desenvolupament i de la jerarquia que hi ha entre els objectes Swift. Per tal de simplificar el diagrama (figura 5.13), s'han obviat els mètodes i totes les propietats que no es corresponen amb els objectes desenvolupats.

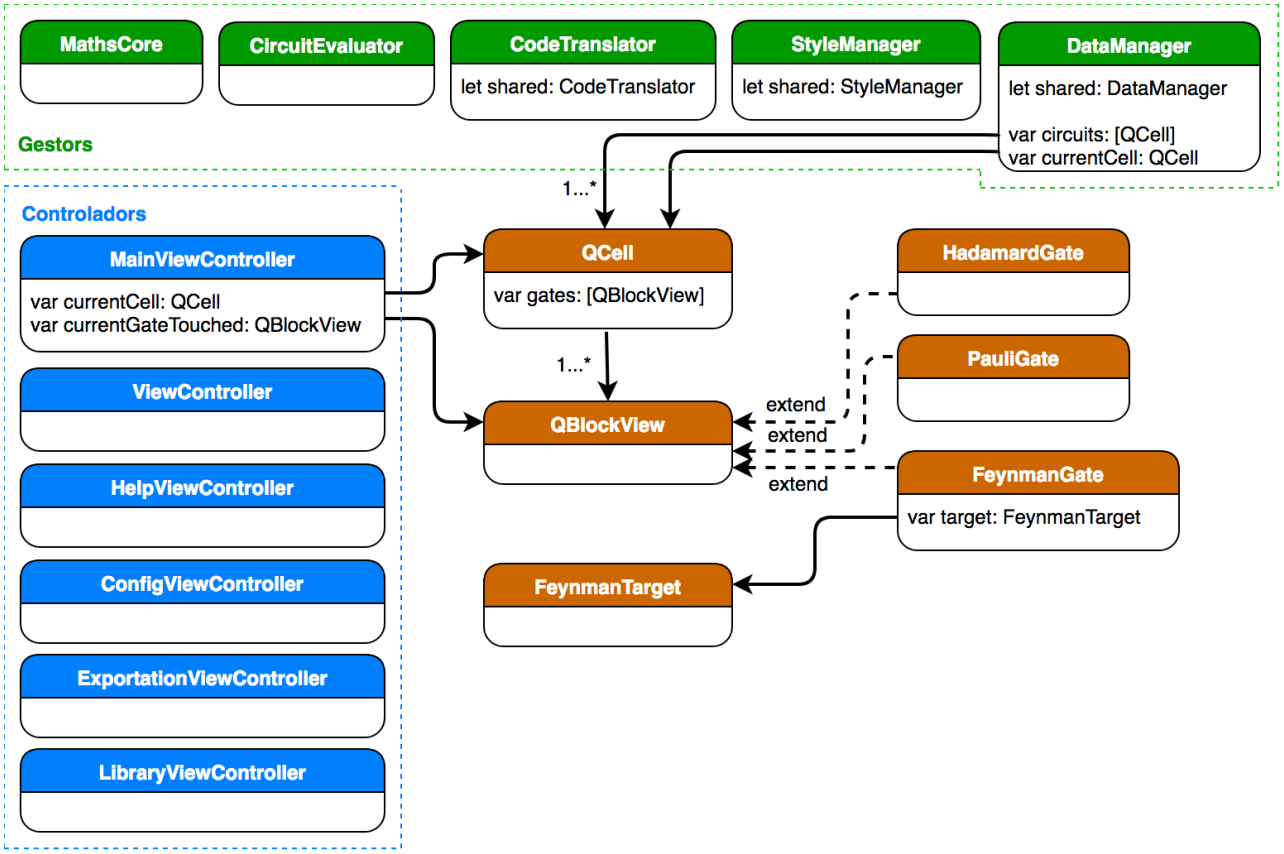

Figura 5.13: Diagrama de relacions entre objectes.

En la figura 5.13 s'ha utilitzat la notació «1...*» per als *arrays* d'objectes, les fletxes contínues representen els punters, i les discontínues, les extensions.

És interessant destacar que, tot i que els gestors *singleton* s'han representat aïllats, les propietats *shared* són com punters a l'objecte únic compartit i, per tant, estan més que relacionats amb la resta d'objectes.

CAPÍTOL

6

RESULTAT I TESTS

6.1 Resultat del desenvolupament

Un cop finalitzat el desenvolupament es pot mostrar el resultat obtingut.

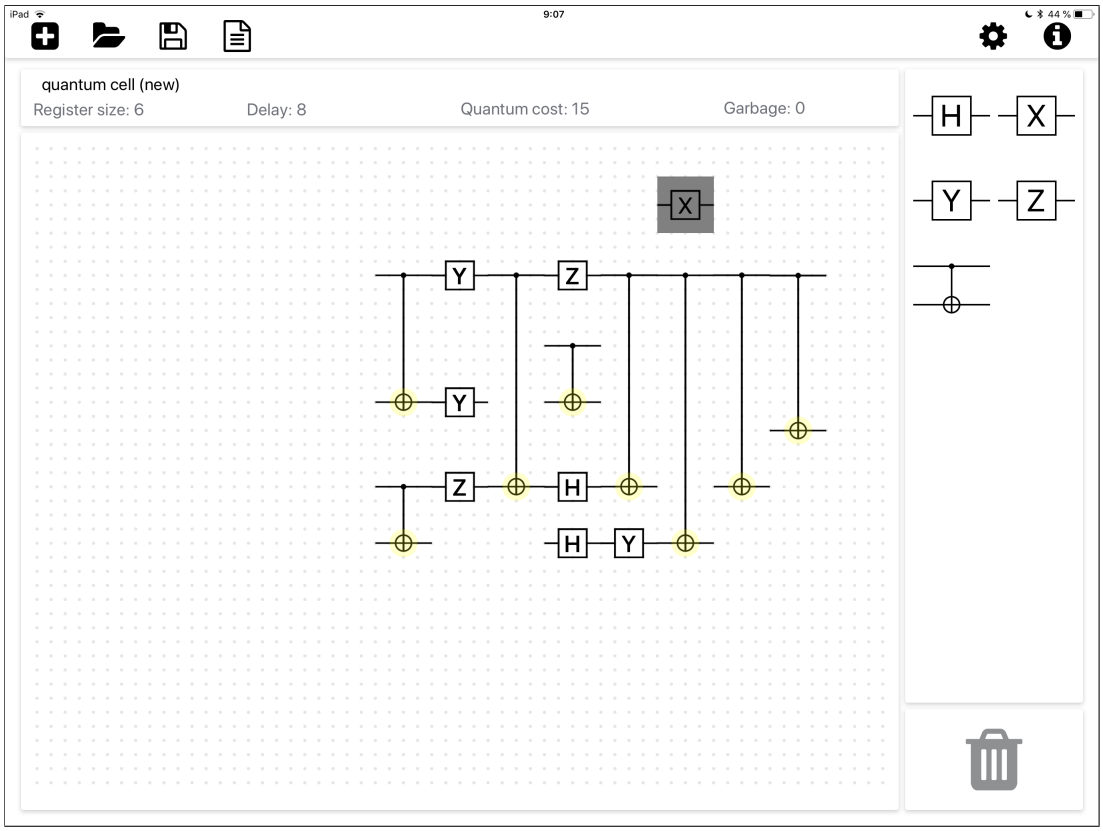

Figura 6.1: Aspecte final de la vista principal de l'aplicació.

La vista principal (figura 6.1) presenta tots els elements, zones de treball i enllaços descrits en l'apartat de disseny i incorpora l'implementació dels fluxos principals de l'aplicació. Això inclou la interacció amb les portes del *grid*, la gestió de la persistència (amb les alertes corresponents) i l'actualització automàtica de la informació de la cel·la en curs. La part més complicada del controlador associat a aquesta vista ha estat la interacció amb les portes. A més, també s'ha implementat el cas concret de la interacció amb el *target* de la control-not. Aquesta interacció no era imprescindible per als objectius del projecte, però dona molt joc de cara a dissenyar circuits més complexos i és esencial de cara a la versió final de l'aplicació.

D'altra banda, amb l'objectiu d'avançar en la navegació de l'aplicació completa, s'han implementat els controladors de les vistes secundàries. El més rellevant és el de la vista d'exportació de codi (figura 6.2), el qual inclou un botó per copiar el codi generat al *clipboard* per usar-lo a les plataformes externes. També es pot veure en gris fosc el missatge temporal informatiu «*QASM code has been copied to clipboard*», que apareix quan es fa clic al botó d'exportació. L'últim pas per executar el codi és accedir a la plataforma corresponent des d'un navegador i copiar el contingut que hi ha al *clipboard*.

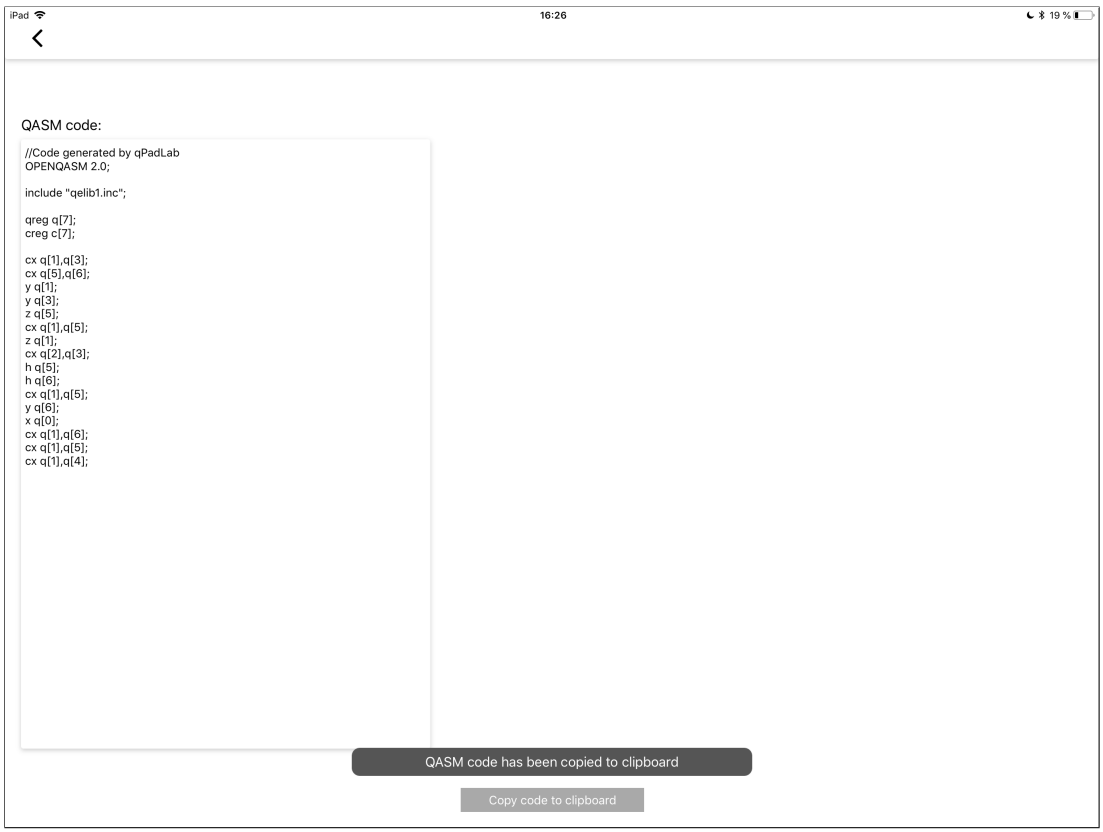

Figura 6.2: Aspecte final de la vista d'exportació de codi.

S'ha optat per aquest sistema d'exportar els resultats per dos motius: d'una banda, és molt simple d'implementar i fàcil d'utilitzar per a l'usuari i, de l'altra, en l'ecosistema iOS amb el *universal clipboard* és possible realitzar el disseny amb l'iPad i enganxar el codi resultant al Mac (cosa que ha estat molt útil en el procés de test). Aquest sistema no és l'únic que s'ha plantejat (ni implementat) al gestor de traduccions. De fet, segons la plataforma, es podria

arribar a realitzar una connexió directa amb l'API del *back-end* per obtenir els resultats de la simulació directament a l'aplicació. No obstant això, aquest desenvolupament de comunicació amb les plataformes queda fora de l'abast d'aquest projecte i la relació benefici/cost respecte de la solució de copiar al *clipboard* és baixa.

6.2 Tests d'usabilitat

El darrer pas és provar si l'aplicació és funcional i compleix els objectius establerts. Per realitzar els testos d'usuari s'ha utilitzat un iPad Pro de 12,9 polzades (el mateix que s'ha utilitzat en el procés de desenvolupament) i els tres usuaris següents:

- **David Arcos:** el desenvolupador de l'aplicació i autor del projecte. Els testos han estat principalment enfocats als diversos casos d'ús (i els fluxos principals associats) fent especial èmfasi en l'anàlisi de la fluïdesa i la velocitat de resposta en la interacció amb les portes. Aquests testos han permès ajustar alguns paràmetres d'interacció i detectar errors en el codi. També s'han fet proves de dissenys concrets amb diferents configuracions per exportar a les plataformes d'IBM [3] i de Google [5] i comprovar que els diagrames generats coincideixen. Finalment, s'han fet proves bàsiques de persistència en carregar i guardar circuits a la memòria (per a aquest test s'ha hagut de desenvolupar una versió bàsica de la llibreria de circuits).

- **Dani Gabriel:** un perfil tècnic amb coneixements bàsics de computació quàntica. El primer test ha estat per estudiar com d'intuïtiva és l'aplicació. Sense explicar com funciona se li ha deixat perquè descobreixi què pot fer i com es fa. Després s'han fet algunes comparacions amb altres eines similars i s'ha comprovat la millora respecte del disseny directe en una plataforma concreta. Finalment, s'han fet algunes proves de persistència (apagant el dispositiu, eliminant l'aplicació del multitasca, etc.).

- **Mireia Trias:** un usuari novell amb pocs coneixements sobre el tema, però amb destresa en l'ús d'aplicacions mòbils multitàctils. Se li ha fet una breu introducció, se li ha explicat com funciona i se li ha dibuixat un circuit en paper que ha hagut de reproduir amb l'aplicació. Els testos han estat, sobretot, per analitzar la part d'interacció. S'ha detectat que el *longpress* d'edició de porta no és prou intuïtiu, ja que l'usuari esperava que es poguessin moure directament. També s'ha vist la necessitat d'incloure les línies de registre que estan actives (o bé la porta identitat) i la possibilitat de seleccionar múltiples portes alhora. Els testos amb aquest usuari han acabat de confirmar la necessitat del tutorial inicial.

A continuació, es mostren els resultats dels testos més rellevants del projecte.

6.2.1 Test d'interacció amb les portes

Es tracta de provar que el moviment de les portes i la interacció entre aquestes i la resta d'elements es correspon amb l'esperat. En la figura 6.3 es mostra una porta Y de Pauli que s'està arrossegant per les diferents zones de la vista principal i canvia el color de fons segons la posició en la qual es troba.

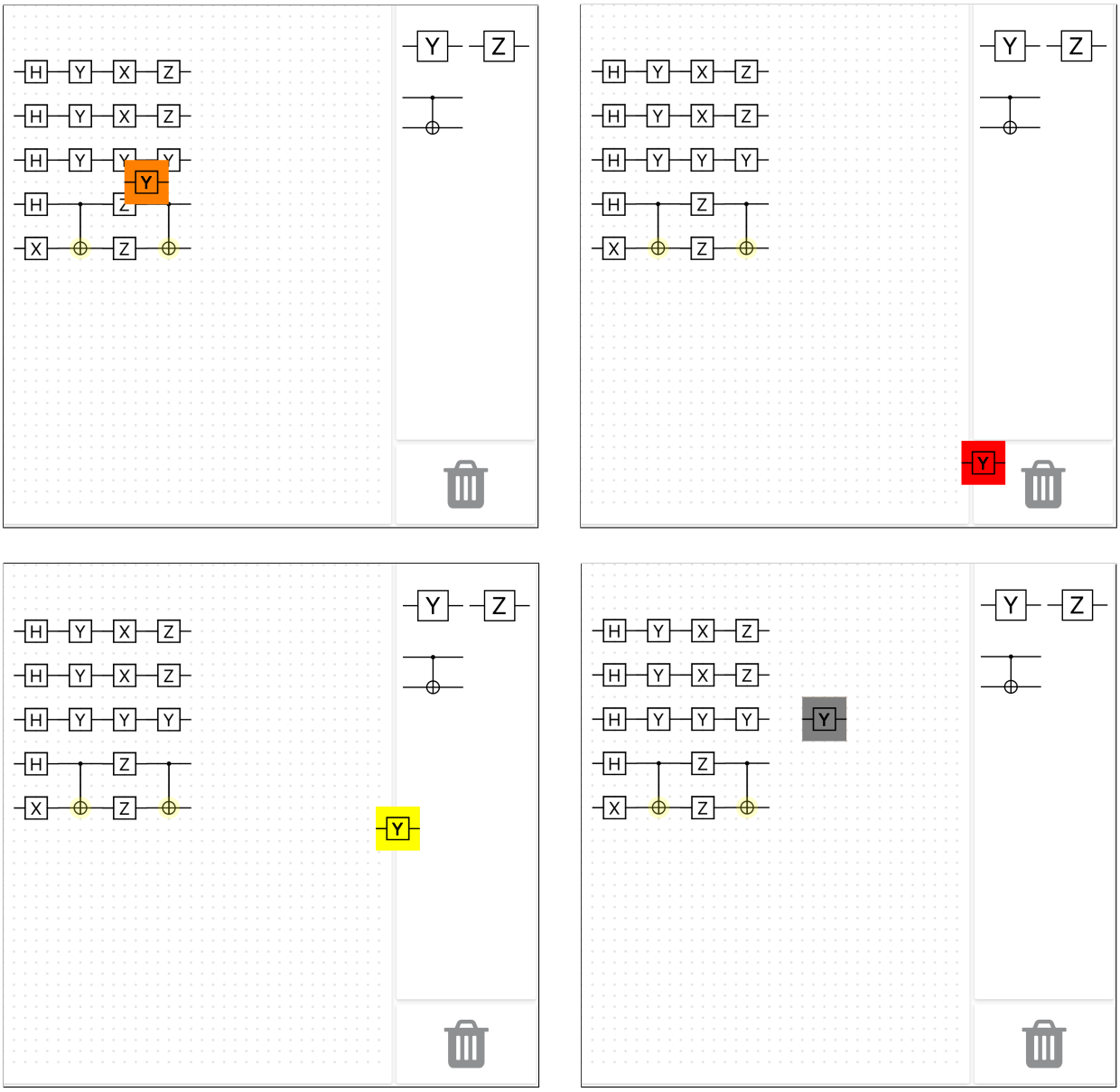

Figura 6.3: Estats de la porta en curs.

D'esquerra a dreta i de dalt a baix: intersecció amb altres portes (taronja), eliminació de porta (vermell), fora del *grid* (groc) i posició estable en mode edició (gris fosc). Tot i que la porta sempre segueix la posició del dit de l'usuari, quan es troba en el *grid* el desplaçament és discret (amb salts) i s'ajusta als punts, mentre que quan es troba fora és continu.

6.2.2 Test de generació de codi

El test més important és el del flux complet de dissenyar un circuit, avaluar-ne les figures de mèrit, exportar el codi i carregar-lo en una plataforma externa. Tenint en compte l'arquitectura del xip d'IBM, s'ha realitzat un circuit de 5 qubits (figura 6.4), s'ha generat el codi QASM (figura 6.5) i s'ha copiat a l'editor QASM de la plataforma d'IBM (figura 6.6).

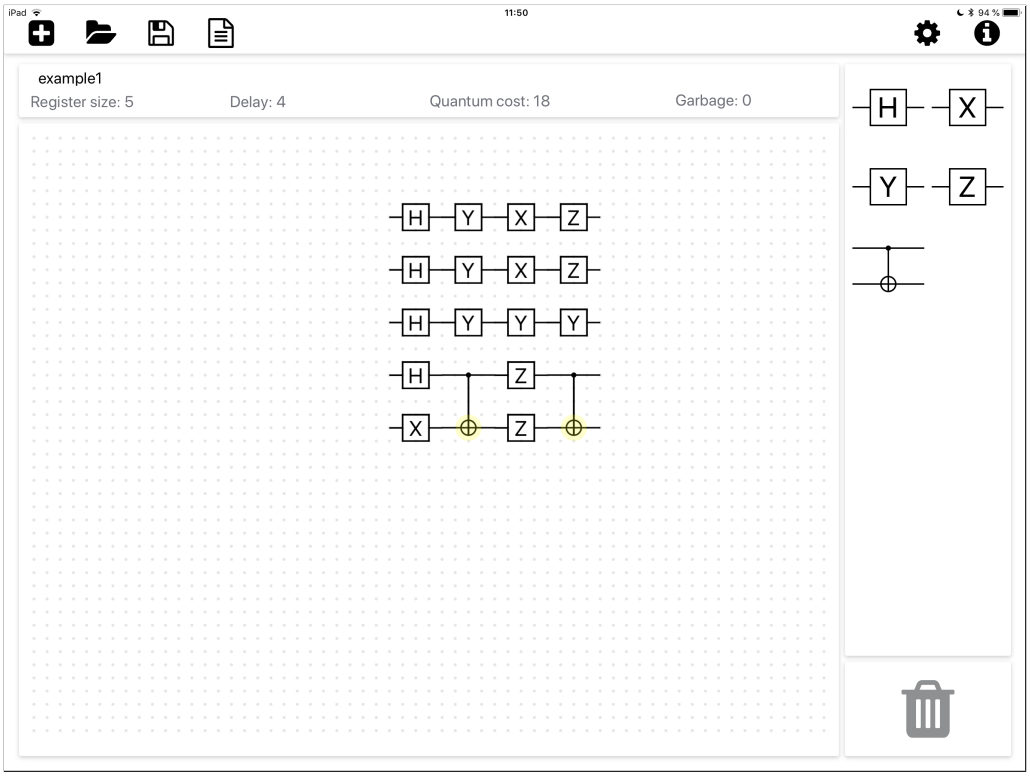

Figura 6.4: Exemple de circuit quàntic.

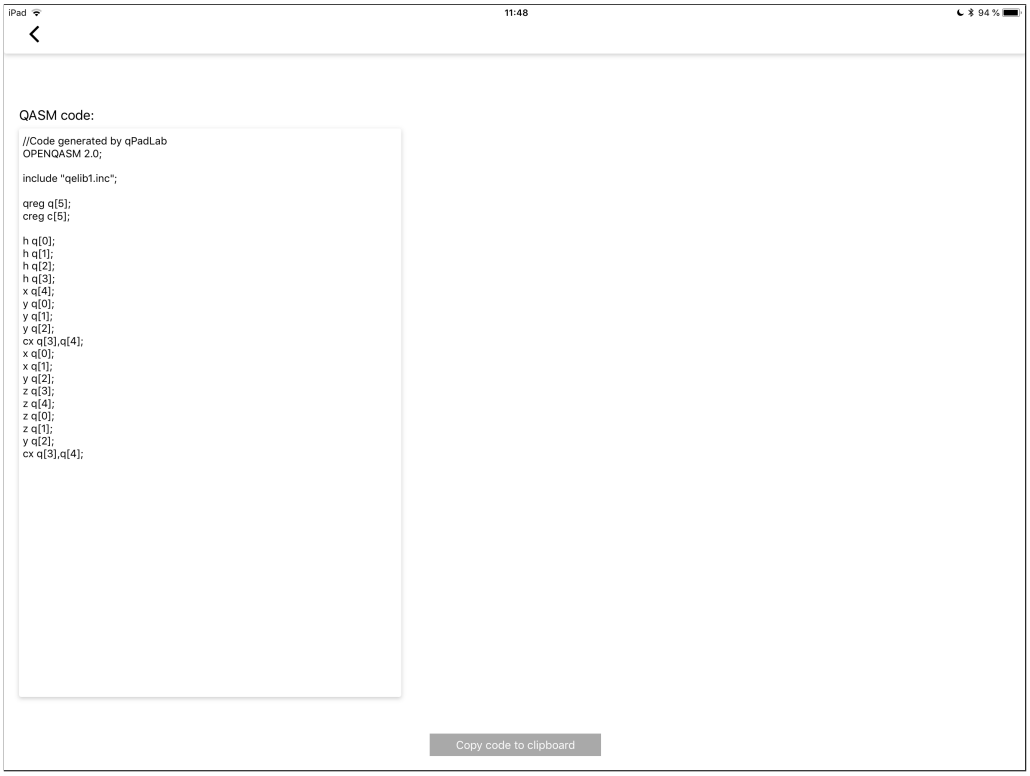

Figura 6.5: Exportació del codi.

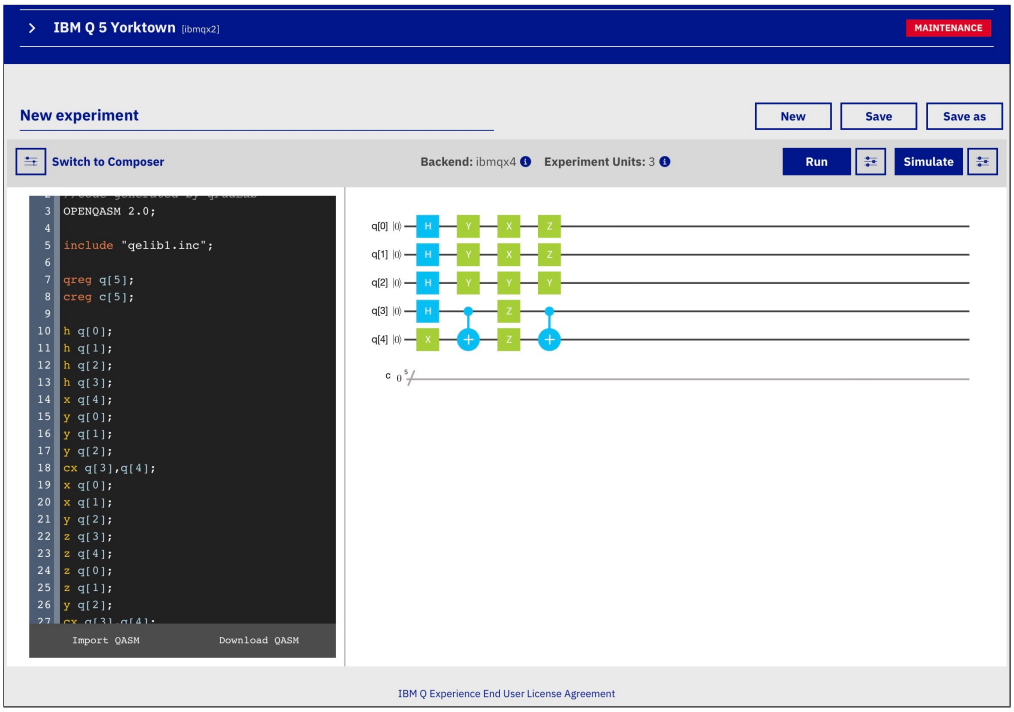

Figura 6.6: Codi carregat a l'IBM Quantum Experience.

Es pot veure que el circuit que es genera en les línies de registre del *composer* d'IBM coincideix amb el circuit dissenyat. Val a dir, però, que la forta dependència amb la topologia del qubits del xip fa que no tots els circuits que es dissenyin siguin compatibles amb la plataforma. Per exemple, si s'afegeix una control-not entre els qubits 2 i 3, en carregar el circuit apareix un missatge d'error (figura 6.7).

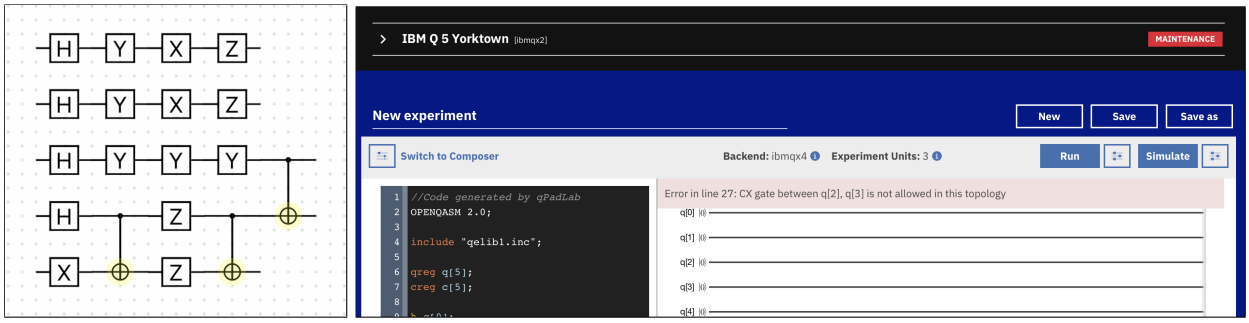

Figura 6.7: Error de compatibilitat.

Això, tot i ser un problema de la implementació pràctica del xip (principalment, de la geometria dels circuits superconductors que el formen), té un impacte directe en l'experiència d'usuari de l'aplicació, ja que la interfície del qPad Lab no imposa restriccions derivades de la implementació d'un xip concret. Aquest problema obre noves vies per seguir millorant l'aplicació.

CAPÍTOL

7

CONCLUSIONS

7.1 Anàlisi crítica

L'aplicació que s'ha desenvolupat permet realitzar circuits quàntics d'una manera àgil i còmoda i les figures de mèrit es calculen en temps real. A més, la funcionalitat d'exportació de codi, tot i que té marge de millora, permet aprofitar els avantatges de simulació i test de dues de les plataformes més importants (el *Quantum Experience* d'IBM i el *Quantum Computing Playground* de Google). Per tant, la valoració global del projecte és positiva, ja que, en termes generals, s'han assolit tots els objectius plantejats (tant pel que fa a la conceptualització com al desenvolupament de l'aplicació) i s'han respectat els terminis especificats en l'etapa de planificació. D'altra banda, el projecte ha suposat un repte constant i això ha permès continuar creixent com a desenvolupador d'aplicacions i també com a investigador en el camp del disseny de circuits quàntics.

No obstant això, hi ha molts punts que es podrien haver millorat tenint en compte les eines i el temps de què es disposava. En primer lloc, s'ha infravalorat la dificultat de desenvolupar una aplicació totalment en Swift. Tot i que la corba d'aprenentatge d'aquest llenguatge no és gaire gran (i més si ja es té experiència en desenvolupament per a iOS), en certs punts del desenvolupament s'han hagut de dedicar hores extres per superar entrebancs relacionats amb elements propis de Swift (per exemple, en l'ús dels *opcionals*). La decisió d'usar Swift ha estat encertada, sens dubte, ja que es tracta del llenguatge actual de desenvolupament per a les plataformes d'Apple, però s'haurien d'haver previst més hores d'aprenentatge en l'estimació d'alguns punts del desenvolupament.

Un altre punt que es podria haver millorat i que està relacionat amb l'anterior és el desenvolupament de la interacció amb les portes. Tenint en compte que és un element central del

projecte i que té un impacte directe en l'experiència d'usuari, aquest desenvolupament s'hauria d'haver separat de la tasca corresponent al desenvolupament de la vista principal, ja que, de fet, està més relacionada amb el model de les portes que amb el controlador en si. S'ha hagut de refer el codi força cops per tenir en compte diverses maneres de resoldre els fluxos definits i s'ha hagut d'incloure lògica de control d'estat als objectes que representen les portes.

Un altre punt que no ha estat ben plantejat és el nombre de vistes a desenvolupar. En la part final del desenvolupament de l'aplicació, s'ha vist que algunes vistes que havien quedat excloses d'aquest projecte eren necessàries perquè l'aplicació fos funcional (com, per exemple, la vista d'exportació de codi per a la funcionalitat d'exportació o la llibreria de cel·les per a la persistència). Com que d'aquestes vistes no se n'havia realitzat el disseny en alta definició ni se n'havien destil·lat les característiques mínimes, s'han hagut d'anar completant sobre la marxa. Això ha provocat que tinguin un estil poc cuidat i un aspecte inacabat.

Finalment, el temps de redacció de la memòria i de creació de vídeos no s'ha dimensionat correctament. Per a ambdues tasques han estat necessàries moltes hores addicionals de creació d'elements gràfics, d'edició dels continguts, de correcció i de millora i cohesió dels capítols de la memòria.

7.2 Línies de futur

Per continuar aquest projecte hi ha diversos camins a seguir. El més obvi i directe és implementar la versió completa de l'aplicació. Això comportaria desenvolupar el model del circuit quàntic (el graf que relaciona les cel·les), actualitzar els gestors per treballar amb circuits i separar la vista principal que s'ha desenvolupat en dues (la de disseny de circuit i la de disseny de cel·la). També caldria completar les vistes secundàries i incloure-hi la biblioteca de cel·les (la que hi ha actualment és, de fet, la biblioteca de circuits). A més, seria interessant continuar explorant la possibilitat d'incloure la persistència externa i l'intercanvi de dissenys entre usuaris.

A banda de completar l'aplicació, hi ha altres vies interessants per continuar amb aquest projecte. A continuació, se n'esmenten algunes:

- **Adaptació automàtica de les plataformes externes**. En la part de test ja s'ha vist la importància de l'arquitectura del *hardware* real a l'hora d'exportar els codis. Coneixent l'arquitectura, seria possible readaptar automàticament els dissenys per fer-los compatibles amb cada plataforma.
- **Connexió directa amb el *back-end* de les plataformes**. Seria interessant —per a aquelles plataformes que ho permetin— implementar la comunicació amb el *back-end* i mostrar els resultats sense sortir de l'aplicació. Aquesta funcionalitat, de fet, ja està suportada en el *Quantum Compiler* (vegeu l'apartat 2.2.3).
- **Implementar algorisme d'optimització de circuits**. Donat un circuit és possible redistribuir les portes i/o modificar-les per, mantenint la funcionalitat, reduir alguna de

les figures de mèrit. Per exemple, segons l'ordre en què s'apliquin les portes, es pot disminuir el retard total de la cel·la (posant-ne algunes en paral·lel). Un altre cas seria la combinació de portes en cascada aplicades sobre un mateix qubit que es poden agrupar en un únic operador. Per exemple, la combinació (Hadamard + Pauli X + Hadamard) sobre un qubit es pot substituir per una única porta (Pauli X). Aquestes millores es poden analitzar de manera sistemàtica i obtenir diversos patrons d'optimització que l'usuari podria aplicar lliurement. També es podria donar l'opció de córrer aquests algorismes d'optimització abans de generar els fitxers de sortida.

- **Implementar mètodes d'interacció addicionals**. Per exemple, selecció múltiple de portes, *zoom* i desplaçament del *grid*, etc.

- **Dissenyar i desenvolupar el tutorial**. Amb una aplicació tan específica i d'un àmbit tan tècnic és tot un repte preparar-ne un tutorial que permeti introduir l'usuari inexpert al funcionament i als objectius de l'aplicació d'una manera senzilla i atractiva. Aquesta línia estaria més enfocada al perfil de dissenyador.

BIBLIOGRAFIA

[1] «IBM Q Devices». A: *IBM* [en línia].
Disponible a: quantumexperience.ng.bluemix.net/qx/devices [Consulta: febrer 2018].

[2] «Rigetti». A: *Rigetti Computing, Berkley, CA & Fremont, CA (USA)* [en línia].
Disponible a: www.rigetti.com/foreste [Consulta: febrer 2018].

[3] «IBM Q Experience». A: *IBM* [en línia].
Disponible a: quantumexperience.ng.bluemix.net/qx/experience [Consulta: febrer 2018].

[4] «QISkit. Quantum Information Software kit»[en línia].
Disponible a: https://www.qiskit.org [Consulta: febrer 2018].

[5] «Quantum playground». A: *Google* [en línia].
Disponible a: www.quantumplayground.net/#/home [Consulta: febrer 2018].

[6] «Language Integrated Quantum Operations: LIQUiD». A: *Microsoft* [en línia].
Disponible a: www.microsoft.com/en-us/research/project/language-integrated-quantum-operations-liqui/ [Consulta: març 2018].

[7] «LIQUiD». A: *GitHub* [en línia].
Disponible a: http://stationq.github.io/Liquid/ [Consulta: març 2018].

[8] «Quantum Circuit Simulator». A: *Google Play.* [en línia].
Disponible a: play.google.com/store/apps/details?id=mert.qcs [Consulta: març 2018].

[9] «Qubit Simulator». A: *Google Play.* [en línia].
Disponible a: play.google.com/store/apps/details?id=com.khazam.qubitsimulator [Consulta: març 2018].

[10] «Logic Simulator Pro». A: *Google Play.* [en línia].
Disponible a: play.google.com/store/apps/details?id=com.KAJORY.Logicsimulatorpro [Consulta: març 2018].

[11] «Quantum Programming Compiler». A: *iTunes* [en línia]. Apple Inc., Infinite Loop, Cupertino (USA).
Disponible a: itunes.apple.com/US/app/id1337663853?mt=8 [Consulta: març 2018].

[12] «Quirk». A: *GitHub, Inc.* [en línia].
Disponible a: github.com/Strilanc/Quirk [Consulta: març 2018].

[13] Boixo, S. *et al.* (2017). Characterizing Quantum Supremacy in Near-Term Devices. *arXiv:quant-ph/1608.00263.*

[14] Nielsen, M; Chuang, I. (2000). *Quantum Computation and Quantum Information.* EE.UU.: Cambridge University Press, 1 edition. ISBN 978-1-107-00217-3.

[15] Hillegass, A.; Conway, J. (2011). *Desarrollo de aplicaciones para iPhone & iPad.* Madrid, Anaya Multimedia. ISBN: 978-8441529328.

[16] Rogers, Y.; Sharp, H.; Preece, J. (2011). *Interaction design.* John Wiley & Sons, Inc. ISBN 978-0-470-66576-3.

[17] Universitat Oberta de Catalunya (2013). «Te presentamos a Jordi y Martina, las "Personas" de la UOC». *El blog de las aulas* [en línia]. Barcelona: Universitat Oberta de Catalunya. aula.blogs.uoc.edu/2013/10/23/et-presentem-al-jordi-i-la-martina-les-persones-de-la-uoc [Consulta: març 2018].

[18] *Societat Catalana de Física* [en línia].
Disponible a: blogs.iec.cat/scfis [Consulta: març 2018].

[19] *Barcelonaqbit* [en línia].
Disponible a: www.barcelonaqbit.com [Consulta: març 2018].

[20] *Balsamiq* [en línia].
Disponible a: balsamiq.com [Consulta: març 2018].

[21] *Material design* [en línia].
Disponible a: material.io/guidelines [Consulta: març 2018].

[22] *Fontastic* [en línia].
Disponible a: fontastic.me [Consulta: març 2018].