Universitat de les Illes Balears

escola politècnica superior

Treball Fi de Grau

GRAU D'ENGINYERIA TELEMÀTICA

Geolocalització interior mitançant el senyal wifi de 3 punts d'accés

ENRIC BLANC FERNÁNDEZ

Tutor
PEDRO JOSÉ PONS BONAFÉ

Escola Politècnica Superior
Universitat de les Illes Balears
Palma, 8 de juliol de 2019

Moltes gràcies a la meva família, professors, amics i totes les persones que m´han ajudat durant aquest grau, ja que sense ells el camí hauría estat molt més difícil. Gràcies també a totes les persones que en cap moment m'han vist capacitat per treure el grau endavant, ja que han estat la meva principal motivació i a qui també va dedicat.

SUMARI

Sumari **iii**

Índex de figures **vii**

Resum **ix**

1 Introducció **1**
1.1 Objectiu . . . 1
1.2 Objectiu del projecte . . . 2
1.3 Fases de realització . . . 2

2 Marc actual **5**
2.1 Posicionament interior . . . 5
2.2 Tecnologies existents . . . 6
2.2.1 Beacons Bluetooth . . . 7
2.2.2 Ultra-wide-band . . . 7
2.2.3 Wifi . . . 8
2.2.4 RFID . . . 8
2.3 Empreses qué ofereixen aquest servei . . . 9
2.4 Per què WIFI? . . . 10

3 Anàlisi i planificació **11**
3.1 Requeriments . . . 11
3.1.1 Requeriments funcionals . . . 11
3.1.2 Requeriments no funcionals . . . 12
3.2 Metodologia de desenvolupament . . . 12
3.3 Seguretat . . . 13
3.4 Fonaments teòrics . . . 13
3.4.1 Model de pèrdues a l'espai lluire . . . 13
3.4.2 Trilateració . . . 14

4 Disseny **17**
4.1 Arquitectura de la aplicació . . . 17
4.2 Tecnologies utilitzades . . . 17
4.2.1 Android Studio . . . 17
4.2.2 Dispositiu de proves . . . 17
4.2.3 Punts d'accés utilitaris . . . 18

4.3 Disseny de pantalles 19

5 Implementació **23**
5.1 Estructura 23
5.2 Wifi Manager 24
5.3 Classes i mètodes 24
5.3.1 Coordinated 24
5.3.2 GridMap 25
5.3.3 ScreenMap 25
5.3.4 LoopMap 26
5.3.5 Image 26
5.3.6 State 26
5.3.7 LviCheckbox 26
5.3.8 Classe ListViewItemViewHolder 26
5.3.9 ListViewItemBaseAdapter 26
5.3.10 Trilateration 26
5.3.11 ScanList 26
5.4 Activitats 27
5.4.1 PresentationActivity 27
5.4.2 ListViewWithCheckboxActivity 27
5.4.3 MainActivity 27
5.5 Layout 27
5.5.1 Layout activity-list-with-checkbox 27
5.5.2 Layout activity$_{l}ist_{w}ith_{c}heckbox_{i}tem$ 27
5.5.3 Layout presentation 27
5.5.4 Layout mapa 27
5.6 Manifest 27
5.7 Problemes durant la implementació 27

6 Proves **51**
6.1 Proves funcionals 51
6.2 Cumpliment de requeriments 52
6.2.1 Requeriments funcionals 52
6.3 Proves usuari final 53
6.3.1 Usuaris menors 18 anys 53
6.3.2 Usuaris adults 54
6.3.3 Usuaris persones majors de 65 anys 54

7 Conclusions **55**
7.1 Precisió del senyal Wifi 55
7.2 Bona gestió del temps 55
7.3 Aplicació 56
7.4 Idees 56

8 Millores aplicables **57**
8.1 La idea principal 57
8.2 Adaptació del projecte 58

8.3 Millores 58
8.4 Treball futur 58

A Annexos **61**
A.1 Manual d'usuari 61
A.1.1 Instal·lació 61
A.1.2 Utilització 62
A.2 Carta alumne a UIB 62

Bibliografia **69**

ÍNDEX DE FIGURES

1.1 Tècnica triangulació a mapa antic . . . 2
1.2 Torre localització estadi . . . 3

2.1 Estadi Camp Nou . . . 6
2.2 Mòbil i transmissor Beacon . . . 7
2.3 Logo senyal wifi . . . 8
2.4 Transponder . . . 9
2.5 Empreses que oferten geolocalització interior . . . 9

3.1 Metodologia model prototipus . . . 13
3.2 Trilateració . . . 15

4.1 Arquitectura aplicació . . . 18
4.2 Logo Android . . . 18
4.3 Samsung GT-19 195 . . . 19
4.4 Punts d'accés utilitzats . . . 19
4.5 Captura de pantalla pantalla presentació . . . 20
4.6 Captura de pantalla selecció AP . . . 20
4.7 Captura de pantalla pantalla principal . . . 21

5.1 Estructura general aplicació . . . 25
5.2 Classe Coordinated . . . 28
5.3 Classe GridMap . . . 29
5.4 Classe GridMap . . . 30
5.5 Classe GridMap . . . 30
5.6 Classe ScreenMap . . . 31
5.7 Classe ScreenMap . . . 32
5.8 Classe LoopMap . . . 32
5.9 Classe LoopMap . . . 33
5.10 Classe LoopMap . . . 34
5.11 Classe Image . . . 35
5.12 Classe State . . . 35
5.13 Classe LviCheckbox . . . 36
5.14 Classe ListViewItemViewHolder . . . 36
5.15 Classe ListViewItemBaseAdapter . . . 37
5.16 Classe ListViewItemBaseAdapter . . . 38
5.17 Classe MainActivity . . . 39

5.18 Classe MainActivity . . . 39
5.19 Classe Trilateració . . . 40
5.20 Classe Trilateració . . . 41
5.21 Classe Trilateració . . . 42
5.22 Classe ScanList . . . 43
5.23 Classe ScanList . . . 43
5.24 Classe PresentationActivity . . . 44
5.25 Classe ListViewWithCheckboxActivity . . . 45
5.26 Classe ListViewWithCheckboxActivity . . . 46
5.27 Classe ListViewWithCheckboxActivity . . . 47
5.28 Layout activity$_{l}ist_{w}ith_{c}heckbox$ . . . 48
5.29 Layout activity$_{l}ist_{w}ith_{c}heckbox_{i}tem$ . . . 48
5.30 Layout presentation . . . 48
5.31 Layout mapa . . . 49
5.32 Manifest . . . 50

6.1 Resultats obtinguts . . . 54

7.1 Esvaïments senyal Wifi . . . 56

8.1 Logo KNX . . . 57

A.1 Captura de pantalla Configuració . . . 63
A.2 Captura de pantalla Configuració amb mode desenvolupador activat . . . 64
A.3 Captura de pantalla Configuració amb mode debug activat . . . 64
A.4 Captura Android Studio . . . 65
A.5 Captura Android Studio finestra dispositius . . . 65
A.6 Captura de pantalla pantalla presentació . . . 66
A.7 Captura de pantalla selecció AP . . . 66
A.8 Captura de pantalla pantalla principal . . . 67

RESUM

Al present document es presenta una aplicació per a la localització interior mitjançant el senyal emès per 3 punts d'accès Wifi coneguts, ja sigui dins una vivenda, dins un centre comercial o qualsevol espai on el sistema GPS no ens pugui ubicar. Sempre coneguent la posisció dels 3 AP i amb detecció de la seva senyal.

L'objectiu d'aquest projecte és aconseguir establir una aplicació que mitjançant tècniques de posicionament, com la triangulació o la trilateració, sigui capaç de representar gràficament la nostra posició a un mapa del recinte, visible per pantalla. Aquesta informació podrà esser transmesa a la instal·lació domòtica de la casa i tractada per a que es realitzin diferents events en funció de la informació rebuda.

S'han analitzat les diferents tecnologíes existents per a la geolocalització interior i hem implementat la primera versió d'un software que ens geolocalitza dins la vivenda.

Tot el software vendrá descrit al document, amb l'explicació del funcionament de cada part i de la seva arquitectura. Finalitza amb les conclusions generals del projecte i futures millores.

CAPÍTOL 1

INTRODUCCIÓ

Desde que l'ésser humà ha tingut seny, la necessitat d'ubicar-se al mon on vivim ha estat un dels principals problemes amb el que vàrem haver de conviure.Varen aparèixer els primers mapes de ciutats que et localitzaven vagament dins el terreny urbà, però al sortir del terreny urbà conegut, ja no teniem cap tipus de referència.

Després del descobriment d'Amèrica i de resoldre la controvèrsia de la forma geomètrica de la terra els mapes varen començar poc a poc a representar fidelment el territori. Gràcies a això, es varen desenvolupar tècniques i formules matemàtiques per descobrir la teva geolocalització al mon amb diversos punts de referència o mètriques. A la figura 6.1 podem observar com es varen començar a representar les tècniques de geolocalització per triangulació.

Ara per ara, la majoria de sistemes de geolocalització es basen en aquestes tècniques arcaiques de geolocalització aplicades a les tecnologies actuals.

1.1 Objectiu

Després de tots els avanços en tecnologia i en telecomunicació, per què no introduir la geolocalització interior?

Avui en día hi ha moltes grans superfícies com grans magatzems, centres comercials, estadis... on es necessari tenir un coneixement de la teva localització dins l'edifici. Per aquesta raó es sol proveïr al visitant mapes per tenir una referència.

Amb la nostre idea d'aplicació podriem prescindir dels mapes físics i informatitzar-los. A més tenriem l'opció de tenir-los en qualsevol idioma, apart d'optimitzar l'espai que solen ocupar els típics mapes torre, com el que podem veure a la figura 1.2.

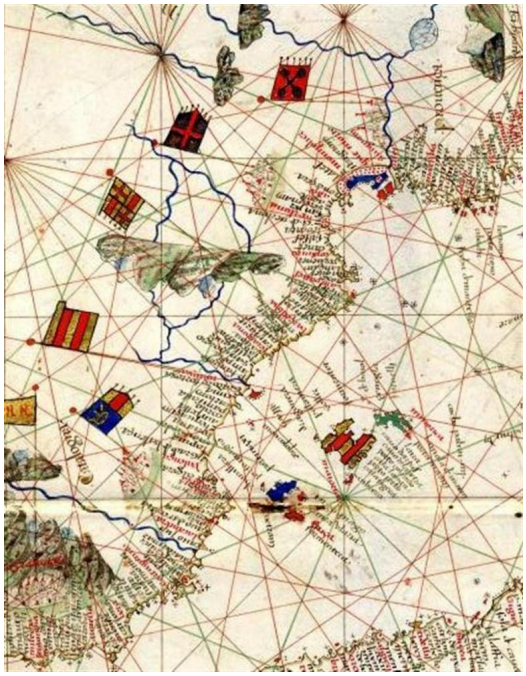

Figura 1.1: Tècnica triangulació a mapa antic

També hi ha vivendes inteligents que per oferir diversos serveis necessiten de la geolocalització dels usuaris dins la vivenda; ja sigui per oferir un servei o per autogestionar el consum.

Per aquest motiu, l'enfocament elegit per a la realització d'aquesta l'aplicació es geolocalitzar a l'usuari. En concret en aquesta primera versió s'han realitzat les proves dins una vivenda.

1.2 Objectiu del projecte

El projecte neix amb l'objectiu de proporcionar a l'usuari una aplicació mòbil que ens geolocalitzi a tot tipus d'espais interiors. A més que sigui capaç d'identificar establiments, habitacions o escenaris que pugui haver dins un recinte i indicar-te quina direcció seguir per arribar a al teu destí.

1.3 Fases de realització

Aquí es dóna una visió general de les fases que s'han seguit al llarg del projecte, que són la majoria de fases que es segueixen durant el desenvolupament d'un software.

- **Estudi**: Investigació sobre les possibles tecnologies a utilitzar, amb un possible estudi de mercat i viabilitat. Indagar i provar aplicacions relacionades, amb l'objectiu d'extreure informació útil per aplicar o ampliar el nostre projecte.

Figura 1.2: Torre localització estadi

- **Anàlisi**: Definició de requisits amb l'objectiu de concretar que ha de realitzar l'aplicació i quins serveis ha d'oferir
- **Investigació**: La contínua cerca de informació sobre el tema i estar informat de totes les notícies científiques ajudarà a millorar al llarg del temps la nostra aplicació i a trobar algoritmes més eficients.
- **Disseny**: Després de realitzar les fases d'estudi i de anàlisi, amb el llistat de requisits ben definit, procedirem a dissenyar l'arquitectura de la nostra aplicació. En el nostre cas inclourem el disseny de la nostra idea principal i l'actual.
- **Implementació**: Programació de l'aplicació mòbil amb Java. En el nostre cas, la fase que més temps ha necessitat.
- **Proves**: Una vegada finalitzades totes les fases anteriors, es necessari la realització d'cfuna fase de proves generals, per comprovar la funcionalitat de la nostra aplicació, l'eficiència i la qualitat del software. Una vegada resoltes les incidències generals que es poden trobar, començarem amb proves més específiques que poden no ser no detectades per l'usuari final, com per exemple ús de memòria.
- **Documentació**: Segons una guia sobre desenvolupament programari, la documentació és una cosa indispensable per al correcte manteniment de qualsevol producte software. Tant a nivell de codi com de document, per tal de deixar constància per escrit de les fases dutes a terme en el cicle de vida del producte software.

- **Manteniment**: El manteniment del producte és una de les fases més importants si volem que el nostre producte pugui ser utilitzat a mesura que passa el temps. Hem de dur a terme el manteniment tant a nivell de codi com a nivell de documentació.

CAPÍTOL 2

MARC ACTUAL

2.1 Posicionament interior

Un sistema de posicionament interior està format per una xarxa de diversos dispositius utilitzats per localitzar sense fils objectes o persones dins d'un edifici. Actualment existeixen diverses empreses que ofereixen aquest servei, cada una d'elles utilitzant una tecnologia diferent obtenint de cada tecnologia els seus avantatges i els seus inconvenients.

Els casos d'ús més habituals on es pot aplicar el posicionament en interiors són a:

- **Magatzems**: Avui en dia podem trobar magatzems tan grans que saber la nostra posició dins el magatzem inclús a on es troba la secció que cercam pot arribar a ser un problema.

- **Aparcaments**: Moltes vegades estacionam el nostre vehicle a aparcaments amb superfícies molt extenses i, a vegades, recordar en quina planta i secció hem estacionat el nostre vehicle ens pot resultar difícil.

- **Museus**: La majoria de museus ofereixen visites individuals autònomes. Ens podem trobar a la situació de que no trobem l'objecte que se'ns està descrivint

- **Xarxes socials**: Seria aplicable també a les xarxes socials, ja que està molt de moda compartir el que feim i on esteim, apurant al máxim el grau de localització. Gràcies a això els altres usuaris de la xarxa social podran reconèixer fins i tot la teva geolocalització interior de la teva ubicació.

- **Campus universitaris**: Una altre aplicació es donaria al arribar a una Universitat desconeguda i que nessecitem arribar a un punt en concret, sense saber tenir idea de on ens trobam i cap a on hem de dirigir-nos.

- **Hospitals**: També és útil per a hospitals. Molta gent arriba a hospitals sense saber on es localitza l'àrea de l'especialitat que necessiten. D'aquesta manera es posicionarien dins l'edifici i podran localitzar a quines parts de l'edifici es localitzen les diferents especialitats.

- **Aeroports**: Per localitzar terminals específiques o zones específiques de aeroports grans, com es el cas de Son Sant Joan a Mallorca.

- **Centres comercials**: En grans centres comercials, moltes vegades trobar on es sitúa una botiga en concret pot resultar un suplici.

- **Recintes esportius**: Quan necessitam arribar a un seient en concret, o a la tribuna reservada, a estadis o recintes esportius és una tecnologia aplicable. Tal i com podem observar a la figura 2.1 saber on es troba la porta d'entrada o el nostre seient pot resultar-nos difícil.

Figura 2.1: Estadi Camp Nou

Segur que és aplicable a molts més camps però només hem llistat els camps d'aplicació més habituals.

2.2 Tecnologies existents

Avui en dia hi ha un gran nombre de empreses que ofereixen serveis de posicionament interior arreu del món. La gran majoria utilizen les tecnologies existents més eficients per realitzar aquesta localització. Tenim moltes formes de localitzar l'objecte, establiment o persona a un edifici, però les tecnologies més utilitzades són bluetooth, UWB , Wifi i RFID

2.2.1 Beacons Bluetooth

La tecnologia dels Beacons [1] que utiliza Bluetooth Low Energy (BLE) és una de les tecnologies que ens dóna la posició amb un marge d'error de 5-6 metres. Es tracta d'una de les tecnologies més eficients ja que el consum de energía es mínim gràcies a l'ús de BLE. Normalment es transmissors s'alimenten amb piles de botó i depenguent de la freqüència i el tipus de senyal tenen una autonomia diferent. Per exemple, una senyal que s'emet de manera poc freqüent, pot tenir una autonomia de 10 anys.

Per contra, hem de tenir una gran quantitat de transmissors Bluetooth per tot el recinte per aconseguir que donin cobertura a totes les àreas del nostre edifici o recinte. A més, necessitem tenir uns emissors, que transmetin paquets broadcast. Aquests paquets s'anomenen beacons.

Els beacons poden transportar informació i actuar de sensor, qualitat que els fa molt interesant per utilitzar-los a segons quins projectes. A la figura 2.2 podem observar quina és la seva forma més habitual.

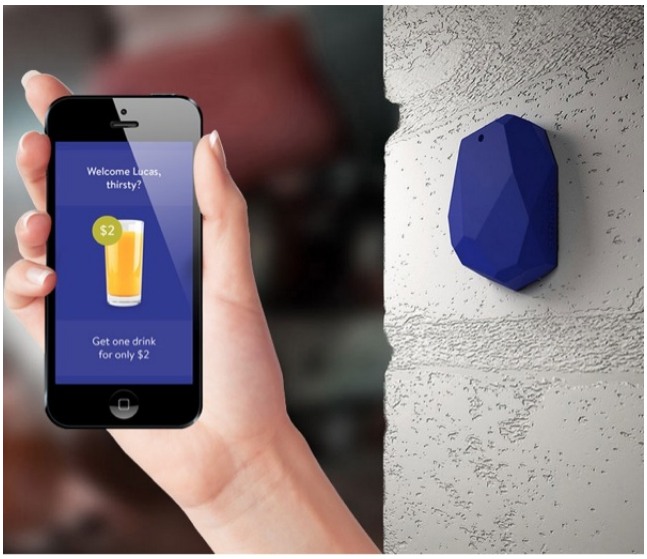

Figura 2.2: Mòbil i transmissor Beacon

Són molt útils per exemple per al seguiment interior. Moltes empreses l'utilitzen per saber quan entren i surten els seus treballadors.

2.2.2 Ultra-wide-band

Ultra-wide-band (UWB) [2] és una tecnologia de ràdio de curt abast que es pot utilitzar per al posicionament en interiors.Utilitza frequències majors de 500 MHz. A diferència de Bluetooth Low Energy i Wi-Fi, el càlcul de la posició es fa a través del temps de trànsit del senyal, enlloc de mesurar les intensitats del senyal. Aquest mètode mesura el temps que tarda en arribar la senyal al receptor.

Per a la localització exacta d'un objecte, s'han de tenir almenys 3 receptors. A més, ha d'haver una línia de visió directa entre el receptor i el transmissor.

L'objecte que s'ha de seguir ha de estar equipat amb una petita etiqueta o tag que funciona amb bateria. Aquest envia dades als nodes de localització.

Els nodes tenen una posició fixa a la infraestructura i com hem dit abans utilitzen el temps de trànsit de llum per calcular la distància de l¡objecte o persona. La combinació de les dades obtengudes per els 3 nodes ens dóna la posició cercada amb un marge d'error només de 10-30cm.

2.2.3 Wifi

El posicionament interior i la navegació interior basada en Wi-Fi s'utilitza a bastants projectes. El motiu d'això és per què pot reutilitzar els diferents punts d'accés Wi-Fi existents. A més, el posicionament aes pot activar quan els usuaris han habilitat la connexió Wi-Fi als seus telèfons intel·ligents. No obstant això, hi ha alguns desavantatges per al posicionament interior amb Wi-Fi el més important es que el marge d'error està entre 5-15 metres.

L'aplicació ha de reconèixer tres punts d'accès WI-FI i mitançant la tècnica de trilateració obtenir la posició de l'usuari.

Es un sistema de localització bastant irregular ja que la senyal WI-FI es pot veure afectada per molts factors externs: Obstacles, temps de latència degut als dispositius conectats a la xarxa que utilitzem, persones...

Figura 2.3: Logo senyal wifi

2.2.4 RFID

La RFID (identificació de radiofreqüència) [3], utilitza ones de ràdio per transmetre sense fil la identitat o altres característiques d'un objecte o persona. Ès una tecnolo-

gia de posicionament emergent que permet el seguiment de la mobilitat d'objectes o persones. Com els receptors necessiten molta proximitat no és adecuada per grans superfícies, si no per a petits projectes a vivendes o localització i seguiment d'objectes. És rentable, fàcil de mantenir i proporciona identificació i ubicació. Això fa que la localització mitjançant RFID sigui especialment adequada per al seguiment de solucions en entorns industrials.
Un sistema de posicionament d'interior de RFID sol contenir transponders, que actuaràn com a receptors, connectats a objectes o persones, i és necessari un lector de transponders, que actuarà com a transmissor. Un clar exemple seria el cas dels supermercats, on s'estàn començant a col·locar a una gran part dels productes.
Els transponders són petites antenes que mitjancant inducció tenen la capacitat de transmetre una senyal a curta distància. La forma típica d'un transponder es pot observa a la figura 2.4.

Figura 2.4: Transponder

2.3 Empreses qué ofereixen aquest servei

Actualment hi ha moltes empreses que han vist una oportunitat per introduïr-se al mercat utilitzant amb un producte que ofereix la geolocalització interior, utilitzant tecnologies descrites anteriorment. Trobam empreses que ofereixen serveis de geolocalització interior amb totes les tecnologies anteriors i d'altres especialitzades en un tipus de geolocalització interior. A la figura 2.5 podem veure algunes de les empreses que tenen aquest servei.

Figura 2.5: Empreses que oferten geolocalització interior

És un mercat que cada vegada creix més i te més demanda, per això moltes empreses existents comencen a oferir serveis de geolocalització.

2.4 Per què WIFI?

Després de analitzar totes les tecnologies existents, es pot veure que la geolocalització mitjançant la intensitat del senyal Wifi és de les més ineficients ja que la senyal és molt variable.

Per el contrari és una de les tecnologies més exteses i que es pot trobar instal·lada per molts d'edificis, raó per la cual utilitzar-la donaia la nostra aplicació molta versatilitat alhora de fer-la arribar a l'usuari final.

En l'aspecte econòmic també té grans avantatges, ja que al ser una tecnologia utilitzada a moltes llars i oficines la infraestructura ja la tendriem. Va ser una de les raons principals per decantar-nos per aquesta tecnologia.
Per tant, una vegada seleccionada la tecnología per els grans beneficis comentats, ens vàrem decantar per realitzar el nostre propi software, ja que així podriem tractar les dades segons les nostres necessitats.

CAPÍTOL 3

ANÀLISI I PLANIFICACIÓ

La fase d'anàlisi i planificació consta de dues parts ben diferenciades. La primera es realitza abans de començar el desenvolupament. És on s'intenta fer la primera traducció de la idea que ha sorgit a una forma més estructurada. Aquesta fase, a la vegada, plasma els avenços que es van aconseguint i es van aportant més detalls als requeriments. D'aquesta forma se li dóna un millor enfocament al projecte i s'obté un millor resultat final.

Un cop es comença el desenvolupament del producte, la fase d'anàlisi i planificació es converteixen en un procediment que cal realitzar de forma constant. En començar a escriure codi poden sorgir nous dubtes, problemes, o qualsevol imprevist que requereixi tornar a realitzar la fase de anàlisi i planificació.

3.1 Requeriments

Una vegada plantejada la idea hem de donar-li forma mitjançant la redacció dels requeriments de l'aplicació.

Alhora de que els nostres treballadors, o la persona encarregada de realitzar el projecte, l'hagi de dur a terme, una de les primeres coses que farà serà llegir la llista de requeriments de l'aplicació. Per això és necessari redactar una llista de requeriments de l'aplicació. A continuació tenim el llistat de requeriments

3.1.1 Requeriments funcionals

Els requeriments funcionals descriuran totes les interaccions que tendran els usuaris amb l'aplicació.

- **Requeriment funcional 1**: L'usuari ha de poder identificar la seva posició mitjançant un sistema de navegació intern.

- **Requeriment funcional 2**: L'usuari ha de poder seleccionar sobre quins 3 punts d'accés es realitzarán els càlculs de posicionament.
- **Requeriment funcional 3**: L'usuari ha de ser notificat en qualsevol moment si la intensitat de la senyal wifi és baixa o no és suficient.
- **Requeriment funcional 4**: L'usuari ha de identificar per a que serveix i que es realitza en cada una de les pantalles de l'aplicació
- **Requeriment funcional 4**: L'usuari ha de poder tornar a seleccionar els seus punts d'accés en cas d'error al seleccionar-los.

3.1.2 Requeriments no funcionals

Els requeriments no funcionals representen característiques generals i restriccions de l'aplicació o sistema que s'estigui desenvolupant.

- **Requeriment no funcional 1**: No ha de ser necessari estar connectat a cap xarxa Wifi per a que l'aplicació pugui funcionar.
- **Requeriment no funcional 2**: L'aplicació ha de estar realitzada amb l'entorn de programació Android Studio
- **Requeriment no funcional 3**: L'aplicació ha de ser compatible amb tots els sistemes Android fins a la seva darrera actualització 9.0.
- **Requeriment no funcional 4**: El programador introduirà el posicionament dels 3 punts d'accés per programa. En cas de canvi de posició d'aquests es notificarà per entregar nova versió al client o canviar l'actual programa.
- **Requeriment no funcional 5**: El software s'ha de actualitzar de manera automàtica

3.2 Metodologia de desenvolupament

S'ha optat per elegir una metodologia de prototip [4], ja que per les necessitats de treball hem pensat que era la més indicada.

El Model de prototips pertany als models de desenvolupament evolutiu. El prototip ha de ser construït en poc temps, fent servir els programes adequats i no s'han d'utilitzar molts recursos.

El disseny ràpid es centra en una representació d'aquells aspectes del programari que seran visibles per al client o l'usuari final. Aquest disseny condueix a la construcció d'un prototip. Gràcies als resultats obtinguts es refinen els requisits del programari que es desenvoluparà. La interacció ocorre quan el prototip s'ajusta per satisfer les funcionalitats que ha d'oferir.

Consta de diferents etapes les quals s'han seguit per a la realització del projecte:

- **Comunicació**
- **Plantejament ràpid**
- **Modelat i disseny ràpid**
- **Construcció del prototipus**
- **Desenvolupament, entrega i retroalimentació**
- **Entrega final**

A la figura 3.1 podem observar de manera gràfica com és la metodologia utilitzada:

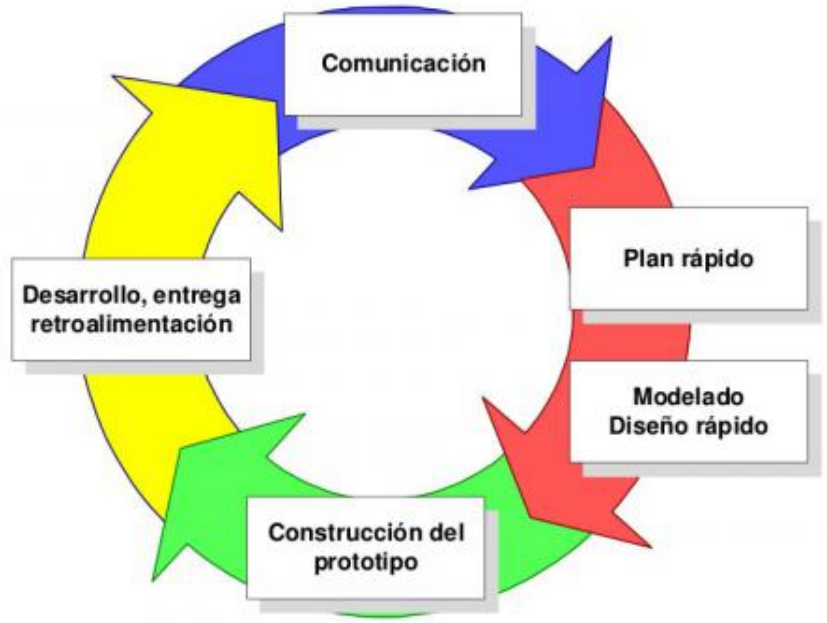

Figura 3.1: Metodologia model prototipus

3.3 Seguretat

L'aplicació no requereix mesures d'un alt nivell de seguretat ja que interactúa només internament. Simplement asegurar-se de que al moment de la descàrrega, l'arxiu no ha estat modificat i és el correcte.

3.4 Fonaments teòrics

3.4.1 Model de pèrdues a l'espai lluire

Per calcular les distàncies [5] segons la intensitat del senyal rebut hem utilitat la fórmula del model de pèrdues a l'espai lliure. Fórmula amb la que obtenim Db, per tant haurem de tractar els Dbm que obtenim de la potència del wifi. La formula resultant és

aquesta:

$$d = 10^{\frac{dB-(20log(f)+k)}{20}} \tag{3.1}$$

En el nostre cas k = 92,45 i la frequència 2,4

3.4.2 Trilateració

La trilateració [6] és un mètode matemàtic per determinar les posicions relatives d'objectes usant la geometria de triangles de forma anàloga a la triangulació. A diferència d'aquesta, que fa servir mesures d'angle (juntament amb almenys una distància coneguda per calcular la localització del subjecte), la trilateració usa les localitzacions conegudes de dos o més punts de referència, i la distància mesurada entre el subjecte i cada punt de referència. Per a determinar de forma única i precisa la localització relativa d'un punt en un pla bidimensional usant només trilateració, es necessiten generalment almenys 3 punts de referència.

Pot trobar-se una derivació de la solució de problema de trilateració tridimensional prenent les fórmules de tres esferes i igualant-. Per fer això, hem d'aplicar tres limitacions als centres d'aquestes esferes; totes han d'estar en el pla z = 0, una altra ha d'estar en l'origen, i una tercera en l'eix x.

Comencem amb tres esferes,

$$r1^2 = x^2 + y^2 + z^2 \tag{3.2}$$

$$r2^2 = (x-d)^2 + y^2 + z^2 \tag{3.3}$$

$$r3^2 = (x-i)^2 + (y-j)^2 + z^2 \tag{3.4}$$

Restem la segona a la primera i resolem per x:

$$x = \frac{r1^2 - r2^2 + d^2}{2d} \tag{3.5}$$

En substituir això en la fórmula de la primera esfera es produeix la fórmula d'un cercle, la solució a la intersecció de les dues primeres esferes:

$$y^2 + z^2 = r1 - \frac{(r1^2 - r2^2 + d^2)^2}{4d^2} \tag{3.6}$$

Igualant aquesta fórmula a la fórmula de la tercera esfera, tenim:

$$y = \frac{r1^2 - r3^2 - x^2 + (x-i)^2 + j^2}{2j} = \frac{r1^2 - r3^2 + i^2 + j^2}{2j} - \frac{i}{j}x \tag{3.7}$$

Ara que tenim les coordenades x i y del punt solució, podem simplement aclarir z de la fórmula de la primera esfera:

$$z = \sqrt{r1^2 - x^2 - y^2} \tag{3.8}$$

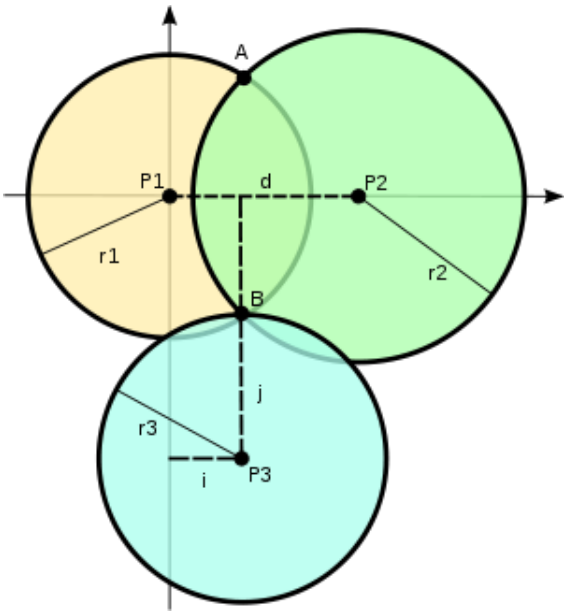

Figura 3.2: Trilateració

CAPÍTOL 4

DISSENY

4.1 Arquitectura de la aplicació

En aquest cas l'arquitectura de l'aplicació és molt simple ja que només hi ha un programari instal·lat a un mòbil que interactúa amb altres elements de l'entorn. Aquest elements de l'entorn són els 3 punts d'accés wifi dels que ha de rebre la senyal. A la figura 4.1 podem observar l'arquitectura típica dins un espai interior. Amb la configuració correcta del posicionament dels 3 punts d'accés Wifi i un mòbil que estigui dins la zona de cobertura, l'aplicació ja ha de ser capaç de calcular la posisció de l'usuari dins l'edifici.

4.2 Tecnologies utilitzades

Durant la fase d'anàlisi s'ha de fer una recerca de totes les tecnologies actuals disponibles per a realitzar el projecte. Després de realitzar aquesta investigació i degut a la familiarització que es te amb diverses tecnologies. Les tecnologies que es veuen a continuació han estat les utilitzades.

4.2.1 Android Studio

Ha estat l'entron de desenvolupament elegit ja que es tracta d'un codi obert amb les llibreries necessàries per al desenvolupament del projecte. El llenguatge de programació es Java i totes les explicacions i codi que ofereix Android han simplificat molt la tasca d'implementació. A la figura 4.2 podem veure el logo actual d'aquesta eina.

4.2.2 Dispositiu de proves

Ha estat l'eina on hem anat realitzant les diverses proves. Es tracta d'un samsung que utilitza el sistema operatiu Android 4.4.2 i amb el mode programador activat per poder instalar aplicacions pròpies.A la figura 4.3 podem veure l'aspecte del model de mòbil

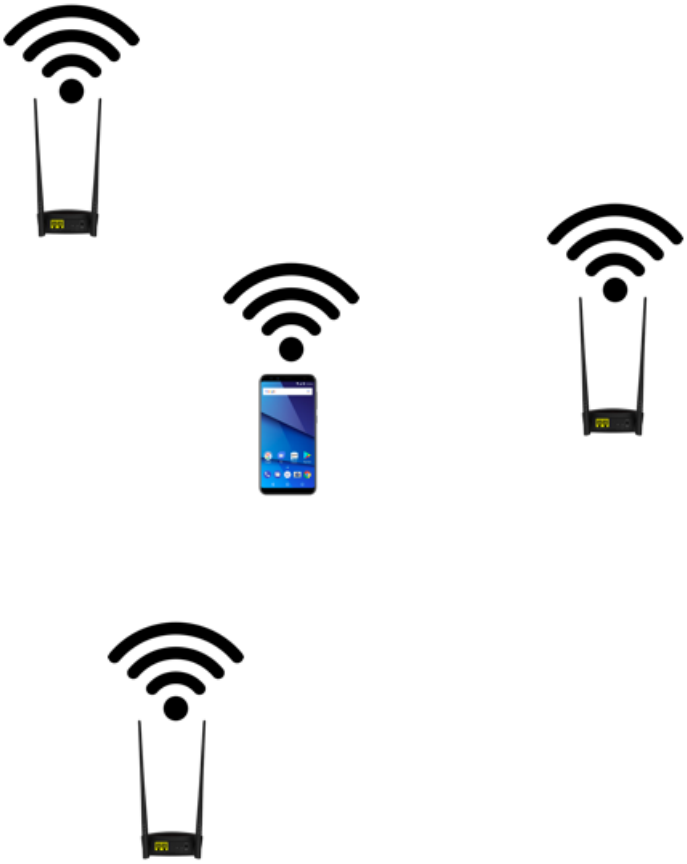

Figura 4.1: Arquitectura aplicació

Figura 4.2: Logo Android

utilitzat. Durant el projecte hem estat conscients de que es una versió de Android un poc anticuada, però hem utilitzat els permisos adecuats perque funcioni a les versions actuals.

4.2.3 Punts d'accés utilitaris

Ha estat necessari reutilitzar els punts d'accès, en el meu cas, els que tenía al lloc on realitzava les proves. Es tracta d'un router Lowi i un repetidor D-link. El tercer punt d'accés ha estat l'ordenador utilitzat en mode compartir internet. No és el més correcte

Figura 4.3: Samsung GT-19 195

però aquests han estat els elements dels que he disposat. Sempre hauria estat millor amb tres punts d'accés iguals. A la figura 4.4trobarem els punts d'accés utilitzats.

Figura 4.4: Punts d'accés utilitzats

4.3 Disseny de pantalles

L'aplicació consta de 3 pantalles principals: pantalla de presentació que podem veure a la figura 4.5, pantallde de selecció dels 3 punts d'accés, que podem veure a la figura 4.6 i pantalla principal de posicionament, que podem veure a la figura 4.7

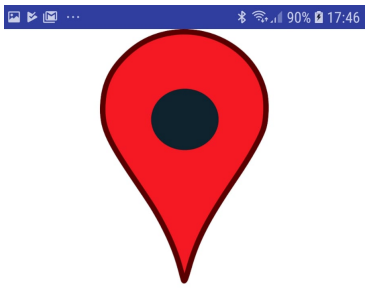

Figura 4.5: Captura de pantalla pantalla presentació

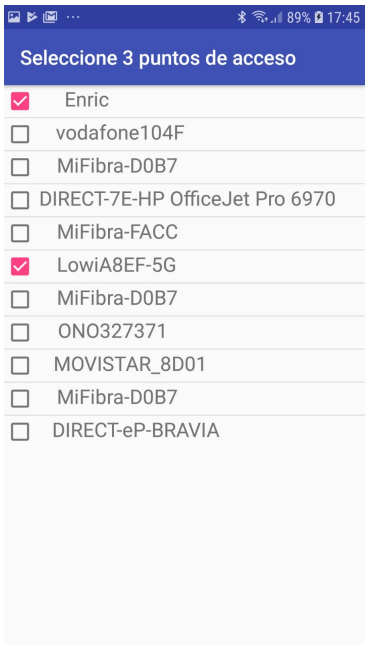

Figura 4.6: Captura de pantalla selecció AP

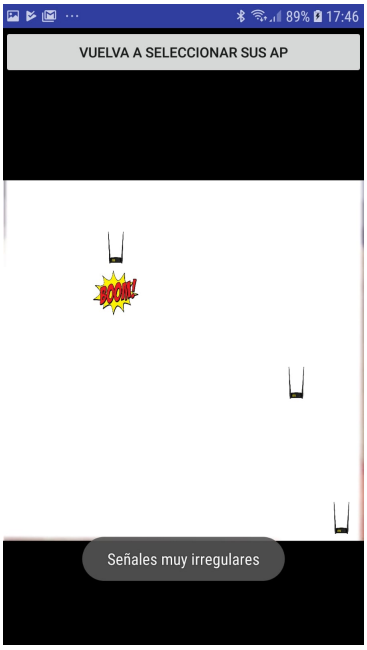

Figura 4.7: Captura de pantalla pantalla principal

CAPÍTOL 5

IMPLEMENTACIÓ

L'implementació del projecte ha estat la fase en que més temps hem invertit, a més de ser una de les més importants. A contínuació veurem tots els punts relacionats amb la implementació del projecte. Totes les figures referenciades es trobaran al final del capítol Implementació.

5.1 Estructura

L'estructura de l'aplicació es podria dividir en 5 parts importants, cada part s'encarrega de realitzar una funció diferent i es cridada per programa sempre sigui necessari.

- **Presentació**: És l'activitat encarregada de presentar l'aplicació. Cada vegada que iniciam el programa ens apareixerà aquesta activitat. Simplement es tracta d'una classe que millora estèticament l'aplicació i una manera d'introduir l'aplicació a l'usuari.

- **Selecció AP**: És la segona activitat que ens apareix per pantalla. Es l'activitat predecessora de l'activitat presentació. En aquesta activitat es presenta un llistat de punts d'accés Wifi, on l'usuari, ha de seleccionar els tres punts d'accés que utilitzará la classe trilateració per calcular el nostre posicionament.

- **Main Activity**: És la classe pricnipal i la que s'encarrega de representar per pantalla la posició de l'usuari a una pantalla estructurada en forma de graella, que serà el nostre mapa. Amb l'ajuda de les classes Graella, Loop, ScanList o diversos mètodes com per exemple tots els de pintat aquesta activitat és capaç de representar i situar a l'usuari.

- **Graella**: Gràcies a l'objecte Coordinada hem estat capaços de una graella que utilitzarem com a mapa. Aquesta graella estará formada per caselles, que tendran unes coordinades per poder identificar-les a cada una d'elles al mapa.

- **ScanList**: Es tracta de la classe encarregada de realitzar un escàner de tots els punts d'accés wifi i obtenir la informació necessària de cada un d'ells per posteriorment poder tractar la informació.
- **Mètodes pinta**: La graella i el loop utilitzen mètodes de pintat, ja que volem representar imatges i aquestes imatges poden anar canviant depenent de la nostra posició.
- **Loop**: És el thread que en segon plà ens actualitzarà la nostra posició i recarregarà les imatges actualitzades.
- **Trilateració**: Classe encarregada de realitzar les operacions pertinents per trobar los coordenades x i y mitjançant la tècnica de trilateració.
- **Graella**: La classe loop és l'encarregada d'actualitzar el contingut de la nostra aplicació.

A la figura 5.1 podem observar gràficament l'estructura comentada anteriorment

5.2 Wifi Manager

És una de les llibreries més importants de Android que s'ha utilitzat al programa. Gràcies a aquesta llibreria hem pogut obtenir tota la informació necessària a tractar dels punts d'accès wifi. Ja sigui potència rebuda o SSIDs entre d'altres més opcions que ens ofereix. Ens permet realitzar accions a:

- **Llistes de xarxes configurades**: la llista es pot visualitzar i actualitzar i es poden modificar els atributs de les entrades individuals.
- **La xarxa wifi actual**: ens permet obtenir informació de la xarxa Wi-Fi activa actual, si n'hi ha. La connectivitat es pot establir o descompondre i es pot consultar informació dinàmica sobre l'estat de la xarxa.
- **Entorn**: Escanejar i obtenir resultats de les exploracions de l'entorn per trobar els punts d'accés dels que tengui visibilitat. Aquests resultats contenen informació suficient per prendre decisions sobre el punt d'accés al qual connectar-se, SSIDs, tipus de seguretat utilitzada...

5.3 Classes i mètodes

[7]

5.3.1 Coordinated

És l'objecte base creat per a poder utilitzar una graella. Creant l'objecte Coordinated es podrà tenir control sobre totes les caselles de la nostra graella. Com podem veure té els getters i setters necessaris. A la figura 5.2 podem veure el codi de la classe Coordinated.

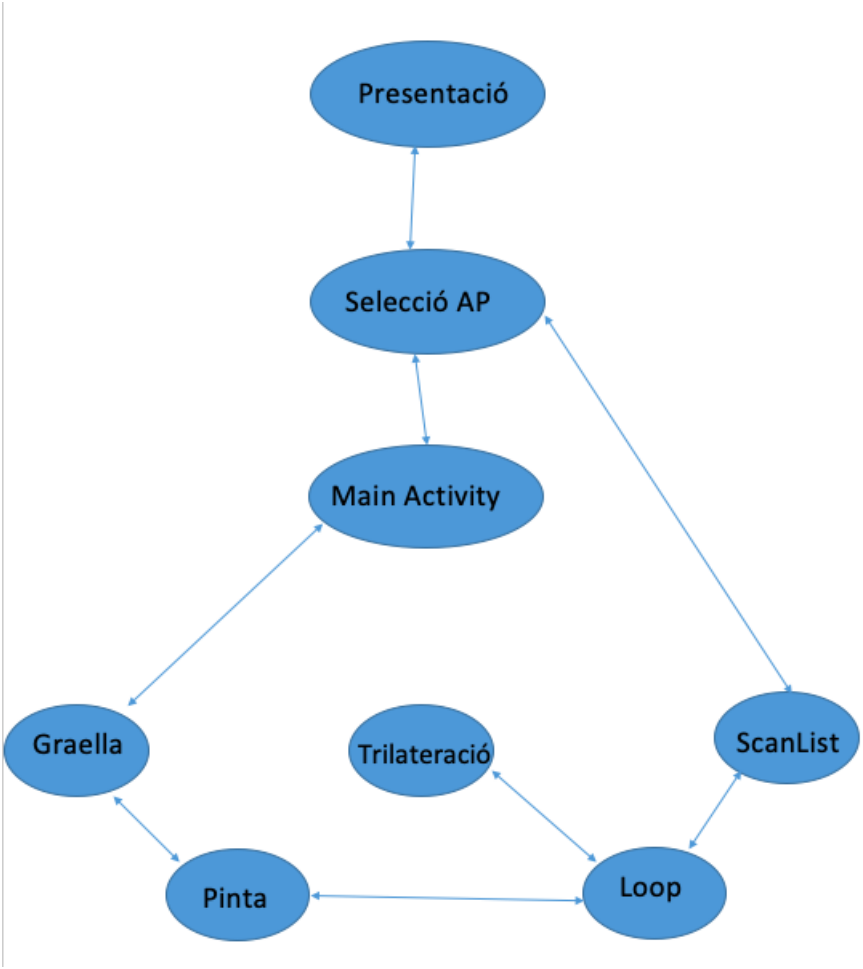

Figura 5.1: Estructura general aplicació

5.3.2 GridMap

Es la classe encarregada de crear la nostra graella amb la que farem feina amb el constructor corresponent. Crea la graella amb un doble blucle. Conté un mètode per carregar totes les imatges necessàries, carregaImatge(). També ens permet canviar la dimensió de les caselles amb el mètode dimensions(). Per últim, el mètode pinta () per anar pintant les imatges de l'aplicació tractant les possibles imatges que ens podrien generar una excepció. A les figurues 5.3 5.4 5.5 podem veure el codi de la classe.

5.3.3 ScreenMap

Mapa amb totes les variables necessàries per construir una graella i un loop. Aquí es defineix com serà la graella i se li atribueixen les característiques. Per exemple cuantes columnes i files tendrà. Creant un objecte loopThread es controla per darrere el temps de pintat de la pantalla. Conté mètodes de pintat de l'objecte ScreenMap i un mètode per inicar la graella. A les figurues 5.6 5.7 podem veure el codi de la classe.

5.3.4 LoopMap

Es el thread que s'estarà executant cada 0,1 segons actualitzant el que veu l'usuari i els càlculs. En aquest interval de temps es realitzaràn els càlculs de la distància actual de l'usuari als tres punts d'accès, també es calcularà la nostra posició actual i es pintarà la nostra posició per pantalla. A les figurues 5.8 5.9 5.10 podem veure el codi de la classe.

5.3.5 Image

Classe per gestionar totes les imatges de la nostra aplicació. A la figura 5.11 es pot veure el codi de la classe Image.

5.3.6 State

Es una classe enumeration pensada per afegir tots els estats futurs que pugui tenir una casella, ja que el més probable es que en un futur l'aplicació evolucioni. A la figura 5.12 podem observar el seu codi

5.3.7 LviCheckbox

LviCheckbox és l'objecte que representa un çheckbox"per a què al moment d'elegir les xarxes i clicar sobre cada una d'elles, es vegi clarament quines hem triat. Estan formades per un booleà i un text que les acompanya amb els getters i setters corresponents. A la figura 5.14es pot observar el codi de la classe.

5.3.8 Classe ListViewItemViewHolder

És una classe que representa un l'objecte que es necessari per a poder representar la llista de items. Amb els seus getters i setter corresponents. A la figura 5.14 podem observar el seu codi amb tots els mètodes corresponents.

5.3.9 ListViewItemBaseAdapter

Classe que ens representa la llista amb els checkbox. Conté els contructors necessaris i els getters que es necessitaran. A les figures 5.15 5.16

5.3.10 Trilateration

És la classe encarregada de calcular la nostra posició retornant les coordinades x y de la casella en la que ens trobam. A les figures 5.19 5.20 5.21podem veure el codi de la classe.

5.3.11 ScanList

És la classe que ens permet obtenir totes les dades dels punts d'accés que hi ha a l'entorn. Amb el mètode ScanWifi() realitzam aquest escàner per analitzar tots els punts d'accés que hi ha a l'entorn. També tenim un mètode getListRedy()que ens permet saber quan està l'escànner finalitzat, retornant un booleà. A les figures 5.22 5.23podem veure el codi de la classe

5.4 Activitats

5.4.1 PresentationActivity

Es la primera activitat que apareix al executar l'aplicació. Ens introdueix l'aplicació en un periode de temps de 3 segons. A la figura 5.24 podem veure el codi.

5.4.2 ListViewWithCheckboxActivity

És l'activitat que ens permetrà visualitzar per pantalla les opcions de selecció de tots els punts d'accés trobats. A la figura 5.25 5.26 5.27podem veure el codi.

5.4.3 MainActivity

És l'activitat principal, activitat que ens crea la pantalla principal, on l'usuari pasarà la major part del temps. Tractam el botó btnNove per tornar a seleccionar els tres punts d'accés.A les figures 5.17 5.18 podem veure el codi de la classe.

5.5 Layout

A continuació veurem tots els codis XML dels layouts de la nostra aplicació.

5.5.1 Layout activity-list-with-checkbox

A la figura 5.28 podem observar el codi del layout activity-list-with-checkbox.

5.5.2 Layout activity$_list_with_checkbox_item$

A la figura 5.29 podem apreciar el codi del layout activity-list-with-checkbox-item.

5.5.3 Layout presentation

A la figura 5.31 podemt veure el codi del layout presentation.

5.5.4 Layout mapa

A la figura 5.31 podem observar el codi del layaout mapa.

5.6 Manifest

El codi del manifest de l'aplicació el podrem trobar a la figura 5.32

5.7 Problemes durant la implementació

El problema més habitual i més perjudicial per al desenvolupament del projecte amb el que ens hem trobat durant la fase de implementació ha estat la procastinació de problemes. Amb un marge tan ampli de temps, a simple vista, hem pogut comprovar que deixar a resoldre problemes per al final no era la millor solució.

```java
package tfg.tfgv2;

/**
 * Created by mac on 21/03/2019.
 */

public class Coordinated {
    private int x;
    private int y;
    private State state;

    //constructor
    public Coordinated(int x, int y) {
        this.x = x;
        this.y = y;
        this.state = State.BUIT;
    }

    public Coordinated() {
    }
    //getters i setters

    public int getX() { return x; }

    public int getY() { return y; }

    public void setState(State state) { this.state = state; }


}
```

Figura 5.2: Classe Coordinated

```java
package tfg.tfgv2;

import android.app.Activity;
import android.graphics.Canvas;
import android.widget.Toast;

/**
 * Classe per gestionar les dades del mapa i de les caselles
 */
public class GridMap {

    private int files; //files i columnes que té la graella (que es veu en pantalla)
    private int columnes;
    private int dimCaselles; //dimensió de cada casella
    private int offsetX; //desplaçaments perquè quedi centrada la graella
    private int offsetY;
    private Coordinated casella[][]; //array que conté totes les caselles
    private Image[] celles = new Image[3]; //array que conté les imatges per les caselles
    private ScreenMap joc;
    private Activity activity;

    //Constructor graella
    public GridMap(ScreenMap joc, int files, int columnes, int maxModelX, int maxModelY, Activity
            activity) {
        this.joc = joc;
        casella = new Coordinated[files + 2][columnes + 2]; //es posen dues columnes i files més per
        // fer un quadre i control·lar els marges de la graella
        this.files = files;
        this.columnes = columnes;
        this.activity = activity;
        dimCaselles = this.dimensions(maxModelX, maxModelY, files, columnes);
        offsetX = (maxModelX - (columnes * dimCaselles)) / 2;
        offsetY = (maxModelY - (files * dimCaselles)) / 2;
```

Figura 5.3: Classe GridMap

```
32          offsetX = (maxModelX - (columnes * dimCaselles)) / 2;
33          offsetY = (maxModelY - (files * dimCaselles)) / 2;
34          //for per crear les caselles
35          for (int i = 0; i < files +2; i++) {
36              for (int j = 0; j < columnes + 2; j++) {
37                  if ((i == 0 || j == 0) || (i == files + 1 || j == columnes + 1)) {
38                      casella[i][j] = new Coordinated( x: 0,  y: 0);
39                      casella[i][j].setState(State.BUIT);//les caselles del marge queden ja destapades
40                  } else {
41                      casella[i][j] = new Coordinated( x: ((j - 1) * dimCaselles) + offsetX,  y: ((i -
42                              1) * dimCaselles) + offsetY);
43                  }
44              }
45          }
46          carregaImatge();
47      }
48
49
50      //getter de casella
51
52      public Coordinated getCoordinades(int x, int y) { return casella[x][y]; }
55
56
57      //mètode on calculam les dimensions de les caselles
58  @   private int dimensions(int maxModelX, int maxModelY, int files, int columnes) {
59          int x = maxModelX / columnes;
60          int y = maxModelY / files;
61          return x < y ? x : y;
62      }
63
```

Figura 5.4: Classe GridMap

```
64      //mètode per carregar totes les imatges del joc
65      private void carregaImatge() {
66          //carga tores les imatges
67          for (int i = 0; i < celles.length; i++) {
68              celles[i] = new Image(joc,  dib: R.drawable.cell_0 + i);
69          }
70
71      }
72
73      //mètode perque les nostres imatges apareguin quan s'esdevè una acció del joc
74      public void pinta(Canvas canvas, Coordinated coordinated) {
75          celles[0].pinta(canvas, offsetX, offsetY,  w: dimCaselles * columnes,  h: dimCaselles *
76                  files); //imatge de fons
77          celles[0].setAlpha(170);
78          celles[2].pinta(canvas,casella[3][3].getX(),casella[2][2].getY(),dimCaselles,dimCaselles);
79          celles[0].setAlpha(125);
80          celles[2].pinta(canvas,casella[5][7].getX(),casella[5][7].getY(),dimCaselles,dimCaselles);
81          celles[0].setAlpha(125);
82          celles[2].pinta(canvas,casella[8][8].getX(),casella[8][8].getY(),dimCaselles,dimCaselles);
83          celles[0].setAlpha(125);
84
85          if( coordinated.getX()<9 && coordinated.getX()>-1 && coordinated.getY()<9 &&
86                  coordinated.getY()>-1) {
87              celles[1].pinta(canvas, casella[coordinated.getX()][coordinated.getY()].getX(),
88                      casella[coordinated.getX()][coordinated.getY()].getY(), dimCaselles,
89                      dimCaselles);
90              celles[1].setAlpha(125);
91          }
92          else{
93              Toast.makeText(activity,  text: "Señales muy irregulares", Toast.LENGTH_SHORT).show();
94          }
95      }
```

Figura 5.5: Classe GridMap

```java
package tfg.tfgv2;

import android.app.Activity;
import android.content.Context;
import android.graphics.Canvas;
import android.view.SurfaceHolder;
import android.view.SurfaceView;

/**
 * Pantalla del mapa
 */
public class ScreenMap extends SurfaceView {
    GridMap graella;
    Image img, img2;
    Activity activity;
    private Context context;
    private int files=8, columnes=8;
    private SurfaceHolder holder;    // per gestionar el pintat de pantalla
    private LoopMapa MapaLoopThread;   // controla el temps de dibuixat a pantalla

    //constructor
    public ScreenMap(Activity activity, Context cont) {
        super(cont);
        this.context = cont;
        this.activity = activity;


        MapaLoopThread = new LoopMapa(activity,  view: this);
        holder = getHolder();
        holder.addCallback(new SurfaceHolder.Callback() {
            public void surfaceCreated(SurfaceHolder arg0) {

                MapaLoopThread.setRunning(true);
                if (!MapaLoopThread.isAlive()) {
                    iniciaMapa();
                    MapaLoopThread.start();

                }
            }

            public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {
            }

            public void surfaceDestroyed(SurfaceHolder arg0) { MapaLoopThread.setRunning(false); }
        });
    }
```

Figura 5.6: Classe ScreenMap

```
50      //iniciam la graella
51      public void iniciaMapa() {
52          // Inicialització de totes les variables necessàries
53          graella = new GridMap( joc: this, files, columnes, getWidth(), getHeight(),activity);
54          img = new Image( mapa: this, R.drawable.cell_2);
55          img2 = new Image( mapa: this, R.drawable.cell_2);
56      }
57
58      //metode per anar visualitzant les imatges que pertoca per pantalla
59      protected void pinta(Canvas canvas, Coordinated coordinated) {
60          graella.pinta(canvas, coordinated);
61      }
62
63      //control del loop
64      public void aturar() { MapaLoopThread.setRunning(false); }
67
68      public void partir() { MapaLoopThread.setRunning(true); }
71
72  }
```

Figura 5.7: Classe ScreenMap

```
1   package tfg.tfgv2;
2
3   import android.app.Activity;
4   import android.graphics.Canvas;
5   import android.net.wifi.ScanResult;
6   import android.os.Handler;
7   import android.os.Looper;
8   import android.util.Log;
9
10  import java.util.List;
11
12  /**
13   * Classe per recarregar la pantalla
14   */
15  public class LoopMapa extends Thread {
16      static final long FPS = 30;
17      private ListViewWithCheckboxActivity listViewCheckBox = new ListViewWithCheckboxActivity();
18      private int[] actualSignal = new int[3];
19      private Handler timerHandler;
20      private ScreenMap view;
21      private boolean running = false;
22      private ScanListObject slo;
23      private Trilateration position = new Trilateration();
24      private Coordinated coordinadas = new Coordinated();
25      private double [] distancias = {0,0,0};
26      private Coordinated coordinated1 = new Coordinated( x: 2, y: 2);
27      private Coordinated coordinated2 = new Coordinated( x: 5, y: 7);
28      private Coordinated coordinated3 = new Coordinated( x: 9, y: 8);
29
30
31      LoopMapa(Activity activity, ScreenMap view) {
32          this.view = view;
33          slo = new ScanListObject(activity);
34
35      }
```

Figura 5.8: Classe LoopMap

```
36
37      void setRunning(boolean run) { running = run; }
40
41      @Override
42      public void run() {
43          slo.scanWifi();
44
45          timerHandler = new Handler(Looper.getMainLooper());
46          timerHandler.postDelayed(() → {
49                  if (slo.getListReady()) {
50                      slo.scanWifi();
51                      getUpdatedSignal(slo.getResults());
52                      Log.d( tag: "señals", msg: ""+actualSignal[0]+"//"+actualSignal[1]+"//"+actualSignal[2]);
53
54                      distancias = distancias();
55                      Log.d( tag: "distancias", msg: ""+distancias[0]+"//"+distancias[1]+"//"+distancias[2]);
56                      coordinadas = position.trilateration(coordinated1,distancias[0],coordinated2, distancias[1]
57                      Log.d( tag: ""+coordinadas.getX(), msg: ""+coordinadas.getY());
58                      drawBg();
59                  }
60
61                  timerHandler.postDelayed( r: this,  delayMillis: 100);
62          },  delayMillis: 100);
64
65      }
66
67      private void drawBg(){
68          long ticksPS = 1000 / FPS;
69          long startTime;
70          long sleepTime;
71          Canvas c = null;
72          if (running) {
73              startTime = System.currentTimeMillis();
74              try {
75                  c = view.getHolder().lockCanvas();
76                  synchronized (view.getHolder()) {
77                      view.pinta(c, coordinadas);
78                  }
79              } finally {
80                  if (c != null) view.getHolder().unlockCanvasAndPost(c);
81              }
82              sleepTime = ticksPS - (System.currentTimeMillis() - startTime);
83              try {
84                  if (sleepTime > 0) sleep(sleepTime);
85                  else sleep( millis: 10);
86
```

Figura 5.9: Classe LoopMap

```
79              } finally {
80                  if (c != null) view.getHolder().unlockCanvasAndPost(c);
81              }
82              sleepTime = ticksPS - (System.currentTimeMillis() - startTime);
83              try {
84                  if (sleepTime > 0) sleep(sleepTime);
85                  else sleep( millis: 10);
86
87              } catch (Exception ignored) {
88              }
89          }
90      }
91
92      private void getUpdatedSignal(List<ScanResult> resultsUpdated) {
93
94          for (int i = 0; i < listViewCheckBox.networks.length; i++) {
95
96              for (ScanResult scanResult : resultsUpdated) {
97
98                  if (scanResult.SSID.equals(listViewCheckBox.networks[i])){
99
100                     actualSignal[i] = scanResult.level;
101                 }
102             }
103         }
104     }
105
106     public double [] distancias(){
107         double k = 92.45;
108         double f = 2.4;
109         double constant = 20*Math.log10(f)+k;
110         distancias[0] = Math.pow(10,(((-actualSignal[0]-30)-constant)/20))*1000;
111         distancias[1] = Math.pow(10,(((-actualSignal[1]-30)-constant)/20))*1000;
112         distancias[2] = Math.pow(10,(((-actualSignal[2]-30)-constant)/20))*1000;
113
114         return distancias;
115     }
116
117
118 }
```

Figura 5.10: Classe LoopMap

```java
package tfg.tfgv2;

import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Paint;
import android.graphics.Rect;

/**
 * Classe per gestionar les imatges
 */

public class Image {


    private Bitmap bmp;
    private Paint paint;
    private int width;  //longitud de la imatge
    private int height;  // altura de la imatge

    public Image(ScreenMap mapa, int dib) {
        bmp = BitmapFactory.decodeResource(mapa.getResources(), dib);
        bmp = BitmapFactory.decodeResource(mapa.getResources(), dib);
        this.width = bmp.getWidth();
        this.height = bmp.getHeight();
    }

    public void pinta(Canvas canvas, int x, int y, int w, int h) {


            Rect src = new Rect( left: 0,  top: 0, width, height);
            Rect dst = new Rect(x, y,  right: x + w,  bottom: y + h);
            paint = new Paint();
            canvas.drawBitmap(bmp, src, dst, paint);



    }

    //nivell de transparència
    public void setAlpha(int alpha) {
        paint.setAlpha(alpha);
    }

}
```

Figura 5.11: Classe Image

```java
package tfg.tfgv2;

/**
 * Classe pensada en futurs estats de les caselles, quan es pugui insertar mapa.
 */

public enum State {BUIT}

```

Figura 5.12: Classe State

```java
package tfg.tfgv2;

/**
 * Created by mac on 08/06/2019.
 */


public class LviCheckbox {

    private boolean checked = false;

    private String itemText = "";

    public boolean isChecked() { return checked; }

    public void setChecked(boolean checked) { this.checked = checked; }

    public String getItemText() { return itemText; }

    public void setItemText(String itemText) { this.itemText = itemText; }
}
```

Figura 5.13: Classe LviCheckbox

```java
package tfg.tfgv2;

/**
 * Classe pensada en futurs estats de les caselles, quan es pugui insertar mapa.
 */

public enum State {BUIT}

```

Figura 5.14: Classe ListViewItemViewHolder

```java
package tfg.tfgv2;

import android.content.Context;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.CheckBox;
import android.widget.TextView;

import java.util.List;

/**
 * Created by mac on 08/06/2019.
 */

public class ListViewItemCheckboxBaseAdapter extends BaseAdapter {

    private List<LviCheckbox> LviCheckboxList = null;

    private Context ctx = null;

    public ListViewItemCheckboxBaseAdapter(Context ctx, List<LviCheckbox> LviCheckboxList) {
        this.ctx = ctx;
        this.LviCheckboxList = LviCheckboxList;
    }

    @Override
    public int getCount() {
        int ret = 0;
        if (LviCheckboxList != null) {
            ret = LviCheckboxList.size();
        }
        return ret;
    }

    @Override
    public Object getItem(int itemIndex) {
        Object ret = null;
        if (LviCheckboxList != null) {
            ret = LviCheckboxList.get(itemIndex);
        }
        return ret;
    }

    @Override
    public long getItemId(int itemIndex) { return itemIndex; }
```

Figura 5.15: Classe ListViewItemBaseAdapter

```
50      @Override
51      public View getView(int itemIndex, View convertView, ViewGroup viewGroup) {
52
53          ListViewItemViewHolder viewHolder = null;
54
55          if (convertView != null) {
56              viewHolder = (ListViewItemViewHolder) convertView.getTag();
57          } else {
58              convertView = View.inflate(ctx, R.layout.activity_list_view_with_checkbox_item,
59                      root: null);
60
61              CheckBox listItemCheckbox = (CheckBox) convertView.findViewById
62                      (R.id.list_view_item_checkbox);
63
64              TextView listItemText = (TextView) convertView.findViewById(R.id.list_view_item_text);
65
66              viewHolder = new ListViewItemViewHolder(convertView);
67
68              viewHolder.setItemCheckbox(listItemCheckbox);
69
70              viewHolder.setItemTextView(listItemText);
71
72              convertView.setTag(viewHolder);
73          }
74
75          LviCheckbox lviCheckbox = LviCheckboxList.get(itemIndex);
76          viewHolder.getItemCheckbox().setChecked(lviCheckbox.isChecked());
77          viewHolder.getItemTextView().setText(lviCheckbox.getItemText());
78
79          return convertView;
80      }
81
82  }
```

Figura 5.16: Classe ListViewItemBaseAdapter

```java
package tfg.tfgv2;

import android.app.Activity;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiManager;
import android.support.v7.app.AppCompatActivity;
import java.util.ArrayList;
import java.util.List;

/**
 * Created by mac on 15/06/2019.
 */

public class ScanList extends AppCompatActivity {

    private List<ScanResult> results;
    private ArrayList<LviCheckbox> arrayList = new ArrayList<>();
    private Boolean listReady = false;
    private Activity activity;
    private WifiManager wifiManager;

    private BroadcastReceiver wifiReceiver = new BroadcastReceiver() {

        @Override
        public void onReceive(Context context, Intent intent) {
            results = wifiManager.getScanResults();
            activity.unregisterReceiver(this);
            listReady = true;
        }

    };

    public ScanList(Activity activity) { this.activity = activity; }

    public List<ScanResult> getResults() {
        if (!results.isEmpty()) {
            return results;
        } else {
            return null;
        }
    }
```

Figura 5.17: Classe MainActivity

```java
    public Boolean getListReady() { return listReady; }

    public void scanWifi() {
        arrayList.clear();
        wifiManager = (WifiManager)
                activity.getApplicationContext().getSystemService(Context.WIFI_SERVICE);
        activity.registerReceiver
                (wifiReceiver, new IntentFilter(WifiManager.SCAN_RESULTS_AVAILABLE_ACTION));
        wifiManager.startScan();
    }

}
```

Figura 5.18: Classe MainActivity

```java
package tfg.tfgv2;


public class Trilateration {

    //Constructor


    public Trilateration() {
    }

    public Coordinated trilateration(
            Coordinated location1, double distance1,
            Coordinated location2, double distance2,
            Coordinated location3, double distance3){



        double[] P1   = new double[2];
        double[] P2   = new double[2];
        double[] P3   = new double[2];
        double[] ex   = new double[2];
        double[] ey   = new double[2];
        double[] p3p1 = new double[2];
        double jval  = 0;
        double temp  = 0;
        double ival  = 0;
        double p3p1i = 0;
        double triptx;
        double tripty;
        double xval;
        double yval;
        double t1;
        double t2;
        double t3;
        double t;
        double exx;
        double d;
        double eyy;

```

Figura 5.19: Classe Trilateració

```
41          //punts a vectors
42          //Punt 1
43          P1[0] = location1.getY();
44          P1[1] = location1.getX();
45          //Punt 2
46          P2[0] = location2.getY();
47          P2[1] = location2.getX();
48          //Punt 3
49          P3[0] = location3.getY();
50          P3[1] = location3.getX();
51
52
53          for (int i = 0; i < P1.length; i++) {
54              t1   = P2[i];
55              t2   = P1[i];
56              t    = t1 - t2;
57              temp += (t*t);
58          }
59          d = Math.sqrt(temp);
60          for (int i = 0; i < P1.length; i++) {
61              t1    = P2[i];
62              t2    = P1[i];
63              exx   = (t1 - t2)/(Math.sqrt(temp));
64              ex[i] = exx;
65          }
66          for (int i = 0; i < P3.length; i++) {
67              t1      = P3[i];
68              t2      = P1[i];
69              t3      = t1 - t2;
70              p3p1[i] = t3;
71          }
72          for (int i = 0; i < ex.length; i++) {
73              t1 = ex[i];
74              t2 = p3p1[i];
75              ival += (t1*t2);
76          }
77          for (int  i = 0; i < P3.length; i++) {
78              t1 = P3[i];
79              t2 = P1[i];
80              t3 = ex[i] * ival;
81              t  = t1 - t2 -t3;
82              p3p1i += (t*t);
83          }
```

Figura 5.20: Classe Trilateració

```
83          }
84          for (int i = 0; i < P3.length; i++) {
85              t1 = P3[i];
86              t2 = P1[i];
87              t3 = ex[i] * ival;
88              eyy = (t1 - t2 - t3)/Math.sqrt(p3p1i);
89              ey[i] = eyy;
90          }
91          for (int i = 0; i < ey.length; i++) {
92              t1 = ey[i];
93              t2 = p3p1[i];
94              jval += (t1*t2);
95          }
96          xval = (Math.pow(distance1, 2) - Math.pow(distance2, 2) + Math.pow(d, 2))/(2*d);
97          yval = ((Math.pow(distance1, 2) - Math.pow(distance3, 2) + Math.pow(ival, 2) +
98                  Math.pow(jval, 2))/(2*jval)) - ((ival/jval)*xval);
99
100         t1 = location1.getY();
101         t2 = ex[0] * xval;
102         t3 = ey[0] * yval;
103         triptx = t1 + t2 + t3;
104
105         t1 = location1.getX();
106         t2 = ex[1] * xval;
107         t3 = ey[1] * yval;
108         tripty = t1 + t2 + t3;
109
110
111         return new Coordinated(Math.abs((int)triptx),Math.abs((int)tripty));
112
113     }
114
115
116 }
117
```

Figura 5.21: Classe Trilateració

```java
package tfg.tfgv2;

import android.app.Activity;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiManager;
import android.support.v7.app.AppCompatActivity;
import java.util.ArrayList;
import java.util.List;

/**
 * Created by mac on 15/06/2019.
 */

public class ScanList extends AppCompatActivity {

    private List<ScanResult> results;
    private ArrayList<LviCheckbox> arrayList = new ArrayList<>();
    private Boolean listReady = false;
    private Activity activity;
    private WifiManager wifiManager;

    private BroadcastReceiver wifiReceiver = new BroadcastReceiver() {

        @Override
        public void onReceive(Context context, Intent intent) {
            results = wifiManager.getScanResults();
            activity.unregisterReceiver(this);
            listReady = true;
        }

    };

    public ScanList(Activity activity) { this.activity = activity; }

    public List<ScanResult> getResults() {
        if (!results.isEmpty()) {
            return results;
        } else {
            return null;
        }
    }
```

Figura 5.22: Classe ScanList

```java

    public Boolean getListReady() { return listReady; }

    public void scanWifi() {
        arrayList.clear();
        wifiManager = (WifiManager)
                activity.getApplicationContext().getSystemService(Context.WIFI_SERVICE);
        activity.registerReceiver
                (wifiReceiver, new IntentFilter(WifiManager.SCAN_RESULTS_AVAILABLE_ACTION));
        wifiManager.startScan();
    }

}
```

Figura 5.23: Classe ScanList

```java
package tfg.tfgv2;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;

/**
 * Created by mac on 13/06/2019.
 */

public class PresentationActivity extends Activity {

    // Duración en milisegundos que se mostrará el splash
    private final int DURACION_SPLASH = 3000; // 3 segundos

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.presentation);

        new Handler().postDelayed(new Runnable() {
            public void run() {
                // Cuando pasen los 3 segundos, pasamos a la actividad ListViewCheckBoxActivity
                Intent intent = new Intent( packageContext: PresentationActivity.this,
                        ListViewWithCheckboxActivity.class);
                startActivity(intent);
                finish();
            }

            ;
        }, DURACION_SPLASH);
    }
}
```

Figura 5.24: Classe PresentationActivity

```java
package tfg.tfgv2;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.wifi.ScanResult;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.Looper;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.AdapterView;
import android.widget.CheckBox;
import android.widget.ListView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by mac on 08/06/2019.
 */

public class ListViewWithCheckboxActivity extends AppCompatActivity {

    private static final int PERMISSIONS_REQUEST_CODE_ACCESS_COARSE_LOCATION = 1001;
    public static String[] networks = new String[3];
    public List<ScanResult> results;
    public ArrayList<LviCheckbox> arrayList = new ArrayList<>();
    public ListViewItemCheckboxBaseAdapter listViewDataAdapter;
    public int i = 0;
    ListView listViewWithCheckbox;
    Handler timerHandler;
    private ScanList slo;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_list_view_with_checkbox);

        setTitle("Seleccione 3 puntos de acceso");
        listViewWithCheckbox = findViewById(R.id.list_view_with_checkbox);


        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M && checkSelfPermission
                (Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
```

Figura 5.25: Classe ListViewWithCheckboxActivity

```java
package tfg.tfgv2;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.wifi.ScanResult;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.Looper;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.AdapterView;
import android.widget.CheckBox;
import android.widget.ListView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by mac on 08/06/2019.
 */

public class ListViewWithCheckboxActivity extends AppCompatActivity {

    private static final int PERMISSIONS_REQUEST_CODE_ACCESS_COARSE_LOCATION = 1001;
    public static String[] networks = new String[3];
    public List<ScanResult> results;
    public ArrayList<LviCheckbox> arrayList = new ArrayList<>();
    public ListViewItemCheckboxBaseAdapter listViewDataAdapter;
    public int i = 0;
    ListView listViewWithCheckbox;
    Handler timerHandler;
    private ScanList slo;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_list_view_with_checkbox);

        setTitle("Seleccione 3 puntos de acceso");
        listViewWithCheckbox = findViewById(R.id.list_view_with_checkbox);


        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M && checkSelfPermission
                (Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
```

Figura 5.26: Classe ListViewWithCheckboxActivity

```java
package tfg.tfgv2;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.wifi.ScanResult;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.Looper;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.AdapterView;
import android.widget.CheckBox;
import android.widget.ListView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by mac on 08/06/2019.
 */

public class ListViewWithCheckboxActivity extends AppCompatActivity {

    private static final int PERMISSIONS_REQUEST_CODE_ACCESS_COARSE_LOCATION = 1001;
    public static String[] networks = new String[3];
    public List<ScanResult> results;
    public ArrayList<LviCheckbox> arrayList = new ArrayList<>();
    public ListViewItemCheckboxBaseAdapter listViewDataAdapter;
    public int i = 0;
    ListView listViewWithCheckbox;
    Handler timerHandler;
    private ScanList slo;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_list_view_with_checkbox);

        setTitle("Seleccione 3 puntos de acceso");
        listViewWithCheckbox = findViewById(R.id.list_view_with_checkbox);


        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M && checkSelfPermission
                (Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
```

Figura 5.27: Classe ListViewWithCheckboxActivity

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

        <ListView
            android:id="@+id/list_view_with_checkbox"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

</LinearLayout>
```

Figura 5.28: Layout activity$_{l}ist_{w}ith_{c}heckbox$

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

        <ListView
            android:id="@+id/list_view_with_checkbox"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

</LinearLayout>
```

Figura 5.29: Layout activity$_{l}ist_{w}ith_{c}heckbox_{i}tem$

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

        <ListView
            android:id="@+id/list_view_with_checkbox"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

</LinearLayout>
```

Figura 5.30: Layout presentation

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout

    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/layoutMapa"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@android:color/black">

    <TableLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:stretchColumns="*">

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="wrap_content">


        </TableRow>
        <TableRow>
            <Button
                android:id="@+id/btnNova"

                android:layout_height="match_parent"
                android:layout_width="wrap_content"
                android:text="VUELVA A SELECCIONAR SUS AP"
                android:layout_column="0"/>
        </TableRow>


    </TableLayout>

</LinearLayout>
```

Figura 5.31: Layout mapa

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="tfg.tfgv2">

    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/zlogo"
        android:label="TfgV2"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".PresentationActivity"><intent-filter>
            <action
                android:name="android.intent.action.MAIN" />
            <category
                android:name="android.intent.category.LAUNCHER" />
        </intent-filter></activity>

        <activity
            android:name=".ListViewWithCheckboxActivity">

        </activity>

        <activity
            android:name=".ScanList">

        </activity>

        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

            </intent-filter>
        </activity>

    </application>

</manifest>
```

Figura 5.32: Manifest

CAPÍTOL 6

PROVES

Per a què el projecte es realitzi amb èxit s'han d'anar realitzant proves a mesura que es va desenvolupant l'aplicació. Primer de tot, es van fent proves als petits elements que formen l'aplicació i, poc a poc, es van realitzant les proves dels elements més complexes, fins obtenir com a resultat una aplicació que compleix tots els requisits.

6.1 Proves funcionals

A mesura que es van realitzant els petits programes, hem anat comprovant en diferents situacions com es comporten per observar si compleixen la seva funció. A continuació veurem el llistat de les proves realitzades durant el periode de programació:

- **Coordinades**: Es va crear l'objecte Coordinades i es varen realitzar les proves dels getters i setters corresponents, a part dels constructors de Coordinades.Una vegada comprovat que funciona es va procedir al següent pas.
- **Graella**: Per crear la graella i posicionar-la al centre es varen realitzar moltes proves abans d'aconseguir el que es volia. El crear la graella va ser el més fàcil, però després, situar-la, centrar-la i aconseguir que tengui forma quadrada va ser molt més difícil.
- **Pinta**: El testeig del mètode pinta només va ser necessari en el moment de pintar la graella. D'aquesta forma, al comprovar que funcionava, aquest mètode s'ha pogut utilitzar i es podrà utilitzar per modificar les imatges i el que es pinti en un futur.
- **Loop**: Ha estat també un dels mètodes que més problemes ens ha donat alhora de la realització de l'aplicació però, poc a poc, els problemes s'han anat solucionant fins a obtenir un fil thread que ens pinti, ens actualitzi les senyals, ens calculi la distància als punts d'accès i ens calculi mitjançant la trilateració la nostra localització.

- **ScanList**: Aquest mètode ha estat un dels que més problemes ha donat. Va ser on més temps es va invertir durant el projecte i probablement el causant de que la primera versió del prototip no acabi de cumplir tots els requisits.

- **Mètode trilateració**: Després de moltíssimes proves es va arribar a una mètode de trilateració que ens oferís els resultats.

- **Run en mode debug**: Sense aquest mode la feina de trobar per què no funcionava el programa o més d'un mètode hauria estat una tasca impossible. Gràcies a això, identificar el problema ha estat tasca fàcil per després poder resoldre'l.

- **Log.d**: Moltes vegades enlloc d'utilitzar el mode debug, era necessari treure per pantalla, per Android Studio, els resultats obtenguts o veure si entrava o no als bucles al realitzar alguna acció.

6.2 Cumpliment de requeriments

En acabar l'aplicació és molt important comprovar que es compleixen tots els requisits i en cas de que no es compleixin tots s'ha de treballar i solucionar els problemes de l'aplicació per a què compleixi tots els requisits. A continuació veurem el llistat de requisits funcionals i no funcionals amb el resultat de les proves.

6.2.1 Requeriments funcionals

- **Requeriment funcional 1**: L'usuari ha de poder identificar la seva posició mitjançant un sistema de navegació intern.

  Requeriment aconseguit, encara que s'ha de millorar l'algoritme o hem de trobar una manera per a què aquesta posició sigui més precisa.

- **Requeriment funcional 2**: L'usuari ha de poder seleccionar sobre quins 3 punts d'accés es realitzarán els càlculs de posicionament.

  Requeriment aconseguit, l'usuari abans de saber la seva localització ha d'introduir els 3 punts d'accés.

- **Requeriment funcional 3**: L'usuari ha de ser notificat en qualsevol moment si es talla la senyal wifi.

  L'usuari es notificat amb un missatge a la pantalla quan la senyal és molt irregular i es probable que la seva posició no sigui precisa.

- **Requeriment funcional 4**: L'usuari ha d'identificar per a què serveix i què es realitza a cada una de les pantalles de l'aplicació

  Requeriment aconseguit de forma parcial, ja que durant les proves, alguns usuaris han entès què s'havia de fer a cada pantalla i d'altres no tenien molt clar com funcionava.

- **Requeriment funcional 4**: L'usuari ha de poder tornar a seleccionar els seus punts d'accés en cas d'error al seleccionar-los.

  En cas d'errada, existeix la possiblitat de tornar a la pantalla on l'ususari pot elegir els seus 3 punts d'accés.

Requeriments no funcionals

- **Requeriment no funcional 1**: No ha de ser necessari estar connectat a cap xarxa Wifi per a que l'aplicació pugui funcionar.

  L'aplicació només necessita escoltar el senyal del punts d'accès seleccionats però en cap moment ha de realitzar una connexió,

- **Requeriment no funcional 2**: L'aplicació ha de estar realitzada amb l'entorn de programació Android Studio.

  L'aplicació està realitzada amb aquest entorn de programació.

- **Requeriment no funcional 3**: L'aplicació ha de ser compatible amb tots els sistemes Android fins a la seva darrera actualització 9.0.

  Existeixen tots els permisos necessaris per arribar a la versió 9.

- **Requeriment no funcional 4**: El programador introduirà el posicionament dels 3 punts d'accés per programa. En cas de canvi de posició d'aquests, es notificarà per entregar la nova versió al client.

  Internament, qui ha de col.locar els tres punts d'accés és el programador. En un futur l'existència de una pantalla per a què l'usuari pugui realitzar auqests canvis seria una gran millora.

6.3 Proves usuari final

Les proves realitzades pels usuaris finals són les més importants, degut a que són ells qui utilitzaran l'aplicació i qui ens donaran el seu feedback. En base al seu feedback es podran plantejar nous enfocaments o canviar molts aspectes de l'aplicació. Per això s'han realitzat una sèrie de proves que hem dividit en tres tipus d'usuaris, menors de 18 anys, adults i persones majors de 65 anys.

6.3.1 Usuaris menors 18 anys

Les proves han resultat exitoses a la majoria dels casos amb petites excepcions. En general, amb l'introducció tecnològica que es dóna desde petits, no ha suposat gaire problema.

6.3.2 Usuaris adults

Els resultats de les proves no han estat els esperats. Millors resultats que les persones majors de 65 anys però pitjors que els de les persones menors de 18 anys.

6.3.3 Usuaris persones majors de 65 anys

Han estat millor dels esperats ja que els resultats s'apropen molt al resultat de les persones adultes.

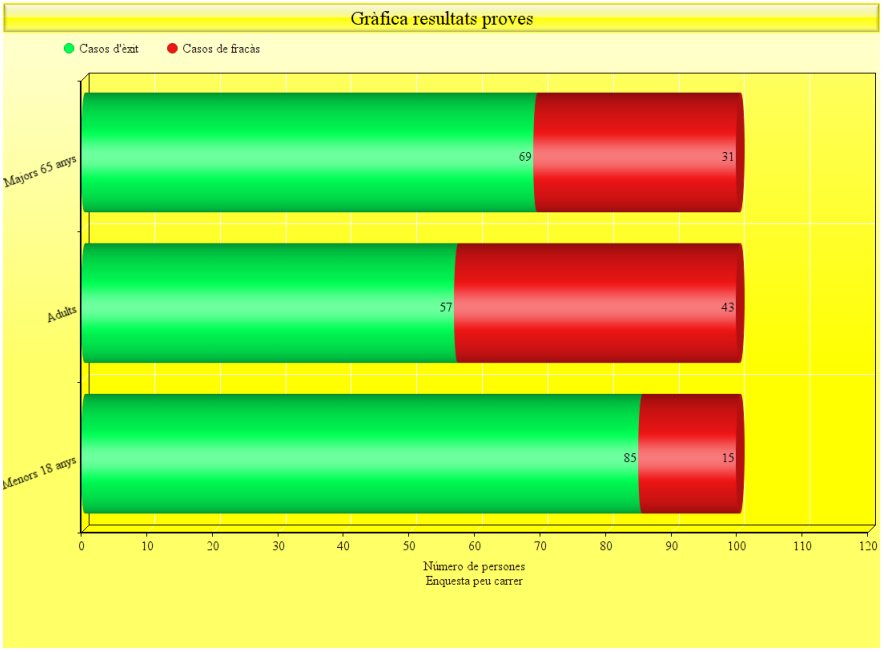

Figura 6.1: Resultats obtinguts

CAPÍTOL 7

CONCLUSIONS

A tot projecte d'aquest tipus s'extreuen moltes conclusions: unes positives i d'altres negatives. Aquí present les conclusions que he pogut extreure durant la realització del projecte.

7.1 Precisió del senyal Wifi

Durant la meva etapa al grau he pogut veure que la senyal que es reb dels diferents punts d'accés wifi no és sempre molt nítida. Al llarg del projecte he pogut comprovar com això ha estat un problema en el moment de poder obtenir la nostra posició.

A partir d'aquí hi ha dos camins a seguir: o millorar l'algoritme i seguir utilitzant la mateixa tecnologia o canviar radicalment de tecnologia i d'idea per intentar trobar-ne una millor i més eficient.

Sempre es pot millorar molt l'algoritme tenint en compte tots els tipus d'esvaïments que pot sofrir la senyal. A la figura 7.1 podem observar els esvaïments que pot sofrir la senyal del Wifi.

Encara així hem estat capaços de trobar una aproximació de la nostra localització.

7.2 Bona gestió del temps

Una bona gestió del temps és molt necessària per aconseguir les dates d'entrega proposades. Durant aquest projecte he pogut comprovar que amb una planificació i gestió del temps no adequades, els projectes es poden veure afectats per canvis aliens que alteren el que s'ha planificat. També he de dir que aconseguir una bona gestió del temps sense cap experiència anterior ha estat una tasca difícil, ja que no vaig calcular amb exactitud

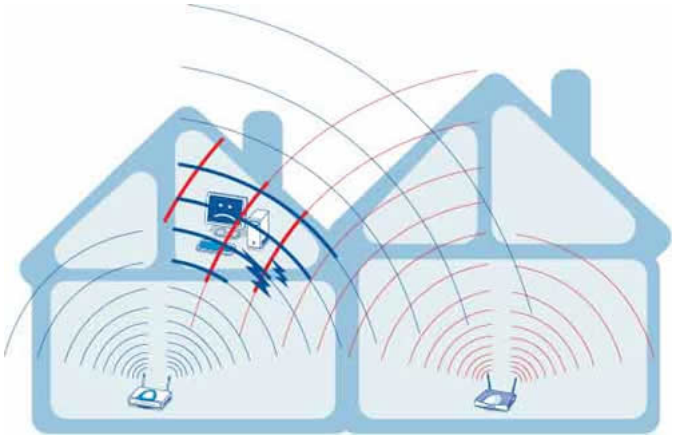

Figura 7.1: Esvaïments senyal Wifi

el temps que vaig haver d'invertir per a la realització de cada tasca. Punt positiu ha estat la capacitat d'adaptació enfront de les adversitats i que el temps no marca mai quina es la data límit d'entrega

7.3 Aplicació

L'aplicació necessita millores però les funcions bàsiques les realitza: es possible la geolocalització al lloc de l'experiment, ja que al ser testat a una vivenda hem pogut observar que els resultats no han estat del tot negatius, encara que millorables.

He pogut observar alguns punts dèbils, com per exemple que al realitzar l'escàner no es trobi la xarxa que es necessita i, per tant, no actualitzi la senyal rebuda o que la senyal que arriba vengui afectada per diversos esvaïments. Aquests efectes negatius fan que l'aplicació no dóni els resultats esperats.

Encara així, sempre es pot trobar un algoritme què ens indiqui la posició amb més precisió.

7.4 Idees

He pogut comprovar que les idees sempre són molt abstractes. Moltes vegades pensam que hem concretat en la realització d'una idea i, després, quan la realitzam veim que era més abstracta del que pareixia. Com més concreta sigui la idea i més clara la forma de realitzar-la, més temps estalviarem i menys canvis sofrirà el projecte. Aquesta conclusió l'he extreta directament del que he sofrit durant el projecte. Encara així acabar la idea principal com a projecte futur es una opció que seria molt interessant.

CAPÍTOL 8

MILLORES APLICABLES

Abans de tot ptresentarem la idea amb la que es va començar el projecte i amb el que ha derivat. Per aquesta raó encara queda molt recorregut per a que aquesta aplicació agafi la forma desitjada desde un principi, però per tema de temps s'ha hagut de presentar aquest prototip amb moltes millores aplicables. Per això aqui es presentarà el treball futur i possibles millores per tenir un producte de qualitat màxima.

8.1 La idea principal

La idea principal del projecte era implementar una aplicació de geolocalització indoor aplicable a la domòtica, en concret en un sistema de so envolvent, que a mesura que et desplaces amb el teu mòbil per las habitacions de la casa, l'aplicació et localitzés dins la vivenda i a través de una pasarela Wifi a Knx reproduïr la música a l'habitació que es troba l'usuari. KNX és un estàndard de protocol de comunicacions de xarxa, basat en OSI, per a edificis intel·ligents. A la figura 8.1 podem veure el seu logo

Figura 8.1: Logo KNX

8.2 Adaptació del projecte

Degut a una mala gestió del temps, es va haver de adaptar el projecte inicial. Es va veure que per a la quantitat de coses planificades, no hi havia temps necessari. Per tant es va decidir només implementar la part de l'aplicació de geolocalització interior. Encara així no ha estat un projecte fàcil de desenvolupar i hem estat més temps de l'esperat.

8.3 Millores

A continuació trobam un llistat amb les possibles millores a aplicar:

- **Estètica i navegació**: Una de les millores aplicables fa referència a l'estètica que es pot oferir a l'usuari. Millorar la presentació de l'aplicació y que la navegació entre pantalles sigui més fluida i intuitiva.
- **Pantalla Configuració**: Implementar una pantalla més on l'usuari pugui resituar els punts d'accés, ja que es probable que es canviïn al llarg del temps, jugar amb la exactitud de la posició en el cas en que no es rebi molt bé el senyal, canviar el mapa on es geolocalitza... moltes configuracions que poc a poc es poden anar afegint.
- **Disseny**: Després de realitzar les fases d'estudi i de anàlisi, amb el llistat de requisits ben definit, procedirem a dissenyar l'arquitectura de la nostra aplicació. En el nostre cas inclourem el disseny de la nostra idea principal i l'actual.
- **Millora algoritme**: Tots els algoritmes de càlcul utilitzats a l'aplicació són millorables. Sempre es poden fer actualitzacions en millorar la precisió d'aquests o a mesura que passi el temps veure com es comporten i fer les correcions necessaries.
- **Aplicable a qualsevol entorn i poder aplicar mapa**: Com ja hem mencinoat a l'apartat de la millora amb una pantalla de configuració es necessari que si l'usuari passa a viure a un altre lloc ell pugui insertar el nou mapa de la localització actual.
- **Tenir en compte obstacles**: En aquest projecte no s'han tengut en compte els posibles objectes que es puguin trobar a l'interior de una vivenda o de qualsevol edificació, per això es important tenir-los en compte si es vol millorar el rendiment de l'aplicació.

8.4 Treball futur

Al descartar la primera idea principal sempre hi haurà treball futur per fer feina en aquesl la idea. A més sempre s'ha de fer feina escoltant el que diu l'usuari.

- **Realitzar aplicació domótica**:Com hem comentat abans, sempre sería possible implementar la primera idea de projecte, una vegada s'haguin realitzat les millores es començará a investigar sobre domòtica i KNX.

- **Recollir sempre feedback dels ususaris**: El feedback dels usuaris es una cosa que s'hauria de fer de manera diària i a final de cada mes analitzar-los i apuntar tots els aspectes que preocupen als usuaris, per fer així una aplicació molt més cómoda per l'usuari.

APÈNDIX A

ANNEXOS

A.1 Manual d'usuari

Per a totes les aplicacions és necessari un manual d'usuari, tant per a una aplicació molt complexa com per a l'aplicació més simple que es pugui desenvolupar. En el meu cas el manual d'usuari està format per l'apartat Instal·lació, on s'explica més endavant com instal·lar l'aplicació a un mòbil que utilitzi un sistema Android i un apartat d'Utilització, on es descriu com utilitzar correctament l'aplicació.

A.1.1 Instal·lació

El primer que s'ha de fer es agafar el nostre telèfon mòbil i assegurar-nos de que utilitza el sistema operatiu Android.

Una vegada comprovat, passarem a activar el mode programador del nostre telèfon mòbil. Per això hem d'obrir la configuració del nostre mòbil i entrar a l'apartat "Acerca del teléfono".Apartat que es pot veure a la figura A.1

A continuació, hem de clicar les vegades que siguin suficients a "modo desarrollador"fins que ens surti per pantalla que el mòbil ja està en mode programador. Una vegada fet això ens sortirà un apartat nou a configuració. Com es pot observar a la figura A.2

Arribats a aquest punt, s'ha d'activar el mode depuació USB. que ens permetrà instal·lar aplicacions pròpies.

En mode programador al connectar el nostre dispositiu en mode programador i el mode depuració activat, com podem observar a la figura A.3, la plataforma Android Studio reconeixerà el nostre dispositiu per instal·lar i executar l'aplicació.

Una vegada realitzats els passos anteriors procedirem a ubicar l'arxiu "tfgV1"a la carpeta de projectes de l'Android Studio. Serà una ubicació diferent per a cada ordinador.

Una vegada situat l'arxiu hem d'obrir-ho a la plataforma Android Studio i executar-ho pitjant el botó de "run"; botó que podem veure als cercles de la figura A.4

Ens apareixerà una finestra on podrem trobar els nostres emuladors i si hem aconseguit posar el nostre mòbil en mode programador amb èxit, també apareixerà el nom del nostre mòbil a la llista de dispositius on executar el programa, com podem observar a la figura A.5

Seleccionam el mòbil i esperam que compili i instal·li l'aplicació al mòbil. Probablement Android Studio hagi de realitzar canvis interns que haurem d'acceptar. Una vegada instal·lada veurem com s'executa l'aplicació al nostre mòbil.

NOTA: Totes les figures es troben al final de l'apartat Annexos

A.1.2 Utilització

Una vegada completada la instal·lació trobarem l'icone de TfgV1 i hem de clicar-hi. Ens apareixerà la pantalla de presentació de l'aplicació com podem veure a la figura A.6

Posteriorment ens apareixerà una pantalla on es farà un primer escàner dels punts d'accès wifi disponibles i seleccionarem per ordre d'introducció de posició dins programa, ordre que el programador us proveïrà abans. Una vegada sellecionats els tres punts d'accés, com podem observar a la figura A.7, es passarà a la pantalla principal.

A la pantalla principal veurem un quadre blanc l'espai on ens pot situar l'aplicació, amb 3 punts d'accés situats al recuadre com podem observar a la figura A.8. La imatge de l'explosió representa la nostra ubicació actual. A mesura que ens desplaçam aquesta imatge es desplaçará.

En cas de mala recepció de la senyal ens apareixerà un missatge per pantalla advertint que no tenim bona senyal i per això no es podrà representar la nostra posició o probablement sigui errònea, com es pot observar a la figura A.8

En cas de haver introduït erròneament els tres punts d'accés anteriors, amb un botó a la pantalla principal podem reelegir els 3 punts d'accés. Botó que podem apreciar a la figura A.8

Una vegada completat el porcés la aplicació farà la seva funció.

A.2 Carta alumne a UIB

Aquests anys com a estudiant de la Universitat de les Illes Balears he après moltes coses, pero del que més te'n adones ara al final, es de que vares entrar a estudiar a la universitat el primer día amb una mentalitat i surts amb una altra totalment diferent. Has vist que pots afrontar els problemes i que no es regala mai res. Per això esper que

aquesta experiència sigui una de les més útils a la meva vida.

Us posaré l'exemple més clar, que es el que he vist els darrers anys. Quan vaig entrar aquí cercava sempre el camí més fàcil per a tot. L'exemple més clar era agafar sempre el company de práctiques més inteligent. Ara m'empenedesc de tot això ja que fins que no he estat tot sol no m'he vist amb dificultats. Gràcies a això se m'han presentat i he estat capaç de superar-les.

Per aquest motiu, demanaria que en la mesura del que es pugui, es pensi quan es bona idea fer treballs en grup i quan no. Moltes gràcies.

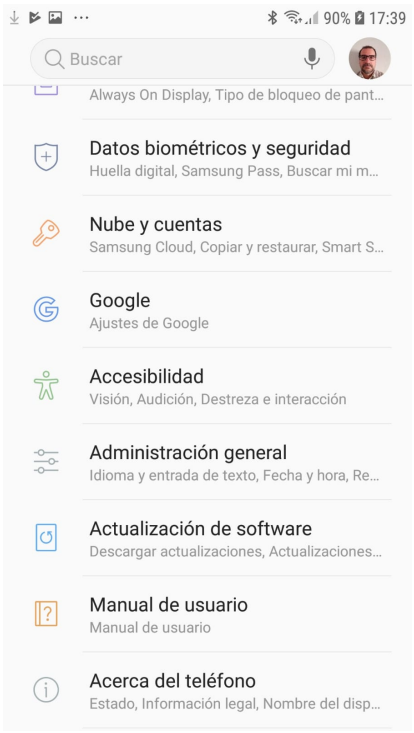

Figura A.1: Captura de pantalla Configuració

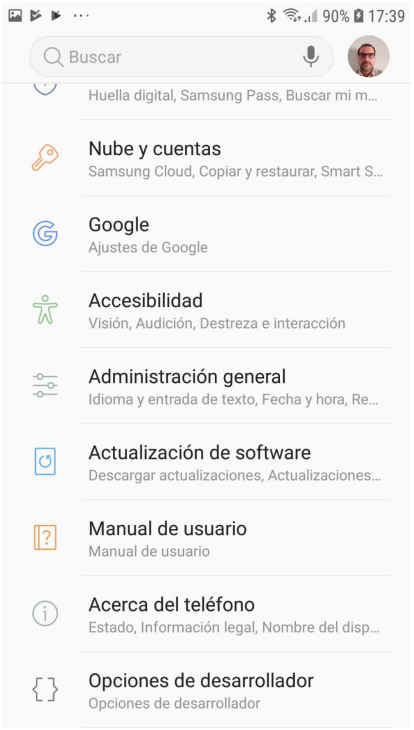

Figura A.2: Captura de pantalla Configuració amb mode desenvolupador activat

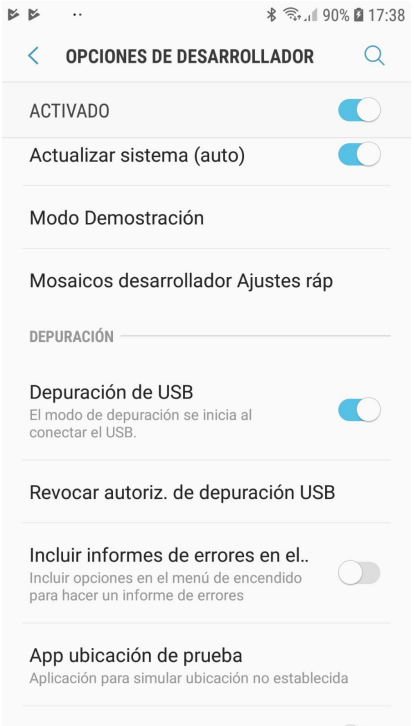

Figura A.3: Captura de pantalla Configuració amb mode debug activat

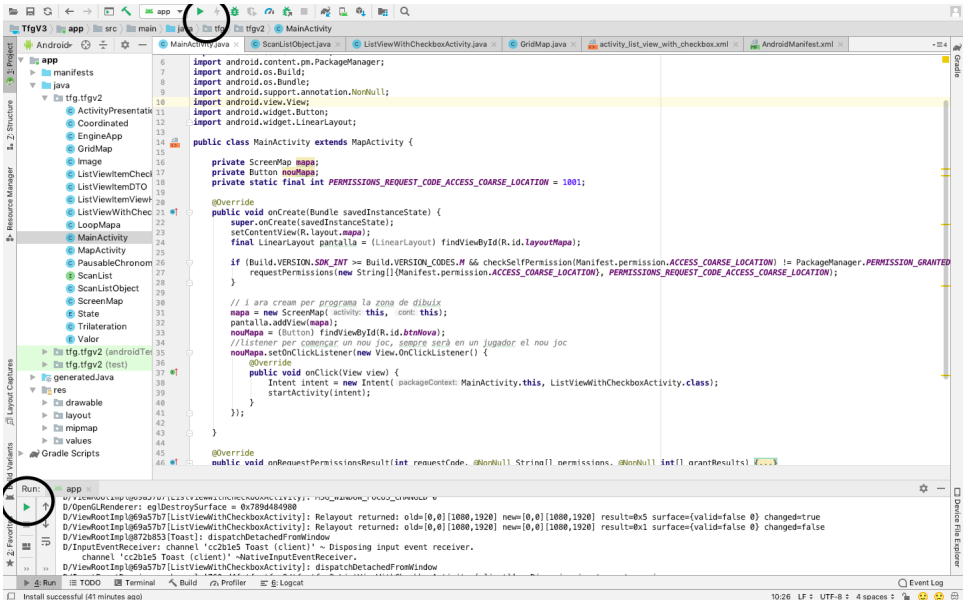

Figura A.4: Captura Android Studio

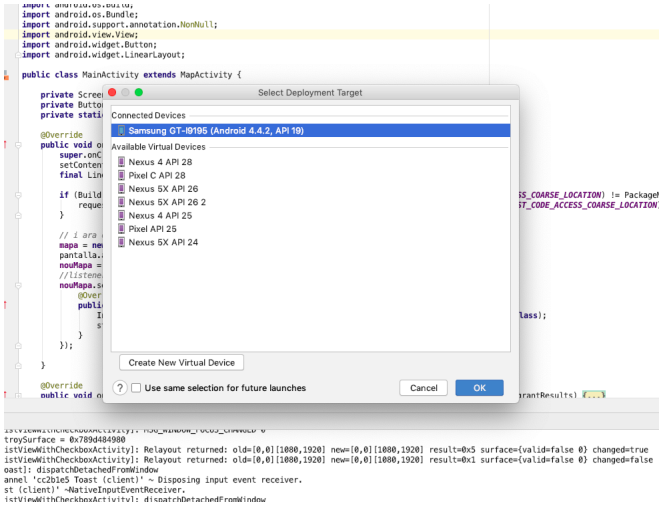

Figura A.5: Captura Android Studio finestra dispositius

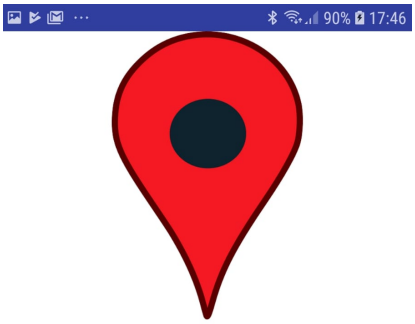

Figura A.6: Captura de pantalla pantalla presentació

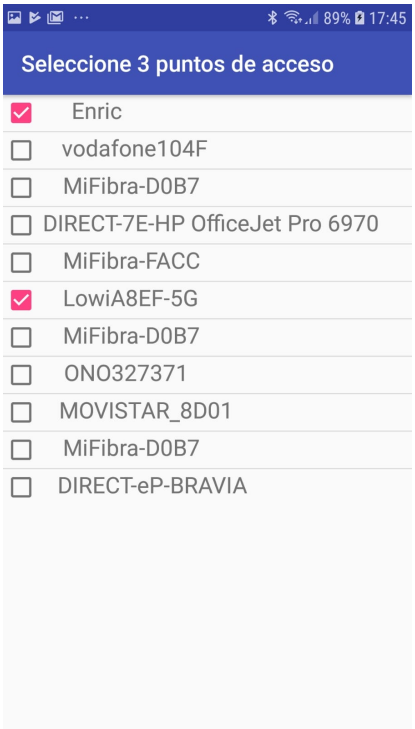

Figura A.7: Captura de pantalla selecció AP

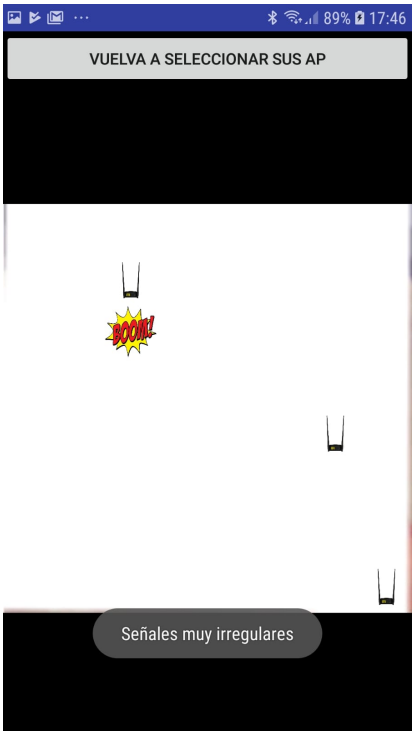

Figura A.8: Captura de pantalla pantalla principal

BIBLIOGRAFIA

[1] “Beacons, "https://es.wikipedia.org/wiki/beacon".” 2.2.1

[2] “Uwb, "https://es.wikipedia.org/wiki/ultrawideband".” 2.2.2

[3] “Rfid, "https://www.by.com.es/blog/que-es-rfid/".” 2.2.4

[4] “Mètodo prototipos, "https://www.ecured.cu/modelo$_d e_p rototipos$".″ 3.2

[5] L. Carrasco, “Apunts xarxes d’operadora.” 3.4.1

[6] “Trilateración, "https://es.wikipedia.org/wiki/trilateraci 3.4.2

[7] “Llibreries android studio, "https://developer.android.com/".” 5.3