# Factura Electrònica

**Màster Interuniversitari en Seguretat de les TIC**

## Treball Final del Màster

*Nom i cognoms:* ***Moisés Fernández Blanco***
*(mfernandezbl@uoc.edu)*

*Directors:* ***Jordi Castella Roca*** *i* ***Alexandre Viejo Galicia***
*Coordinador:* ***Francesc Serratosa Casanelles***

# Índex

# Índex de figures

# Índex de taules

# 1. Introducció: àmbit i problemàtica

En aquests moments, actualment, estem assistint a una verdadera revolució tecnològica en lo que respecta a les tecnologies mòbils. Aquesta revolució tecnològica, podria ésser equiparada amb la Revolució Industrial que va ser viscuda entre finals del segle XVIII i principis del XIX.
L'aparició dels telèfons intel·ligents i l'arribada d'un nou concepte d'us del mòbil, que ja no tan sols es limita a ser una eina per fer trucades, sinó que també per accedir a tota mena de continguts en Internet, així com, el seu ús en tota mena d'aplicacions: Serveis de Localització, Xarxes socials, compartir informació, accedir a serveis de vídeo per *streamming*... així com l'arribada propera de les tecnologies 4G, situa a aquests aparells tecnològics o mini computadors en una eina clau, per proveir de múltiples serveis i utilitats a la nostra societat.

En lo que respecta a aquest projecte, neix precisament pensant en nous serveis que es poden dur o que és duran a terme amb l'ús d'aquestes innovadores tecnologies, més concretament en l'àmbit del comerç electrònic amb dispositius mòbils, on es preveu que sigui un mitjà de pagament d'ús habitual. En aquest procés, de pagament telemàtic, és necessari obtenir un comprovant de pagament, equiparable a l'obtingut per mitjans físics, en resum, cal obtenir un tiquet virtual amb les garanties de seguretat i legals pertinents, aquest projecte neix per donar solució a aquesta necessitat.

Per altra banda el projecte que es presenta a continuació està basat en un anterior Projecte Final de Carrera, de la Universitat Rovira i Virgili dins del Departament d'Enginyeria Informàtica i Matemàtiques, Enginyeria Tècnica en Telecomunicacions, especialitat en Telemàtica, no obstant el projecte anterior presentava algunes limitacions: en quant a la validació de la factura o tiquet electrònic (Factura-e com veurem més endavant), així com, de la validació de la signatura digital doncs no era capaç de realitzar ambdues validacions en el mateix dispositiu mòbil, el tiquet o factura s'havia d'enviar a un servei web per tal de que fos validat, amb els conseqüents riscos de seguretat. Per altra banda, també presentava la limitació de que la informació s'emmagatzemava en el dispositiu en text clar, sense xifrar, cosa no del tot òptima de cara a dissenyar un sistema d'aquestes característiques. Per últim, comentar que s'ha agafat la idea, però el desenvolupament i el codi són totalment nous, partint totalment de cero, sense aprofitar absolutament res del altre projecte.

## 1.1. Motivació

Els mitjans de pagament electrònics s'estan estenent, formant part de la vida quotidiana, i ja es comencen a veure els primers models de pagament electrònic mitjançant la tecnologia mòbil. Vegis per exemple el pilot per realitzar pagaments amb terminals mòbils en el metro de Londres; Figura 1

**Figura 1. Pagament de viatge en metro amb tecnologies mòbils**

Font: http://e-global.es/b2b-blog/2007/11/30/pago-con-movil-en-el-metro-de-londres-piloto-con-tecnologia-nfc/

Tot plegat, això comporta tota una sèrie de necessitats, entre les quals està comptar de mecanismes de pagament prou segurs i que permetin incorporar factures electròniques de forma que aquestes siguin emeses de forma oficial i tinguin la validesa legal pertinent, aquest projecte neix sota aquest context, i aquesta és la principal motivació: crear un sistema de tiqueting que permeti la recepció de factures al mòbil amb validesa jurídica i legal, garantint la seguretat, confidencialitat i integritat de la informació processada. Amb aquest propòsit, com a format de factura o tiquet electrònic s'ha triat el format de Factura-e del "Ministerio de Industria, Turismo y Comercio" conjuntament amb el "Ministerio de Hacienda y Administraciones Públicas", aquest format garanteix la validesa jurídica i legal del tiquet electrònic.

Per altra banda, tal i com s'ha esmentat anteriorment, el projecte neix d'un PFC realitzat a la URV, el qual tenia certes carències en quant a mesures de seguretat, bàsicament.

- La informació s'emmagatzemava en clar en el terminal mòbil.
- La validació de format i de la signatura digital és realitzava de la banda servidor, transmeten totes les dades per la xarxa amb els conseqüents riscos de seguretat.

Per últim, encara que el producte final és una aplicació mòbil, no ens centrarem tant en la part d'aplicació que no fa referència a la seguretat, sinó que ens centrarem més en aquesta darrera. No s'ha d'oblidar que aquest projecte forma part del Màster Interuniversitari de Seguretat de les TIC, per lo que la motivació principal, és la de proveir a l'esmentada aplicació de mitjans telemàtics prou segurs per tal de realitzar el procés de *tiqueting* o facturació.

## 1.2. Objectius

Tal i com ja s'ha introduït aquest projecte persegueix proporcionar una eina de *tiqueting* segura, durant el procés de compra amb terminals mòbils.
Amb aquesta finalitat és crearà una aplicació mòbil per la plataforma ***Android*** la qual podrà rebre mitjançant l'ús de tecnologies sense fils, i més concretament mitjançant ***NFC*** (Near Field Communications) i/o ***Bluetooth***, el tiquet de la compra realitzada amb el terminal mòbil.

El tiquet rebut serà emmagatzemat en el terminal mòbil de forma segura, utilitzant algorismes per al xifrat de les dades, i seguirà el format de Factura Electrònica del *Ministerio de Industria, Energía y Turismo*, que bàsicament és tracta d'un format estàndard basat en el llenguatge de marques *XML* (eXtensible Markup Language), així com, basat a la vegada en altres iniciatives europees (UN/CEFACT - UNECE).
Aquests tiquets prèviament hauran d'haver sigut signats, utilitzant la signatura electrònica avançada, i més concretament utilitzant el format *XAdES*, la qual en termes legals s'equiparà a la signatura manuscrita.
L'aplicació mencionada, rebut i emmagatzemat el tiquet, s'encarregarà de la validació de format, així com, de la verificació de la signatura digital, per tal de validar que la factura rebuda és legalment vàlida. Tot aquest procés serà realitzat sense que les dades surtin del terminal mòbil, per motius de seguretat.

Segons això s'han definit els següents requeriments funcionals:

| Requeriment | Nom abreujat | Descripció |
|---|---|---|
| R01 | Validació de la signatura digital | Validació de la signatura digital i el certificat al telèfon mòbil. Creació d'un prototip encarregat de realitzar la validació d'un joc de factures/tiquets signats electrònicament. |
| R02 | Validació del document XML | Validació del tiquet/ document XML contra les especificacions/esquemes XSD del projecte Facturae del MINETUR. Creació de prototip encarregat de la validació de factures en l'esmentat format. |
| R03 | Definició de l'arquitectura del programari mòbil. | Es crearà un projecte Android integrat amb MAVEN, i es definirà l'arquitectura del programari. |
| R04 | Integració de mòduls de validació. | S'integraran els mòduls de validació de la signatura i de validació del format del XML (requeriments R01 i R02) |
| R05 | Mòdul d'autenticació. | S'implementarà un mòdul per l'autenticació de l'usuari a l'aplicació. |
| R06 | Mòdul d'emmagatzematge. | S'implementarà un mòdul per l'emmagatzematge segur de la informació. |
| R07 | Mòdul de descobriment de factures. | S'implementarà un mòdul per enviar les factures al dispositiu mòbil utilitzant Bluetooth i/o NFC. |
| R08 | Mòdul de consulta o visualització d'una factura. | Permetrà consultar o visualitzar una factura determinada. |
| R09 | Mòdul de cerca i llistat de factures | Permetrà cercar les factures emmagatzemades al dispositiu i mostrar un llistat. |
| R10 | Mòdul de definició de categories | Permetrà crear/ editar i esborrar les categories que s'utilitzen per la posterior classificació de factures. |
| R11 | Mòdul per la consulta de categories. | Permetrà consultar el llistat de categories definides a l'aplicació. |

**Taula 1. Requeriments**

Cadascun d'aquests requeriments han estat abordats en les diferents fases del projecte.

Per últim en lo que respecta a aquest apartat, comentar que aquest projecte ha estat batejat amb el nom '*mInvoice*' (Mobility Invoice), és a dir, factura electrònica mòbil.

## 1.3. Organització de la memòria

Aquesta memòria s'estructura/ organitza en deu seccions, tal i com veurem a continuació:

1. A la secció segona es cobreixen els diferents aspectes tecnològics que intervenen en el projecte, és a dir, les tecnologies emprades.
2. La secció tercera conté una descripció de l'arquitectura de la proposta.
3. La quarta secció cobreix lo relatiu al procés de validació de factures, des de la recepció, la validació del format de Factura-e, així com, la validació de la signatura digital en el terminal mòbil.
4. A la cinquena secció es descriu com s'emmagatzema la informació de forma segura mitjançant la capa de persistència.
5. La sisena secció de la memòria, pretén descriure el subsistema de descobriment de factures.
6. A la setena secció, es presenta l'aplicació on es mostren les diverses pantalles o funcionalitats de l'aplicació.
7. La vuitena secció descriu les proves que s'han dut a terme per validar el correcte funcionament de l'aplicació.
8. La novena secció presenta les conclusions del projecte.
9. I la darrera, tota una sèrie d'annexos, amb el glossari i altres consideracions oportunes relatives al projecte.

# 2. Tecnologies emprades

En aquest apartat es pretén presentar les diferents tecnologies que s'han utilitzat per dur a terme el desenvolupament del projecte, a continuació es presenta un catàleg d'aquestes:

| Tecnologia | Descripció |
|---|---|
| Factura-e | Conforma l'estàndard de document XML a seguir per la implementació del tiquet. |
| Android | Sistema operatiu per a dispositius mòbils. |
| JIBX | Tecnologia utilitzada per la validació i interpretació del tiquet (document XML). |
| MITyCLibs | Components de signatura digital del "Ministerio d'Indústria, Turismo y Comercio". |
| Spongy Castle | APIs criptogràfiques per a plataformes mòvils derivades de les conegudes APIs de Bouncy Castle. |
| SQLite | Emmagatzematge/ persistència de la informació. |
| ORMLite | Llibreries utilitzades per al mapeig del model relacional implementat mitjançant SQLite a un model d'objectes. |
| SQLCipher | Llibreries utilitzades per al xifrat de la informació emmagatzemada en la BBDD. |

| | |
|---|---|
| Maven | Eina utilitzada per a la construcció del projecte, així com, per la gestió de dependències. |
| BlueCove | Llibreria Java per la implementació de sistemes de comunicacions bluetooth. |
| Androlog | Llibreria utilitzada per la depuració de l'aplicació mitjançant registres de traces. |

**Taula 2. Catàleg de tecnologies emprades**

En els apartats següents s'entrarà en detall en cadascuna d'aquestes tecnologies i el paper que juguen en el projecte.

## 2.1. Factura-e

El projecte de Factura-e, és un projecte creat conjuntament per el "Ministerio de Industria, Energia y Turismo" i el "Ministerio de Hacienda y Administraciones públicas", que persegueix la finalitat de poder emetre i rebre factures en format electrònic amb les característiques de que aquestes siguin legalment reconegudes. Funcionalment seria l'equivalent de la factura emesa en paper, però amb suport electrònic o telemàtic. Al respecte l'ante-projecte de llei de mesures per l'impuls de la Societat d'Informació defineix la factura electrònica com "un document electrònic que compleix els requisits legals i reglamentaris exigibles per a les factures y que a més a més, garanteix l'autenticitat del seu origen i l'integritat del seu contingut, permeten atribuir la factura al seu obligat tributari".

A més a més, determina tres condicionants per l'implementació de l'e-Factura:

- Es requereix d'un format electrònic de factura.
- Es fa necessària la transmissió telemàtica.
- L'esmentat format i la transmissió telemàtica, deuen garantir l'integritat i autenticitat d'aquesta a traves de la signatura digital reconeguda.

Que la signatura digital sigui reconeguda, implica a la vegada tres condicionants:

- Que sigui una signatura electrònica avançada, entenent-se com a tal, aquella que permet identificar al signant i qualsevol canvi posterior de les dades signades, i que és vinculada al signant de manera única i a les dades a les que fa referència i que ha estat creada per mitjans sota el seu exclusiu control.
- Que estigui bassada en un certificat digital reconegut.
- Que sigui generada mitjançant un dispositiu segur de signatura.

Aquests requeriments són a la vegada de compliment per a aquest projecte, ja que tot sistema de pagament telemàtic i de "tiqueting" deuria permetre el fet de presentar i tractar les conseqüents factures de forma legal, dit d'una altra forma, qualsevol persona que realitzes un pagament telemàtic via un sistema mòbil, deuria poder rebre un tiquet/ factura que pogués presentar de forma oficial als organismes legals, mitjançant la que pugues fer reclamacions... en definitiva que tingués validesa legal i

jurídica, complint-se les garanties de que aquesta ha estat emesa per una entitat física o jurídica garantint la integritat, autenticitat i no repudio.

Per altra banda, el projecte de Factura-e ofereix un estàndard de factura electrònica, l'especificació del qual es bassada en un llenguatge de marques basat en el format XML (eXtensible Markup Language). Actualment l'esmentat estàndard està en versió 3.2, i amb la versió 4.0 en l'estat d'esborrany.

Els fets anteriorment explicats són els motius per els que en aquest projecte s'ha triat el projecte de Factura-e per al tractament digital de tiquets electrònics, de forma que quan el dispositiu electrònic emissor de factures, emet una factura, lo que farà serà generar un fitxer en format Factura-e (XML), enviant-lo al receptor, en aquest cas el dispositiu mòbil, el qual processarà les dades d'acord a aquest format.

La versió suportada de Factura-e per aquest projecte és la versió 3.2, ja que la 4.0 està en fase d'esborrany, tal i com, s'ha comentat.

L'especificació de factura electrònica menciona l'obligació de signar la factura emesa, en aquest projecte el tipus de format que utilitzarem per la signatura digital és el format XADES-EPES.

Per finalitzar aquest apartat, cal dir que malgrat utilitzar el format/estàndard de Factura-e, de cara a processar el XML no s'han utilitzat les API que proporciona aquest projecte, ja que, les tecnologies emprades per l'anterior no permetien la incorporació en dispositius mòbils. L'alternativa ha estat utilitzar les API de JIBX conjuntament amb els XSD proporcionats per Factura-e.

## 2.2. Android

Android és avui en dia el sistema operatiu més utilitzat per dispositius mòbils i tabletes, és un sistema operatiu el cor del qual està construït sobre la versió 2.6 del sistema operatiu Linux. La gran avantatge que aporta és que un sistema operatiu multiplataforma, de caràcter lliure i totalment gratuït. Aquestes tres característiques han fet que fos adoptat per un ampli número de fabricants.

La màquina virtual que porta Android, es l'anomenada màquina virtual, Dalvik, la qual està optimitzada per consumir poca memòria, i dissenyada per executar varies instàncies de màquina virtual a la vegada. Habitualment s'acostuma a programar amb Java, i es generen compilats (arxius dex; bytecode) executables per l'esmentada màquina virtual.
Per totes aquestes característiques s'ha triat aquest Sistema Operatiu.

## 2.3. JIBX

Per al processat de documents XML amb format Factura-e, s'han utilitzat les llibreries de JIBX, aquestes llibreries permeten la serialització o des-serialització, d'objectes a XML i de XML a objectes, respectivament. En aquest projecte, aquestes llibreries s'han utilitzat per a deserialitzar els documents XML, factures rebudes per al sistema de descobriment de factures, i convertir les dades al model d'objectes Java.
Les esmentades llibreries, han demostrat durant les proves un molt bon rendiment, així com, una bona integració sobre Android, fet per el qual ens ha decantat per la utilització d'aquestes.
Aquestes llibreries són programari lliure i estan llicenciades sota llicències BSD,

Apache BCEL i XPP3.

## 2.4. MITyCLibs

Per abreujar, les llibreries del MINETUR, s'han anomenat MITyCLibs, aquestes llibreries permeten la signatura de documents XML (XADES), així com, la seva validació, utilitzant certificats.

Suporten varis tipus de signatura:

- Detached: on el document que es signa i la signatura són documents totalment independents.
- Enveloped: on la signatura s'utilitza per signar part del document que la conté.
- Enveloping: on la signatura forma un envolcall sobre les pròpies dades signades.

I diversos formats:

- XAdES-BES: signatura bàsica de signatura que compleix els requisits legals per la signatura avançada.
- XAdES-EPES: signatura bàsica a la que se li afegeix informació sobre la política de signatura. Una política de signatura és un criteri definit en la política que la signatura ha de complir.
- XAdES-T: afegeix un segell de temps que garanteix el moment en que es va fer la signatura.
- XAdES-C: tipus de signatura que inclou una referència a l'informació relativa a la cadena de certificats, així com, de l'estat de validesa del certificat emprat per la signatura.
- XAdES-X: inclouen un segon segell de temps per robustir la signatura.
- XAdES-XL: afegeix els certificats, així com, l'estat de revocació dels mateixos al document signat, per tal de permetre la verificació en un futur en el cas de que les fonts originals de consulta no estigueren disponibles.

En aquest projecte, tal i com, s'ha comentat utilitzarem XAdES-EPES amb signatura tipus Enveloped, formats que són perfectament suportats com es pot veure per les esmentades llibreries. En aquest sentit encara que ha calgut fer modificacions sobre les llibreries originals per adaptar-les a la plataforma mòbil, les llibreries, s'ajusten perfectament a les nostres necessitats.
Bàsicament les modificacions que ha calgut fer són relatives a les llibreries criptogràfiques de Bouncy Castle que utilitzen les anteriors, doncs no són suportades per la plataforma Android, no obstant existeix un re-empaquetat d'aquestes per la plataforma. Projecte anomenat Spongy Castle que s'integra perfectament amb Android, en aquest sentit ha calgut substituir en les llibreries MITyCLibs unes llibreries per les altres, així com fer petites adaptacions.

Les llibreries del MITyCLibs són programari lliure i estan llicenciades sota llicència LGPL.

## 2.5. Spongy Castle

Tal i com s'ha comentat anteriorment, les llibreries de Spongy Castle són una adaptació a Android de les llibreries criptogràfiques de Bouncy Castle, les quals són utilitzades per les API, MITyCLibs del MINETUR. Aquest és el motiu per al qual s'ha decidit emprar aquestes llibreries.
Per altra banda, els canvis que incorporen aquestes respecte les llibreries de Bouncy Castle són els següents:

- Els paquets de classes org.bouncycastle.* han estat moguts a org.spongycastle.*, per evitar conflictes de càrrega de classes.
- El proveïdor de seguretat de l'API de Java en comptes d'ésser: BC, passa a ser SC.
- No es realitzen canvis en els noms de les classes.

Les llibreries són publicades en el repositori Maven Central, per lo que la incorporació en el projecte al utilitzar Maven ha estat bastant sencilla.

Aquestes llibreries són també programari lliure, i es troben llicenciades sota llicencia MIT X11.

## 2.6. SQLite

El sistema operatiu Android, així com, altres sistemes operatius de plataformes mòbils porten incorporat en el sistema, la BBDD lleugera: SQLite.
SQLite és una base de dades relacional, el motor de la qual està contingut i implementat en una llibreria escrita en llenguatge C i que implementa l'estàndard SQL92. Té com a característica principal que el procés de servidor (de la BBDD) i el client de la BBDD, no són processos separats, sinó que aquest darrer forma part integral del procés en sí.
Per altra banda, la base de dades es multi-plataforma i la seva creació, implica la creació d'un únic fitxer dintre del mateix sistema de fitxers de la plataforma utilitzada. A més a més, aquesta garanteix que les seves transaccions compleixen els principis ACID.
En el nostre cas, utilitzarem SQLite per desar les dades de les factures processades, així com, el fitxer XML rebut per el sistema de descobriment de factures, i poder accedir després a aquestes en mode consulta.

SQLite és programari lliure, i està llicenciada amb llicència GPL.

## 2.7. ORMLite

ORMLite és un paquet o llibreria Java de caràcter lleuger que permet el mapeig de les entitats i relacions de la BBDD relacional en objectes Java del domini o model.
El motiu de la seva utilització és l'abstracció de les operacions SQL, simplement les classes del domini del model es marquen amb anotacions que després seran processades i utilitzades per dur a terme la creació de la BBDD, a la qual una vegada creada, s'accedirà mitjançant el patró DAO, i a través d'aquest, es realitzaran operacions CRUD sobre el model d'objectes.

ORMLite és també programari lliure.

## 2.8. SQLCipher

SQLCipher és una llibreria que proveeix de forma transparent, el xifrat de la BBDD SQLite utilitzant l'algorisme d'encriptació AES de 256 bits.
En el nostre cas, hem utilitzat aquesta llibreria integrant-la amb ORMLite, per xifrar tot el contingut de la BBDD.
SQLCipher es programari lliure i està llicenciada sota llicència, BSD.

## 2.9. Maven

Maven és un software per la gestió i construcció del projecte, permet realitzar tasques de construcció i desplegament, i és àmpliament utilitzat en el mon professional. Aquesta tecnologia utilitza repositoris centralitzats de llibreries i plugins per tal de dur a terme la gestió de dependències del projecte.
En el nostre cas, l'hem utilitzat per estructurar el projecte, gestionar les dependències i fer la construcció del projecte.
Maven és programari lliure, i està llicenciat sota llicència Apache versió 2.0.

## 2.10. BlueCove

BlueCove, es tracta d'una llibreria Java que permet la implementació de comunicacions Bluetooth en diverses plataformes, Windows, Linux i MACOS. En el nostre cas, hem utilitzat l'esmentada llibreria per tal de implementar un servidor d'arxius Bluetooth en el mòdul de descobriment de factures.

BlueCove es programari lliure i està llicenciat sota llicències Apache versió 2.0 i GNU GPL.

## 2.11. Androlog

Androlog, es un component o llibreria que permet implementar un sistema de registre de traces personalitzable, i amb diferents nivells de registre de traces (Debug, Info...). A més a més, permet habilitar i deshabilitar el sistema quan ho creiem oportú.

Androlog és programari lliure i està llicenciat sota llicència Apache 2.0.

# 3. Arquitectura de l'aplicació

En aquest apartat es descriuran tots aquells aspectes relatius a l'arquitectura més rellevants.

## 3.1. Requeriments del projecte

Els requeriments per la construcció del projecte són els següents:

- Màquina Virtual de Java (JVM) 6.0 o superior.
- Eclipse versió Juno o superior (entorn integrat de programació).
- Plugin ADT 21.x o superior (l'Android Development Tools).
- Maven 3.0.
- Plugin d'eclipse, m2eclipse, per la integració amb Maven 3.0.
- Versió d'Android IceScream-Sandwich, 4.0 (API Level 14) o superior.
- Sistema operatiu amb Bluetooth Stack (MS Windows), o bé, amb APIS de BlueZ (Linux).

La utilització de Eclipse, la JVM venen condicionats per la plataforma Android.

Per altra banda, l'aplicació s'ha desenvolupat sobre sistema MACOSX, tant però podríem desenvolupar sobre qualsevol altre plataforma (Linux o Windows).

S'ha testejat en un terminal mòbil Samsung Galaxy S3.

En quant al servidor de Bluetooth, ha requerit lògicament d'un adaptador de bluetooth (dispositiu hardware), i s'ha testejat sota el sistema operatiu Linux Fedora 18 x64 i sota Windows 7 (per tal de que funcioni sobre Windows cal executar-lo amb la JVM de 32 bits).

## 3.2. Mòduls del projecte

L'aplicació s'ha estructurat en varis mòduls de programari, els diferents mòduls de programari es poden observar en la següent figura:

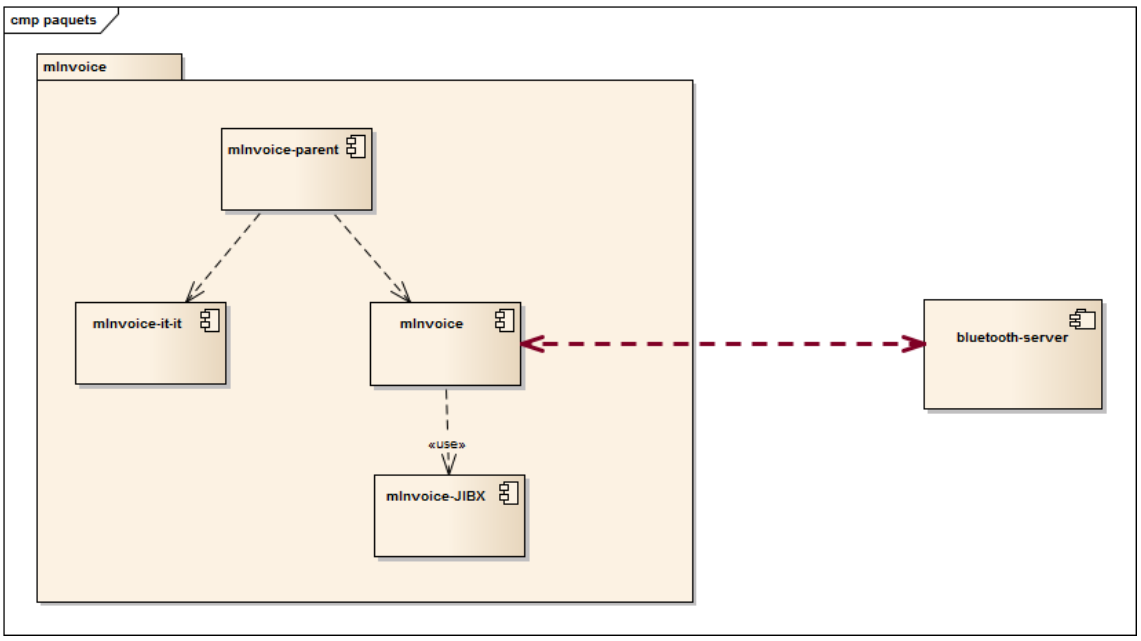


**Figura 2: Mòduls del programari**

Els diversos mòduls es descriuen a continuació:

| Nom del mòdul | Descripció |
|---|---|
| mInvoice-JIBX | L'objectiu del mòdul és generar un model de classes Java a partir dels XSD de Factura-e (v3.2). Aquest mòdul és utilitzat per el mòdul mInvoice (l'aplicació Android), per dur a terme la des-serialització del XML al model d'objectes Java. |
| mInvoice-parent | És el mòdul Maven pare de l'aplicació en sí. |
| mInvoice-it-it | Mòdul de proves unitàries que utilitza les llibreries JUnit per la realització d'aquestes. Malgrat això, només s'ha utilitzat per fer la prova unitària de generació de claus per al xifrat de la BBDD de credencials. És mòdul fill de mInvoice-parent. |
| mInvoice | És l'aplicació en sí, el producte final, es fill de mInvoice-parent. |
| bluetooth-server | Mòdul de servidor, implementat per enviar factures al mòbil mitjançant comunicació bluetooth. |

**Taula 3. Mòduls del projecte**

Comentar que inicialment s'han creat dos proves de concepte o mòduls per comprovar la viabilitat del projecte, *mInvoice – XML JIBX Unmarshalling* i *mInvoice-Xades- XML Validation*, però aquests finalment no han estat lliurats per tractar-se només de proves.

Per altre banda que totes les llibreries afegides al projecte, així com, les construccions dels mòduls es gestionen amb Maven (a excepció de les dos proves de concepte que tenen les llibreries incorporades).

Així doncs cada mòdul d'aquests te incorporat un fitxer, pom.xml (Project Object Model) en el que s'especifiquen a més de les dependències, els plugins utilitzats per la construcció. Vegis a continuació com a exemple un fragment de fitxer:

```
<modelVersion>4.0.0</modelVersion>

<parent>
	<groupId>edu.mistic.pfm</groupId>
	<artifactId>mInvoice-parent</artifactId>
	<version>0.0.1-SNAPSHOT</version>
</parent>


<artifactId>mInvoice</artifactId>
<packaging>apk</packaging>
<name>mInvoice - Application</name>

<dependencies>
	<!-- APIs Android -->
	<dependency>
		<groupId>com.google.android</groupId>
		<artifactId>android</artifactId>
		<scope>provided</scope>
	</dependency>
	<dependency>
		<groupId>com.google.android</groupId>
		<artifactId>support-v4</artifactId>
```

```
        <version>r7</version>
        <scope>provided</scope>
</dependency>
<dependency>
        <groupId>de.akquinet.android.androlog</groupId>
        <artifactId>androlog</artifactId>
</dependency>
<!-- ORM -->
<dependency>
        <groupId>com.j256.ormlite</groupId>
        <artifactId>ormlite-android</artifactId>
        <version>4.44</version>
</dependency>
<dependency>
```

Com es pot veure la gestió de dependències es bastant senzilla, essent suficient afegir unes línies de codi per declarativament afegir la llibreria al *classpath* de l'aplicació.

Això sí, cal tenir configurades en les preferències de Maven l'adreça del repositori públic que estiguem utilitzant.

Per últim en lo que respecta a aquesta secció, cal dir que el producte final que s'obtindrà del resultat de la construcció, és un arxiu APK que serà distribuït sobre el dispositiu mòbil.

## 3.3. Estructura de l'aplicació (mòdul principal)

L'aplicació, mòdul *mInvoice*, la qual permet la construcció del producte final, com a aplicació Android està estructurat de la següent forma:

- Carpeta *src/main/java*, conté el codi font de totes les classes que conformen l'aplicació.
- Carpeta *src/test/java*, carpeta destinada a proves unitàries.
- Carpeta *androlog*, conté el fitxer *androlog.properties*, de configuració de les propietats del sistema de registre de traces.
- Carpeta *assets*, aquesta carpeta en Android s'utilitzarà per desar fitxers que després s'inclouran en el empaquetat, tal i qual, sense cap mena d'alteració. En el nostre cas, s'ha utilitzat per guardar els certificats de les CAs de confiança, així com, un magatzem de certificats buit sobre el qual es construirà el truststore final de l'aplicació, i el fitxer *icudt46l.zip* utilitzat per *SQLCipher*.
- Carpeta *gen*, on es guarden el codi Java generat per l'ADT.
- Carpeta *bin*, carpeta utilitzada per l'IDE per desar els fitxers resultants de la construcció.
- Carpeta *libs*, contindrà les llibreries del projecte que no són gestionades amb Maven. Per exemple, les llibreries natives de SQLCipher.
- Carpeta *res*, contindrà els fitxers de recursos d'Android que utilitza l'aplicació, icones, fitxers de definició de literals de l'aplicació, *layouts*, etc.
- Carpeta *target*, carpeta utilitzar per Maven per desar els fitxers resultants de la construcció.
- Fitxer *AndroidManifest.xml*, fitxer de control que defineix el tipus d'aplicació, així com, components que s'utilitzen en aquesta.
- Fitxer *pom.xml*, com ja s'ha comentat anteriorment, utilitzat per Maven per gestionar les dependències i construcció del projecte.
- Fitxer *project.properties*, fitxer generat per l'ADT, amb propietats de l'aplicació Android.

## 3.4. Paquets de l'aplicació

El mòdul *mInvoice*, se estructura en diversos paquets que composen aquest, a continuació es descriuen els diversos paquests:

| Nom del paquet | Descripció |
|---|---|
| *edu.mistic.pfm.mInvoice* | Paquet principal del qual pengen la resta de paquets. A més a més, en ell s'inclouen les activitats (components gràfics) que composen l'aplicació Android. |
| *edu.mistic.pfm.mInvoice.base* | Conté classes abstractes i interfícies de les quals estenen altres classes. Com per exemple la classe abstracta Identificable de la qual hereten totes aquelles classes que tenen un identificador (ID). |
| *edu.mistic.pfm.mInvoice.bluetooth* | Conté les classes que implementen el client per establir la comunicació bluetooth des del mòbil i que forma part del mòdul de descobriment de factures. |
| *edu.mistic.pfm.mInvoice.db* | Paquet base que conté les classes que permeten accedir a la BBDD, SQLite. |
| *edu.mistic.pfm.mInvoice.db.sqlcipher* | Adaptació de les classes de ORMLite per tal de integrar-les amb SQLCipher, i poder xifrar la BBDD. |
| *edu.mistic.pfm.mInvoice.dto* | Paquet que conté tots els Transfer Objects utilitzats per l'aplicació. |
| *edu.mistic.pfm.mInvoice.model* | Conté les classes del model i que són persistides (totes les classes estan anotades amb anotacions ORMLite) |
| *edu.mistic.pfm.mInvoice.pki.base* | Conté la classe encarregada de comprovar la validesa de la signatura del document XADES. |
| *edu.mistic.pfm.mInvoice.pki.ocsp* | Conté les classes encarregades de realitzar les validacions OCSP. |
| *edu.mistic.pfm.mInvoice.pki.policies* | Conté la classe encarregada de definir la política de la validesa de la signatura en base a les dades del certificat i l'actual. |
| *edu.mistic.pfm.mInvoice.pki.truster* | Conté els validadors de confiança de l'aplicació, així com, la seva factoria. |
| *edu.mistic.pfm.mInvoice.properties* | Conté una classe útil, utilitzada per accedir als fitxers de propietats de l'aplicació |
| *edu.mistic.pfm.mInvoice.service* | Conté les interfícies dels Serveis o Business Objects utilitzats a l'aplicació. |
| *edu.mistic.pfm.mInvoice.service.impl* | Conté la implementació dels serveis (interfícies). |
| *edu.mistic.pfm.mInvoice.ui* | Conté components de la UI, bàsicament els Fragments utilitzats. |
| *edu.mistic.pfm.mInvoice.ui.adapters* | Conté els adaptadors utilitzats en els components gràfics *'llista'*, que permeten vincular una col·lecció d'objectes a la llista |

| | |
|---|---|
| | gràfica. |
| *edu.mistic.pfm.mInvoice.ui.listeners* | Conté els escoltadors d'events que es desencadenen quan l'usuari fa una acció sobre un component, per exemple, en fer clic sobre un botó o un element d'una llista... |
| *edu.mistic.pfm.mInvoice.util* | Conté classes d'utilitats, com per exemple una classe útil per formatar de dates, un conversor d'objectes del model de persistència a DTO... |
| *edu.mistic.pfm.mInvoice.xml* | Conté classes útils per processar el document XML (Factura-e) i convertir-lo a objectes Java. |

**Taula 4. Paquets de l'aplicació**

## 3.5. Altres consideracions

Només fer una breu pinzellada per el disseny del programari, en l'aplicació s'han seguit diversos patrons de programari:

- El patró DAO (Data Access Object): per implementar la capa d'accés a dades.
- El patró DTO (Data Transfer Object): per la transferència de dades entre diverses capes o interfícies.
- El patró BO (Business Object): per implementar la capa que conté la lògica de negoci.

El motiu per utilitzar aquests patrons de disseny, ha estat el de desacoblar en la mida de lo possible la interfície gràfica, de la capa de negoci i la capa d'accés a dades.

Per implementar la lògica de negoci s'han utilitzat interfícies, les quals proveeixen la funcionalitat pertinent mitjançant les seves classes d'implementació.
L'accés a la capa de persistència es realitza a través de DAOs que són consumits per els objectes de negoci a traves dels "DataBase Helpers".
La transferència de dades entre unes capes i d'altres es realitza a través d'objectes DTO, no obstant això afegeix una complicació addicional, i és que aquests DTOs s'han de transformar o convertir en objectes del model de persistència a l'hora de ser emmagatzemats a la BBDD.

# 4. Validació de factures

En aquesta secció es pretén presentar com s'ha desenvolupat el mòdul de validació de factures digitals, el qual està compost per el subsistema de validació del format XML del document digital i per la part de validació de la signatura digital. S'explicarà allò més rellevant, així com, les decisions de disseny i implementació dutes a terme.
Cal dir, que aquest mòdul, és un dels mòduls més rellevants de l'aplicació, i sobre la qual és sustenta tot el sistema, per altra banda, és important assenyalar que totes les validacions dutes a terme, és duen des del mateix terminal mòbil, d'aquesta forma és garanteix la seguretat en no sortir en cap moment la informació del terminal mòbil.

## 4.1. Validació de la signatura digital

El primer pas en la recepció del tiquet o factura digital per el mòdul de comunicacions o descobriment de factures, és la validació de la signatura digital que es realitza tot just abans de

processar el document XML.

### 4.1.1. Descripció funcional

Un dels requeriments principals del sistema de tiqueting és que la factura telemàtica mòbil tingui validesa legal i jurídica, garantint propietats com l'autenticació de la mateixa i el no repudio, per aquest motiu s'utilitzarà la signatura avançada reconeguda per signar el document. Aquesta signatura haurà d'ésser validada per ésser posteriorment acceptada, en aquest apartat es descriurà funcionalment i a alt nivell, el procés de validació de la signatura digital.

Com s'ha comentat, el tipus de document que el mòdul espera rebre és una factura (format Factura-e) signada digitalment utilitzant el format XADES-EPES, i de tipus 'Enveloped'.

La validació de la signatura implica:

1. Validar que el document no ha estat alterat posteriorment a la signatura.
2. Validar la vigència de la data de la signatura, és a dir, la data actual de la signatura haurà d'estar compresa entre les dates de validesa del certificat emprat per la signatura. És a dir, aquesta haurà d'estar compresa entre la data 'no abans de' i la data de 'caducitat' d'aquest.
3. Validar la confiança del certificat, per dur a terme aquesta validació caldrà tenir instal·lats els certificats arrel de les entitats certificadores de confiança (que són les expedidores dels certificats) dels certificats admesos. Amb aquesta finalitat l'aplicació haurà de tenir un magatzem de confiança (truststore), de forma que quan es valida una signatura, s'analitzarà la seva confiança, en funció de si el certificat (o certificats de la cadena) ha estat expedit per alguna de les entitats de les quals tenim instal·lats els certificats a l'esmentat magatzem.
4. Validar si el certificat ha estat revocat o no, per dur a terme aquesta validació s'utilitzarà el protocol OCSP, connectant contra un servidor OCSP, per tal, de validar l'estat del certificat. Les respostes que pot donar l'esmentat servidor, són: GOOD, REVOKED o UNKNOWN, en funció de l'estat de confiança del certificat.

Per dur a terme les validacions anteriors, l'aplicació incorpora (a banda d'altres components):

1. Component de validació de la política de dates.
2. Certificats de les CAs de confiança, actualment l'aplicació suporta els certificats de DNIe i de la FNMT ("Fàbrica Nacional de Moneda y Timbre"), encara que es podrien instal·lar els consideressin adients.
3. Connexió amb OCSP: `http://av-dnie.cert.fnmt.es/ocsp-a/OCSPServlet`, aquest servidor permet validar certificats de la FNMT i DNIe. La idea és que un futur sigui totalment configurable.

### 4.1.2. Components externs

Per tal de dur a terme la validació s'han incorporat les llibreries MITyCLibs (MINETUR) i les llibreries de Spongy Castle, les versions de llibreria utilitzades són, 1.0.4 i la 1.47.0.3 respectivament.

Respecte a la primera, MITyCLibs, els components utilitzats han estat els següents:

- *MITyCLibAPI* és tracta d'una llibreria que conté la funcionalitat base comú a la resta de components de signatura del MINETUR.
- *MITyCLibOCSP* es tracta d'una llibreria que conté la funcionalitat necessària per realitzar les validacions OCSP.
- *MITyCLibTrust* aquesta llibreria proporciona implementacions concretes de validadors de confiança, així com, permet reconstruir cadenes de certificació a partir del certificat proporcionat.
- *MITyCLibTSA* es tracta d'una llibreria que conté la lògica necessària per obtindré i processar segells de temps. Aquesta llibreria s'ha inclòs en el projecte per dependència de la resta de llibreries, però en aquest projecte no s'ha implementat 'segell de temps'.
- *MITyCLibXADES* es tracta d'una llibreria que proporciona la funcionalitat necessària per processar fitxers XML, així com, per l'ús de certificats, la realització de signatures XADES i la seva validació.

Totes aquestes llibreries (MITyCLibs) s'han hagut de modificar, ja que per una banda, inicialment utilitzaven les APIs de Bouncy Castle i per un altre utilitzaven una versió anterior a la 1.47 d'aquesta darrera. Per realitzar les modificacions ha calgut:

- Fer la baixada del codi font de les llibreries del MINETUR de la següent ubicació: http://oficinavirtual.mityc.es/componentes/downloads/componentes-1.0.4-sources.zip

- Modificar codi font anterior, duent a terme la migració a la versió 1.47 de les API de Bouncy Castle. Per realitzar les modificacions s'ha seguit el document de Bouncy Castle següent:

  http://www.bouncycastle.org/wiki/display/JA1/Porting+from+earlier+BC+releases+to+1.47+and+later

  Bàsicament les modificacions han estat degudes per el canvi de certes classes utilitzades per unes de noves, així com, perquè certs mètodes havien canviat.

- Modificar el codi font per tal de canviar la utilització dels paquets *org.bouncycastle* per la utilització dels paquets *org.spongycastle*.
- Una vegada realitzades les modificacions, s'han empaquetat els mòduls mitjançant Maven i s'han afegit les dependències al nostre projecte.

### 4.1.3. Disseny i implementació

Cal comentar que inicialment s'ha dut a terme una prova de concepte, anomenada: "mInvoice – Xades – XML Validation", en aquesta prova de concepte s'ha dut a terme la validació de la signatura digital, així com, de diverses factures en el terminal mòbil. Veient que és factible realitzar la validació en dispositius mòbils, s'ha tirat endavant la resta del projecte.

En primer lloc ha calgut incorporar les llibreries esmentades en el punt 1.1.2 a l'aplicació (veure Annex: Incorporació de llibreries amb Maven). Posteriorment i prenent com a punt de partida el prototip esmentat ha calgut desenvolupar tota una sèrie de classes encaminades a implementar la validació de la signatura digital. Aquestes classes que intervenen en el procés de validació, estan agrupades sota el paquet: `edu.mistic.pfm.mInvoice.pki`.
No obstant, abans de veure i descriure les classes que participen en la validació, veiem primer els diversos recursos que intervenen en aquesta:

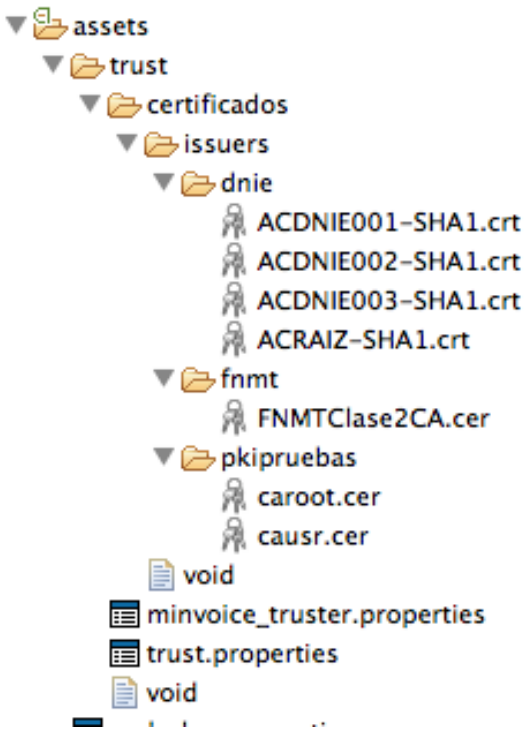


**Figura 3. Recursos utilitzats per la validació de signatura**

Sota el directori `assets`, trobarem els fitxers:

- `void`: magatzem de certificats de tipus BKS (detalls de com realitzar la creació d'aquest, en annex), servirà per construir el magatzem de confiança final/principal (`minvoice.truststore`), a partir dels certificats de les CA que es trobem a `assets/certificados/issuers`.

- `trust.properties`: fitxer de propietats utilitzat per la factoria que gestiona les instancies dels validadors de confiança. Bàsicament en aquest fitxer s'especifica el validador de confiança de l'aplicació:

```
mistic.SignCerts=none
mistic.CRLEmisor=none
mistic.OCSPProducer=none
mistic.TSProducer=none
mistic.All=edu.mistic.pfm.mInvoice.pki.truster.CustomTruster
```

- `minvoice_truster.properties`: en aquest fitxer es defineix el nom del magatzem de certificats de CAs de confiança, utilitzat per l'aplicació (`minvoice.truststore`).
- `assets/certificados/issuers`: directori del qual pengen els certificats de les autoritats certificadores de confiança, aquelles en les que confiem de cara a realitzar la validació de confiabilitat. Actualment, com s'ha comentat en el punt, descripció funcional, suporta DNIe i certificats emesos per la FNMT.

Passem a veure el diagrama de classes JAVA que intervenen en la validació de la confiança del certificat, vegis la següent figura:

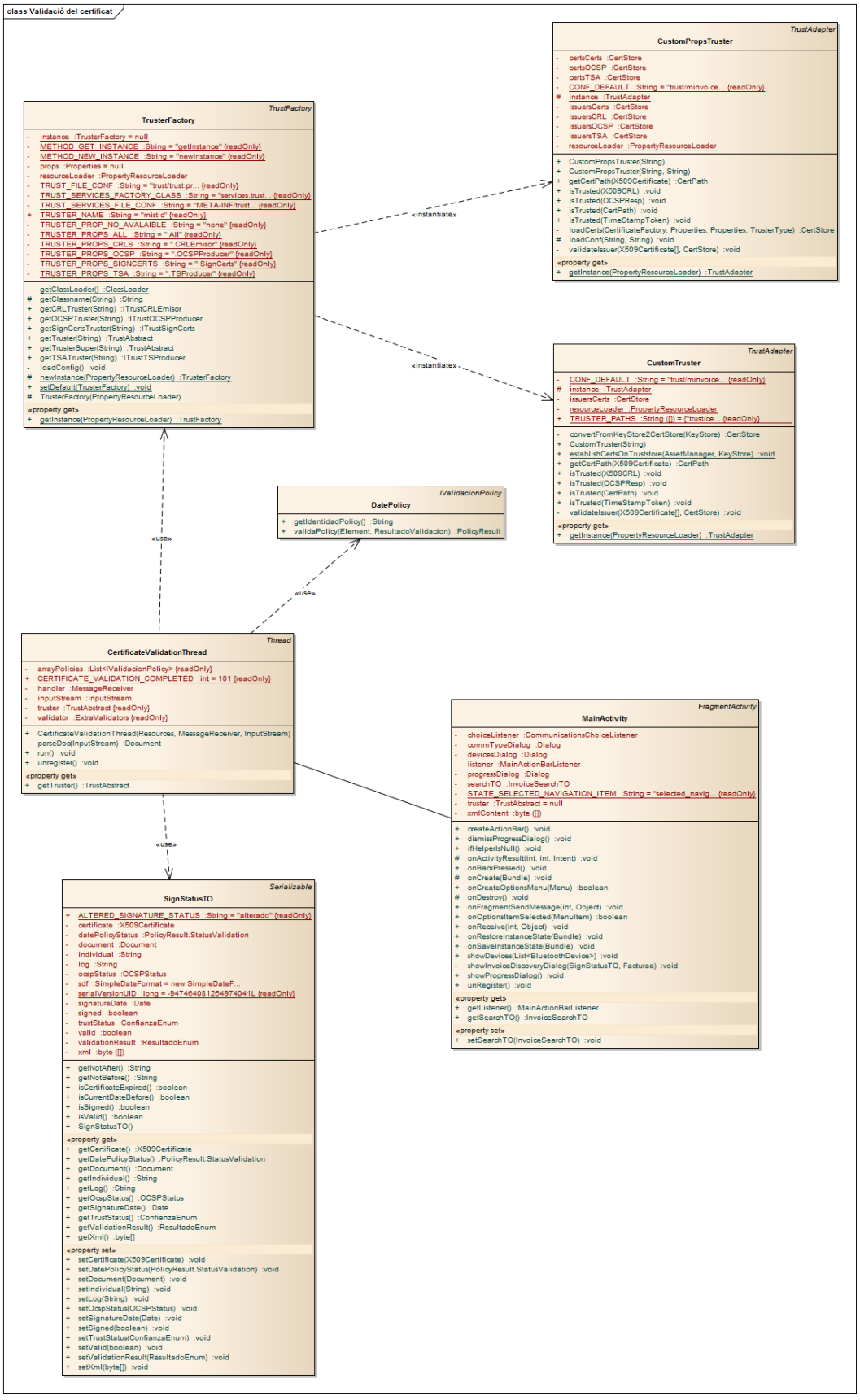

**Figura 4. Diagrama de classes del subsistema de validació de la confiança del certificat**

Passem a descriure el procés de validació del certificat, utilitzant el diagrama de classes anterior, veiem:

1. L'activitat principal `MainActivity` (component de la UI de l'aplicació) desencadena el procés de validació a partir del moment en que el subsistema de descobriment de factures notifica que ha rebut el document o factura.
2. Es crea un nou `Thread` instanciant un objecte de tipus `CertificateValidationThread`. A aquest objecte se li passa un `InputStream` amb el contingut del fitxer XML a processar. Aquest `Thread` serà l'encarregat, utilitzant diverses classes (esmentades en el diagrama), així com, utilitzant les llibreries MITyCLibs, de realitzar la validació del document.
3. El `Thread` creat en el punt anterior, obté a partir de la factoria `TrustFactory` un validador de confiança, objecte `CustomTruster` (o `CustomPropsTruster`). Cal senyalar que la `TrustFactory` és un objecte singleton, que rep en el moment de recuperar la instància un objecte de tipus `PropertyResourceLoader`, a través del qual podrà llegir les propietats dels fitxers de propietats de l'aplicació (Figura 2), amb la finalitat de determinar on es troba el magatzem de certificats de confiança de les CA (a utilitzar per validar la confiabilitat de la signatura).
4. A continuació (en el thread) es crea la política de validació de dates, `DatePolicy`. Aquesta política permetrà realitzar a posteriori la validació de que el certificat utilitzat per la signatura es vigent amb respecte la data actual.
5. A partir dels objectes, `Truster` i `DatePolicy`, es crea un objecte validador, pertanyent a la classe, `es.mityc.firmaJava.libreria.xades.ExtraValidators`, de les llibreries MITyCLibs.
6. Es processa ("parseja") el XML amb el contingut de la factura, obtenint un objecte: `org.w3c.dom.Document`.
7. Es realitza la validació de la signatura del document utilitzant el validador esmentat en el punt 5, i l'objecte, `Document`, esmentat en el punt 6:

   ```
   final ValidarFirmaXML vXml = new ValidarFirmaXML();
   results = vXml.validar(doc, "./", validator);
   ```

   Dit d'un altre forma, per realitzar la validació, s'instancia un objecte `ValidarFirmaXML`, i es crida la funció `validar(..)`, passant com a paràmetres el document i el validador.

8. Es processa la resposta i es crea un objecte `SignStatusTO` (Transfer Object), amb el resultat de la validació i amb el propi document validat.
9. El thread notifica a l'activitat principal, `MainActivity`, el final del procés, retornant-li a aquesta el Transfer Object creat prèviament (punt 7).
10. Per finalitzar, l'aplicació desencadena la comprovació OCSP, o bé, mostra la pantalla amb el resultat de la validació a l'usuari, depenent de les preferències establertes.

La comprovació OCSP es realitza a continuació sempre que a les preferències d'usuari, la preferència `isOcspEnabled`, tingui el valor 'true' (per defecte està inicialitzada amb aquest valor).

Per altra banda, la validació OCSP requereix dels següents permisos de configuració a nivell d'aplicació Android:

```
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.INTERNET"/>
```

Això ha d'ésser així, ja que per realitzar la comunicació OCSP es requereix accedir a la xarxa de dades a la que està connectat el terminal mòbil.

El diagrama de les classes que intervenen en la comprovació OCSP és el següent:

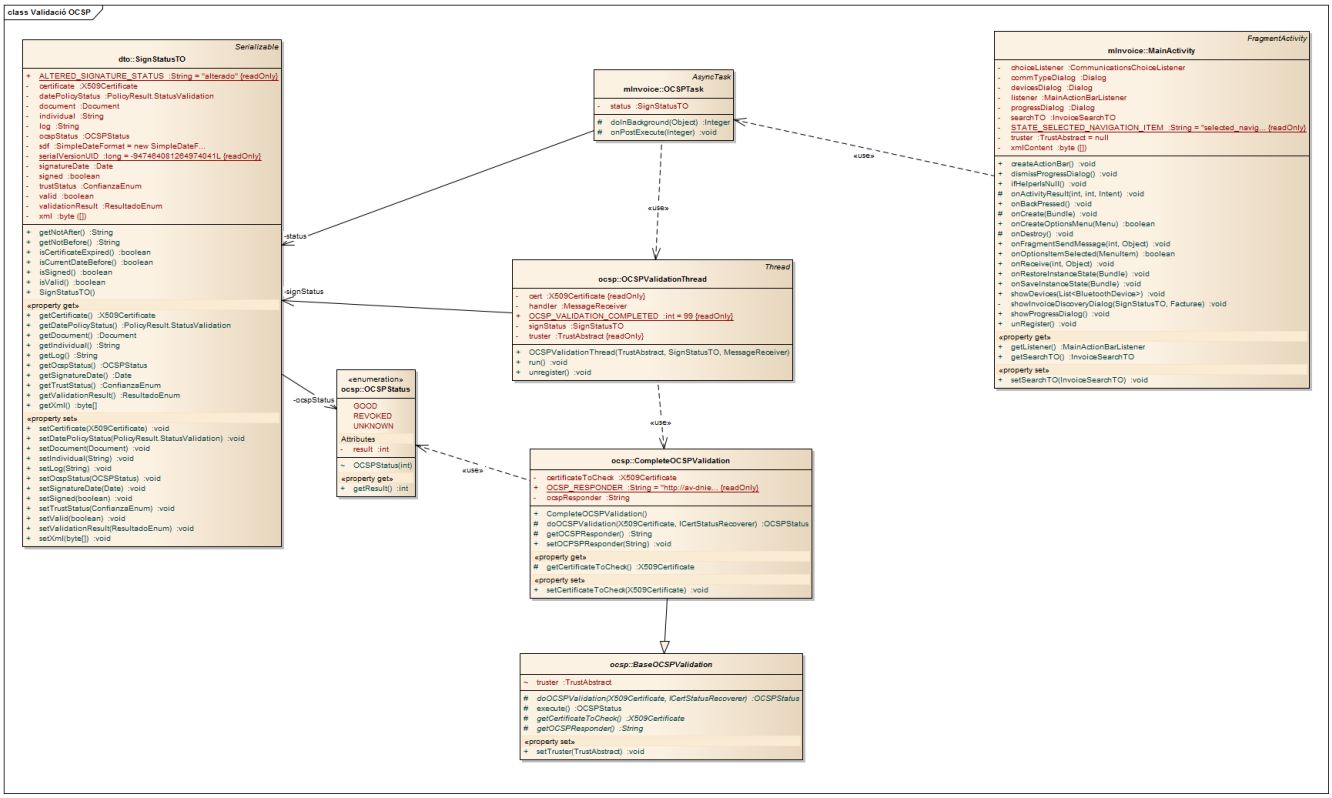


**Figura 5. Diagrama de classes del subsistema de comprovació OCSP**

A partir del model de classes descriurem el procés de validació OCSP:

1. L'activitat principal, `MainActivity`, rep notificació de finalització de validació de signatura. Es consulta el "flag" de validació OCSP, en cas de valdre, 'true', s'inicia el procés de comprovació creant una `OCSPTask` passant-li com a paràmetres durant la construcció un objecte `SignStatusTO` (amb el document i les dades de validació prèvies). Aquesta tasca és de caràcter asíncron i s'executarà en segon pla. El motiu de la execució en segon pla, és que el sistema no permet l'execució de tasques que accedeixen als recursos de xarxa des del Thread principal sobre el que corre la UI.
2. La tasca, `OCSPTask`, crea un nou `Thread` del tipus, `OCSPValidationThread`, i se li passa a aquest com a paràmetres, el *transfer object*, `SignStatusTO`.
3. El Thread creat en el punt anterior, instància un objecte `CompleteOCSPValidation`, que serà l'encarregat de dur a terme la validació, utilitzant les llibreries MITyCLibs.

4. Un cop finalitzada la comprovació, es processa la resposta i es crea un objecte `OCSPStatus` amb el resultat de la validació, aquest objecte serà establert com a atribut de l'objecte `SignStatusTO`.
5. Per finalitzar, es notifica a l'activitat principal la finalització del procés, i aquesta mostra a l'usuari una pantalla amb el resultat de les validacions.

#### 4.1.3.1. Altres consideracions

Dos consideracions importants en lo que respecta a la implementació són:

- Per una banda comentar que la generació del truststore BKS a partir dels certificats de les CAs, es realitza quan per primer cop accedeixes a l'aplicació, a partir d'aquí s'estableix una preferència a l'aplicació per mitjà d'una variable booleana que evitarà que els següents cops que s'accedeix a l'aplicació es torni a crear el magatzem de certificats.
- Ha calgut reempaquetar per la plataforma Android, les llibreries de Apache Xerces (versió 2.9.0) de les quals depenen les del MINETUR, re-definint els noms dels paquets de *org.apache.** a *edu.mistic.org.apache.**. Això s'ha fet en un projecte creat amb aquest propòsit anomenat *xerces_repackage* (es poden consultar els detalls en el següent annex).
- També seguint el mateix mètode que en el cas anterior, ha calgut re-empaquetar les llibreries de Apache XML Security (versió 1.4.7), generant la llibreria "xmlsec-ae-1.4.7.jar".

## 4.2. Validació del XML (Factura-e)

En aquesta secció es descriurà tot allò relatiu a la validació o processat de la factura o tiquet (del sistema de pagament) que és enviada al dispositiu mòbil per el mòdul de descobriment de factures.

### 4.2.1. Descripció funcional

Tal i com ja s'ha comentat, en un sistema de pagament electrònic de caràcter mòbil es fa necessari lliurar un comprovant de pagament, lo que coneixem en el mon habitual com factura o tiquet, en aquest sentit es fa necessari traslladar amb mitjans telemàtics el mateix concepte a la plataforma mòbil.
Per tal d'aconseguir aquests objectius, i en lo que respecta al lliurament del document i la informació que conté, el processat i el format d'aquest, s'han de complir els següents requeriments:

- Que l'element electrònic permeti reflectir tots els conceptes del món real, de forma que quedin reflectits conceptes com: l'emissor de la factura (persona física o jurídica), els imports, taxes i descomptes aplicats, el desglossament de la compra reflectint els preus i la quantitat de producte comprat, les dades del receptor (comprador) i escau, etc.
- Preferiblement que la factura segueixi un estàndard de document electrònic.
- Sota aquest enfoc, que es compleixin requisits de seguretat adients, per garantir la confidencialitat i integritat de la mateixa.

Tots aquests requeriments han condicionat la tria del format o tipus de factura telemàtica, escollint el format de document Factura-e (format XML) en la seva darrera versió aprovada 3.2. Per altra banda, el fet de que s'hagi de garantir la confidencialitat i integritat de la mateixa, comporten el fet de que la validació del document lliurat es realitzi en el mòbil.
S'haurà de validar que el document XML estigui ben format i que segueixi les especificacions de Factura-e, així com, s'haurà de transformar en un model d'objectes Java per al posterior guardat de la informació en la BBDD.

### 4.2.2. Disseny i implementació

Per tal desenvolupar el sistema de validació del document format Factura-e, s'han avaluat diferents alternatives: utilitzar les API d'Android (XMLPullParser), utilitzar APIs ben conegudes en el mon Java (JAXB), utilitzar les API del MINETUR i el Ministeri d'Hisenda, o cercar altres alternatives. Utilitzar les API de JAXB o del MINETUR/ Hisenda és inviable pel fet de que la portabilitat a Android és impossible, utilitzar les API d'Android comportarien una feina enorme donat el temps reduït, així que, s'han estudiat altres alternatives, i finalment s'han triat les API de JIBX.
Aquestes API després d'una prova de concepte, han demostrat molts bon resultats i un rendiment més que acceptable.

Així doncs, han tingut que ésser portades a la plataforma Android, per portar-les a la plataforma s'han seguit els següents passos:

1. Incorporar les llibreries: xml-apis (versió 1.4.01) de Xml Commons re-empaquetant-les per Android. Lo que ha significat l'eliminació de tots els paquets *javax.xml.** això és degut a que Android incorpora els seus propis paquets *javax.xml.**.

2. Afegir les llibreries de JIBX al projecte, tant però, ha calgut a nivell de fitxer de configuració de Maven (pom.xml) excloure totes les dependències de les primeres, això ha estat necessari ja que de lo contrari hi hauria dependències de JIBX que entrarien en conflicte amb les llibreries de la plataforma Android. En el seu lloc s'han afegit una per una al projecte les dependències que són exclusivament necessàries. Podem veure un fragment de la configuració declarativa de Maven important la dependència de JIBX excloent les seves dependències:

```
<!-- Serialització/deserialització de XML -->
<dependency>
	<groupId>jibx</groupId>
	<artifactId>jibx-run</artifactId>
	<version>1.2.3</version>
	<exclusions>
		<exclusion>
			<groupId>*</groupId>
			<artifactId>*</artifactId>
		</exclusion>
	</exclusions>
</dependency>
```

Afegides les llibreries de JIBX a la plataforma, el pas següent ha estat generar a partir del XSD de la versió 3.2 de Factura-e el model de classes sobre les que es des-serialitzarà el document XML per això s'han seguit els següents passos:

1. Construcció de nou mòdul mInvoice-JIBX amb Maven.
2. Descarrega dels XSD del portal de Factura-e i incorporació al mòdul anterior en el directori "src/main/config/".
3. Configuració amb Maven del plugin de JIBX:

```
<plugin>
	<groupId>org.jibx</groupId>
	<artifactId>jibx-maven-plugin</artifactId>
	<version>1.2.4.5</version>
	<executions>
		<execution>
			<goals>
				<goal>bind</goal>
			</goals>
			<configuration>
				<schemaBindingDirectory>
					target/generated-sources
				</schemaBindingDirectory>
				<load>true</load>
				<validate>true</validate>
				<verbose>true</verbose>
				<verify>true</verify>
			</configuration>
		</execution>
	</executions>
</plugin>
```

4. Execució del goal (tasca) de Maven:

```
clean jibx:schema-codegen jibx:bind install
```

Executant aquest goal de Maven, s'ha aconseguit generar el model de classes al directori "target/generated-sources" del mòdul (paquets generats: `es.facturae.facturae.v3.facturae` i `org.w3.xmldsig`; classe base generada: `Facturae`), i empaquetar el mòdul en una llibreria Java (factura32-xml-validation).

A continuació s'ha incorporat la llibreria al projecte/mòdul mInvoice, per fer-ho s'ha instal·lat la llibreria en el repositori local de Maven. I s'ha afegit la dependència:

```
<!-- APIS Factura-e 3.2 con JIBX -->
<dependency>
	<!-- Manually installed -->
	<groupId>edu.mistic.minvoice.jibx</groupId>
	<artifactId>
		factura32-xml-validation
	</artifactId>
	<version>1.0.0</version>
	<exclusions>
		<exclusion>
			<artifactId>*</artifactId>
			<groupId>*</groupId>
		</exclusion>
	</exclusions>
</dependency>
```

Per últim, s'ha implementat una classe útil, `InvoiceXmlValidation`, dintre del paquet: `edu.mistic.pfm.mInvoice.xml`, que permet dur a terme la des-serialització del

XML, a continuació es presenta el detall de la classe en qüestió:

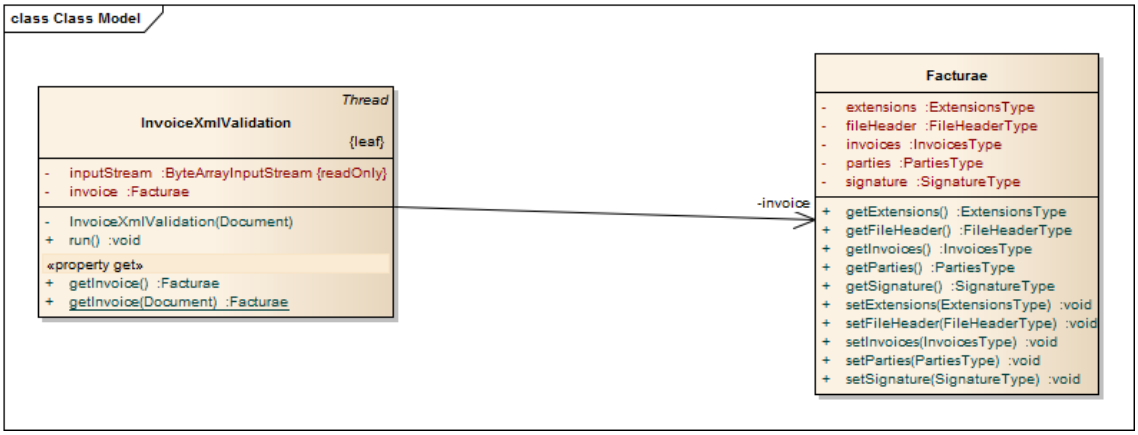


**Figura 6. Diagrama de classes de mètode de validació del XML**

La classe útil esmentada hereta de java.lang.Thread, mitjançant el seu constructor rep el document a des-serialitzar (tipus `org.w3c.dom.Document`), i mitjançant el seu mètode `getInvoice(..)` el qual invoca el mètode `run()`, generarà un objecte `Facturae`, que contindrà totes les dades del XML processat.
Per altra banda, comentar que el codi que permet la recuperació de la factura es bastant senzill, tant però, arribar fins aquí ha estat un procés bastant costós. Vegis a continuació el fragment de codi esmentat:

```
IBindingFactory jc = BindingDirectory.getFactory(Facturae.class);
IUnmarshallingContext unmarshaller = jc.createUnmarshallingContext();
invoice = (Facturae)unmarshaller.unmarshalDocument(
                                        new InputStreamReader(inputStream));
```

Un incís per tal de que aquesta implementació de des-serializador funcioni ha estat necessari especificar la propietat a nivell de sistema:

```
javax.xml.validation.SchemaFactory:http://www.w3.org/2001/XMLSchema
```

amb valor

```
edu.mistic.org.apache.xerces.jaxp.validation.XMLSchemaFactory
```

Definint aquesta propietat s'especifica que la factoria per manipular esquemes XML que segueixen la definició: `http://www.w3.org/2001/XMLSchema`, es la classe `edu.mistic.org.apache.xerces.jaxp.validation.XMLSchemaFactory`, classe de Apache Xerces que hem re-empaquetat per la plataforma Android.

Per últim en lo que respecta a aquesta secció, cal dir, que tota la jerarquia d'objectes Java que pengen de Facturae formen un model bastant extens i complex, per aquest motiu, de cara a treballar amb els objectes generats, s'ha generat la documentació, JAVADOC, aquesta està ubicada dintre del mòdul: *mInvoice-JIBX* dintre del directori *"apidocs"*.

# 5. Emmagatzematge de dades

En aquesta secció es fa referència a l'emmagatzematge d'informació al dispositiu mòbil de forma segura, i única i exclusivament accessible per al propietari de la informació. Aquesta secció es divideix en dos apartats, una descripció funcional on es detallen els requeriments a complir per aquest mòdul, i una secció relativa al disseny i implementació del subsistema, on es detalla com s'han assolit els requeriments esmentats en el punt anterior.

## 5.1. Descripció funcional

Com a requeriments funcionals del projecte, i en lo que respecta a aquest mòdul, existeix la necessitat d'emmagatzemar la informació complint els següents requeriments:

- Que la informació s'emmagatzemi en el dispositiu mòbil de forma permanent, per tal de que en qualsevol moment es pugui accedir a la informació disponible en aquest.
- Que la informació s'emmagatzemi de forma segura, és a dir, garantint les propietats de confidencialitat i integritat de la informació.

## 5.2. Disseny i implementació

Per tal de complir i garantir els requisits anteriors, s'ha utilitzat per una banda el motor de SQLite que porta incorporat el dispositiu mòbil, conjuntament amb les llibreries de SQLCipher, que permeten el xifrat de la informació en el dispositiu mòbil utilitzant l'algorisme d'encriptació AES de 256 bits. L'accés a la BBDD per realitzar operacions CRUD, s'ha realitzat mitjançant ORMLite, tecnologia que permet el mapeig de les entitats i relacions del model a E/R a objectes del model o domini de classes Java.

### 5.2.1. El model E/R

El primer pas a realitzar ha estat dissenyar el model de classes que permetran dur a terme l'emmagatzematge de la informació, aquest model ha estat condicionat per l'estructura de informació del model de classes de les API de Facturae, no obstant per una banda es desa el document XML íntegrament a la BBDD i per un altre en diverses entitats, s'emmagatzemen només aquelles dades exclusivament necessàries per ésser mostrades per l'aplicació a l'usuari. A continuació es presenta el model de dades:

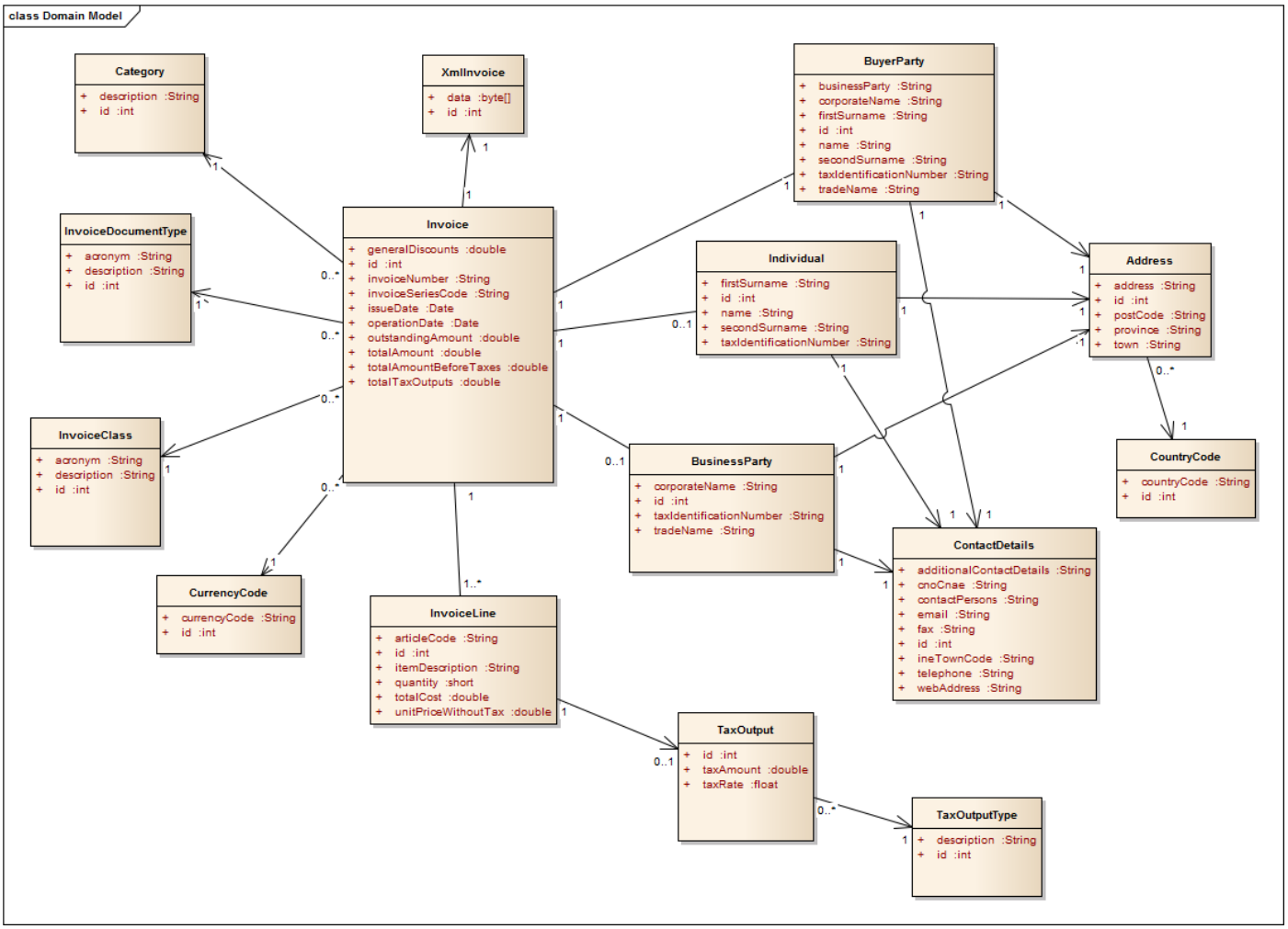


**Figura 7. Domini de classes del model**

Totes aquestes classes del model pertanyen al paquet:

`edu.mistic.pfm.mInvoice.model`,

A continuació de forma breu i alt nivell (doncs no es el propòsit d'aquest PFM) es descriu cadascuna de les classes que composen aquest model:

| Nom de la classe | Descripció |
|---|---|
| Category | Classe que permet classificar les factures (Invoice) i categoritzar-les. Diverses factures poden pertànyer a una mateixa categoria. |
| InvoiceDocumentType | Taula mestra. Defineix el tipus de factura que pot ser AF, FA i FC (Auto Factura, Factura Abreujada o Factura Completa) |
| InvoiceClass | Taula mestra. Defineix la classe de factura:<br>CC<br>Copia recapitulativa.<br>CO<br>Copia original.<br>CR |

| | |
|---|---|
| | Copia rectificativa.<br>OC<br>Original recapitulativa.<br>OO<br>Original.<br>OR<br>Original rectificativa |
| CurrencyCode | Taula mestra. Defineix la moneda utilitzada per la factura. |
| CountryCode | Taula mestra. Defineix el codi de país en que s'emet la factura. |
| TaxOutputType | Taula mestra. Defineix el tipus de taxa a aplicar. |
| Invoice | Emmagatzema les dades de les factures. |
| XmlInvoice | Emmagatzema el document XML (Factura-e) de les diverses factures. |
| InvoiceLine | Emmagatzema les diverses línies de detall o desglossament de les factures. |
| TaxOutput | Emmagatzema els imports i percentatges de les taxes que s'apliquen sobre una línia de desglossament de la factura. |
| BuyerParty | Emmagatzema les dades del comprador vinculat a la factura, ja sigui, una persona física o jurídica. |
| Individual | Emmagatzema les dades del venedor o emissor de la factura, quan aquest és una persona física. |
| BusinessParty | Emmagatzema les dades del venedor o emissor de la factura, quan es tracta d'una persona jurídica. |
| ContactDetails | Emmagatzema les dades de contacte tant de venedors com de compradors. |
| Adress | Emmagatzema les adreces tant de venedors com de compradors. |

**Taula 5. Breu explicació del domini de classes del model**

### 5.2.2. Integració de les diferents tecnologies

El següent pas ha estat dur a terme la integració de les tres tecnologies emprades, SQLite + ORMLite + SQLCipher, per implementar el subsistema de persistència, per assolir aquest objectiu:

1. S'han incorporat les llibreries al projecte mitjançant Maven:

```
<!-- ORM -->
<dependency>
	<groupId>com.j256.ormlite</groupId>
	<artifactId>ormlite-android</artifactId>
	<version>4.44</version>
</dependency>
<dependency>
	<groupId>com.j256.ormlite</groupId>
	<artifactId>ormlite-core</artifactId>
	<version>4.44</version>
</dependency>
<!-- SQLCipher -->
<dependency>
	<groupId>com.google.guava</groupId>
```

```
<artifactId>guava</artifactId>
            <version>r09</version>
        </dependency>
        <dependency>
            <groupId>commons-codec</groupId>
            <artifactId>commons-codec</artifactId>
            <version>1.7</version>
        </dependency>
        <dependency>
            <!-- Manually installed -->
            <groupId>sqlcipher</groupId>
            <artifactId>sqlcipher</artifactId>
            <version>2.1.1</version>
        </dependency>
```

En quant SQLCipher a més a més, ha calgut incloure les llibreries natives:

- libdatabase_sqlcipher.so
- libsqlcipher_android.so
- libstlport_shared.so

En el directori “libs/armeabi” de l’aplicació, així com, el fitxer “icudt46l.zip” en el directori “assets”.

2. A continuació s’ha creat el paquet: edu.mistic.pfm.mInvoice.db.sqlcipher, i s’han sobreescrit les classes de ORMLite per tal de que utilitzin les llibreries per al xifrat de SQLCipher, veiem:

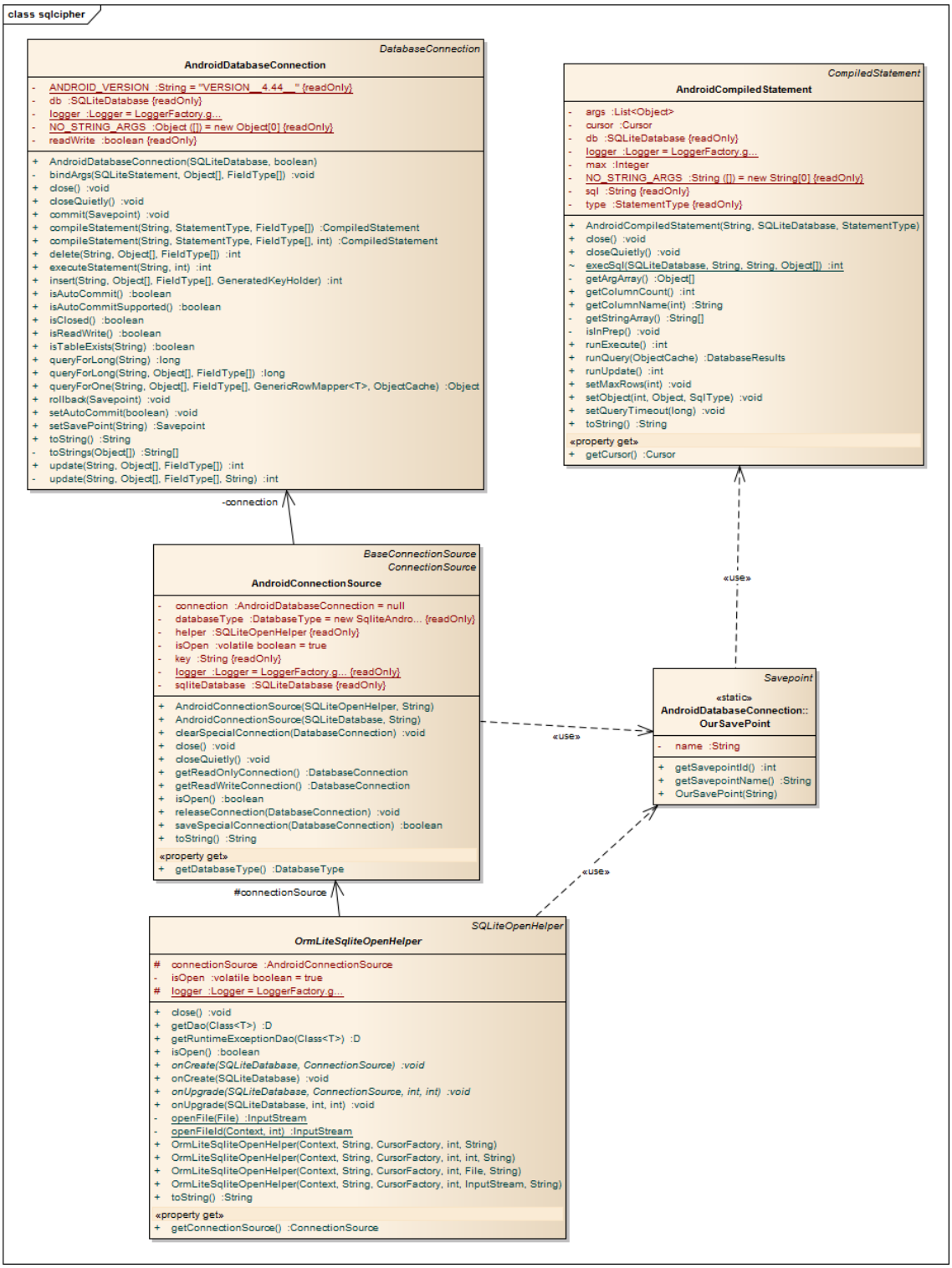


**Figura 8. Diagrama de classes de la integració entre SQLCipher i ORMLite**

Bàsicament les modificacions venen donades pel fet de que la classe `OrmLiteSqliteOpenHelper`, ha calgut fer-la heretar de la classe de SQLCipher: `net.sqlcipher.database.SQLiteOpenHelper`, això comporta modificar les classes de les que depèn aquesta i algunes de les seves relacions, per tal de que s'utilitzin les llibreries de SQLCipher.

Aquesta classe, `OrmLiteSqliteOpenHelper`, s'utilitza per accedir amb ORMLite a la BBDD, SQLite, xifrada mitjançant la tecnologia de SQLCipher.

### 5.2.3. Gestió de les claus de xifrat

Ha calgut definir unes especificacions de disseny per la gestió de claus del sistema de xifrat, a continuació s'especifiquen aquestes:

1. Quan s'accedeix a l'aplicació per primer cop, l'usuari s'ha de donar d'alta a l'aplicació (amb les seves credencials i dades personals) per poder accedir a aquesta, en aquest moment es crea una BBDD de credencials, aquesta BBDD es xifrarà utilitzant una clau pre-calculada a partir de:

    a. L'IMEI del terminal mòbil.
    b. La data de creació de la base de dades amb "paddings" per la dreta del caràcter "E" fins a assolir la llargada del IMEI.
    c. Posteriorment és fa una operació XOR amb l'IMEI i la paraula obtinguda en el pas anterior. Aquesta serà la clau utilitzada per xifrar la BBDD de credencials.

2. Posteriorment, les dades de l'usuari (nom, cognoms, adreça de correu, identificador d'usuari i paraula de pas) es desen a la base de dades de credencials.
3. Quan l'usuari a continuació s'autentica a l'aplicació amb les seves credencials, es crea la base de dades principal si aquesta no ha estat creada abans, aquesta segona BBDD, es xifra amb la paraula de pas utilitzada per l'usuari la qual està emmagatzemada en la base de dades de credencials.

A continuació de forma gràfica es presenta el procés de creació de les BBDD:

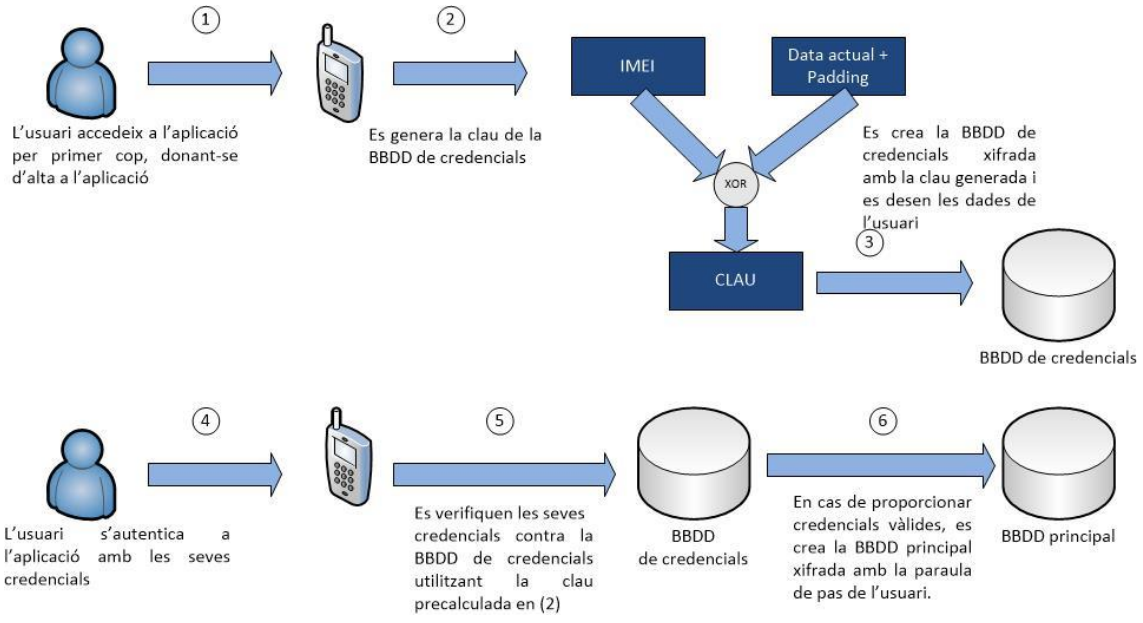


**Figura 9. Procés de creació de la BBDD**

Perquè s'ha triat aquest mecanisme de gestió de credencials? Doncs, el motiu principal és que l'usuari pugui recuperar la seva paraula de pas en cas de que aquesta sigui oblidada, i per tant, es pugui en conseqüència, recuperar la clau de xifrat de la

base de dades principal que emmagatzema la informació. Aquesta recuperació es produiria accedint a la base de dades de credencials amb la clau pre-calculada, recuperant la paraula de pas de l'usuari i enviant-la a l'adreça de correu amb la que s'ha donat d'alta a l'aplicació.
L'algorisme de generació de clau pre-calculada de la base de dades de credencials, basaria la seva seguretat per una banda en el desconeixement de les variables IMEI i data de creació de la BBDD, així com, en el desconeixement de l'algorisme utilitzat per generar la clau. No obstant, aquest desconeixement o secret, deixa de ser-ho quan ho publiquem en aquesta memòria, en qualsevol cas, per assolir aquests objectius, s'ha implementat a nivell de codi un servei (interfície) amb la seva implementació, els quals són responsables de dur a terme la generació de les claus, aquests són respectivament: `KeyService` i `KeyServiceImpl`. Per assolir un algorisme de generació més segur si es considerés adient, es podria substituir la implementació de la interfície amb un altre algorisme de generació de claus, de forma modular, sense que resultés massa traumàtic per l'aplicació.
A continuació es mostra el diagrama de classes del servei de generació de claus de l'aplicació:

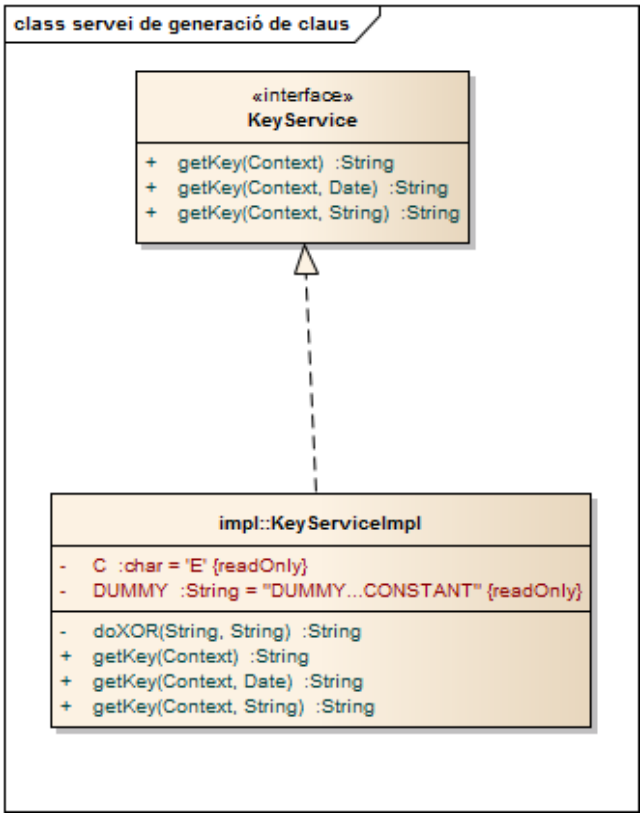


**Figura 10. Diagrama de classes del servei de generació de claus**

L'esmentat servei proveeix de varies signatures de la funció getKey(..) que basant-se en els paràmetres d'entrada, serà l'encarregada de generar la clau per xifrar la BBDD de credencials. La implementació proveeix d'una funció útil que permet aplicar una operació XOR sobre els paràmetres d'entrada, i retornar com a paràmetre de sortida el resultat de l'operació.

### 5.2.4. Creació i accés a la BBDD

Per tal de crear la BBDD i accedir a aquesta mitjançant operacions CRUD, s'han implementar dos classes, "DataBaseHelper", que seran les que proporcionaran de mètodes i funcions útils tant per manipular dades a traves del patró DAO (operacions DML) com per modificar la estructura de la BBDD mitjançant operacions DDL. Aquestes classes pertanyen al paquet: `edu.mistic.pfm.mInvoice.db`, i són: `DBCredentialsHelper` i `DBMainHelper`. La primera d'ella permet la manipulació de la BBDD de credencials i la segona la manipulació de la BBDD principal, aquestes es mostren en el següent diagrama:

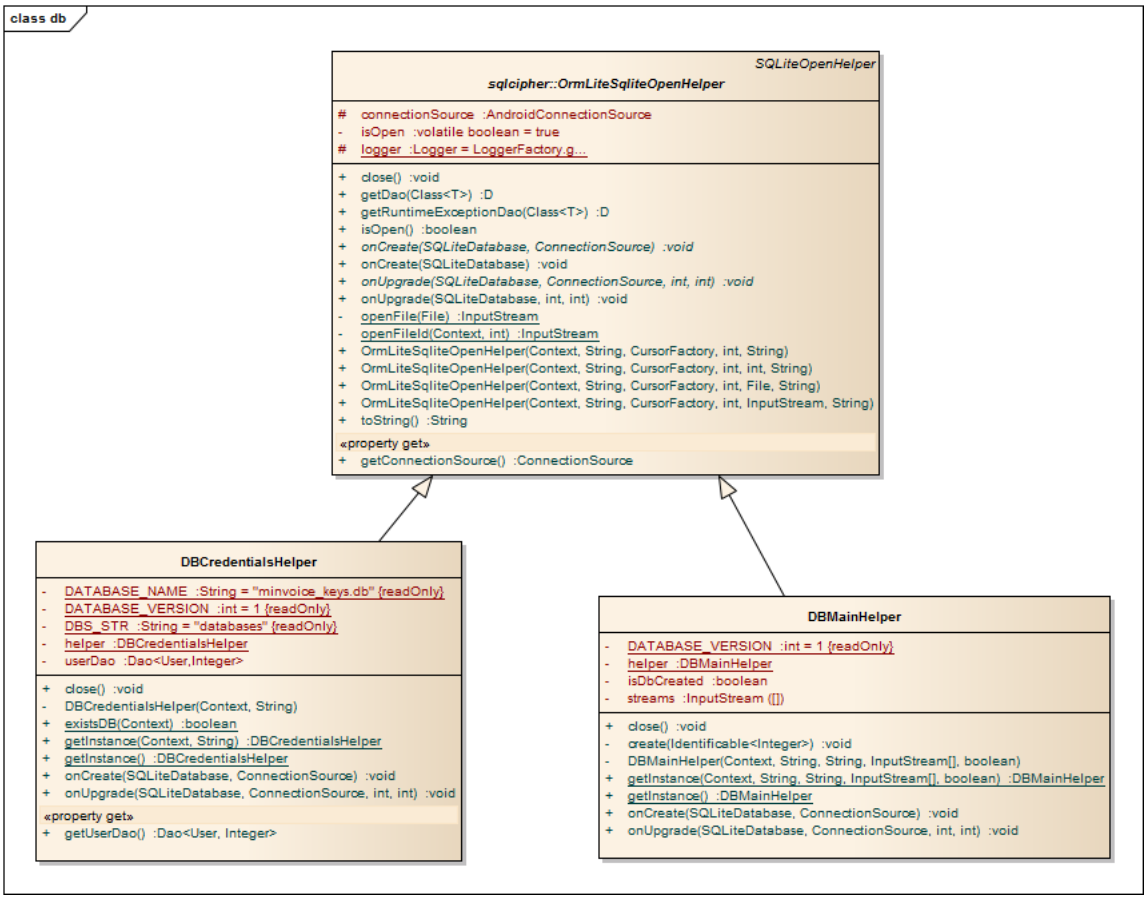


**Figura 11. Diagrama de classes dels DataBaseHelpers**

A continuació es mostra una breu explicació de les funcions i mètodes més rellevants:

- getDao(..): funció que a través dels seu argument, permet recuperar un DAO d'accés a les dades d'un tipus de classe del model.
- getInstance(..): funció estàtica que permet recuperar la instancia de la BBDD, la qual es de tipus "singleton".
- openFile(..): funció utilitzada per obrir la BBDD.
- close(): mètode cridat en la finalització per tancar la BBDD.
- onCreate(..): mètode cridat per crear la BBDD.
- onUpgrade(..): mètode cridat per modificar l'estructura de la BBDD.

Val a dir que la creació de la BBDD, es realitza en la primera activitat d'Android que es carregada, això és, en la `LoginActivity`. Posteriorment l'accés a la BBDD a través dels DAO, es realitzarà recuperant la instància del "DataBaseHelper" corresponent en el constructor de la implementació dels serveis, per a continuació recuperar el DAO a partir d'aquest.

Per altra banda, per tal de crear la base de dades i després accedir-hi, les classes del model s'han d'anotat amb anotacions de ORMLite, vegis un exemple a continuació (les anotacions comencen amb el símbol '@'):

```
@DatabaseTable
public class Category extends Identificable<Integer>{
	/**
	 * Serial version.
	 */
	private static final long serialVersionUID = 6268047780966292610L;
	/**
	 * Constant que defineix el camp de BBDD: DESCRIPTION.
	 */
	private static final String DESCRIPTION="DESCRIPTION";
	/**
	 * L'identificador de l'entitat.
	 */
	@DatabaseField(generatedId = true, columnName = ID)
	private Integer id;
	/**
	 * La descripció de la categoria.
	 */
	@DatabaseField(columnName = DESCRIPTION)
	private String description;
.....
.....
```

En aquest exemple, estem definint una classe del model de persistència "Category" que es correspon amb una taula de BBDD, on aquesta té un camp ID i un camp DESCRIPTION, el primer de tipus enter i el segon es de tipus cadena de caràcters.

A partir d'aquí la BBDD serà creada mitjançant el mètode onCreate(..) del "DataBaseHelper".

Per altra banda, com s'ha comentat les operacions CRUD que es realitzen sobre les entitats del model de persistència, es realitzen a través dels DAO de forma transparent mitjançant objectes Java, sense tenir que utilitzar el llenguatge SQL, per fer-nos idea de com funciona veiem un exemple:

```
	/**
	 * Permet donar d'alta una categoria.
	 * @param category la categoria a donar d'alta.
	 */
	public void save(CategoryTO category) {
		final Category model = new Category();
		PropertyUtils.copyProperties(model, category);
		try {
			dao.create(model);
		} catch (SQLException e) {
			throw new RuntimeException("Cannot save category",e);
		}
	}
```

En aquest exemple, estem creant una categoria a partir de les dades del Transfer

Object rebut com a paràmetre, vegis marcat en groc l'accés al DAO per tal de realitzar la creació.

### 5.2.5. Transaccionalitat

Per garantir la integritat de les dades en operacions de creació, actualització o esborrat d'objectes, i més concretament en la part de l'aplicació en que es desa una factura després d'haver-la validat, es fa necessari l'ús de transaccions, en aquest cas les transaccions són gestionades per ORMLite a través del `TransactionManager`, en el cas de produir-se una errada es fa "rollback" de tota la transacció, en cas contrari es fa "commit" d'aquesta. Veiem un exemple:

```
public InvoiceTO save(final Facturae facturae, final CategoryTO categoryTo,
                final byte[] data) {
        InvoiceTO invoice = null;
        try {
                invoice = TransactionManager.callInTransaction(connectionSource,
                                new InvoiceCallable(facturae, categoryTo, data));
        } catch (SQLException e) {
                throw new RuntimeException(e);
        }
        return invoice;
}
```

En aquest cas s'està desant una factura a la BBDD, la transacció és gestiona per mitjà del `TransactionManager`, com ja s'ha comentat. Es passen com a paràmetres la connexió, i un objecte `InvoiceCallable`, la crida a la funció *`callInTransaction`*, invocarà la funció `call(..)` de l'objecte `InvoiceCallable`, i aplicarà la transaccionalitat a tot lo que s'executi dintre de l'esmentada funció.

## 6. Mòdul de descobriment de factures

En aquest mòduls es pretén descriure com s'ha dissenyat i implementat el mòdul de descobriment de factures o tiquets. Aquesta secció al igual que les anteriors es divideix en dos parts, una corresponent a la descripció funcional i un altre on es parla del disseny i com s'ha dut a terme la implementació.
Cal dir també, que inicialment es va plantejar el fet de que les comunicacions entre l'emissor de factures i el dispositiu mòbil es realitzessin a traves de NFC, tant però, finalment això no ha estat possible, i s'ha implementat comunicació via Bluetooth.

### 6.1. Descripció funcional

En el context de la facturació electrònica, es fa necessària la existència d'un canal de comunicacions entre l'emissor i el receptor, en aquest cas el dispositiu mòbil, per tal de que, es pugui realitzar la transmissió de la factura o tiquet en qüestió.
Aquest canal ha de garantir la seguretat de les transmissions entre ambdós punts de la comunicació, garantint la integritat i confidencialitat de les dades trameses.

## 6.2. Disseny i implementació

Per satisfer els requisits funcionals s'ha dissenyat i implementat un sistema de comunicacions, entre un emissor de factures i el terminal o dispositiu mòbil. Aquest mòdul que permet la comunicació entre ambdós punts ha estat anomenat, mòdul de descobriment de factures.
Inicialment, com s'ha comentat, estava previst implementar el mòdul amb tecnologies NFC, tant però, per limitacions s'ha tingut que implementar amb Bluetooth, no obstant, es preveu en curt termini i com a treball futur implementar un mòdul de comunicacions NFC, aquest tipus de comunicacions són idònies per sistemes de pagament donat el molt curt abast de les comunicacions, només un pocs centímetres entre emissor i receptor, amb la dificultat que comporta la intercepció de les comunicacions.
En quant a Bluetooth a nivell de transmissió, la seguretat ve implícita en el mode de transmissió, doncs utilitza la tècnica de salts de freqüència, i a més a més, garanteix que les connexions només es produeixen si dos dispositius estan emparellats i autoritzats per la transmissió. En quant els salts de freqüència, consisteix en dividir la banda base de connexió en distints canals de connexió, amb longituds de 1 MHz cadascun, i intercalar els paquets tramesos amb seqüències de 1600 salts per segon, tot plegat aquest fet, fa molt difícil la intercepció dels missatges.

De cara a implementar aquest mòdul de descobriment de factures s'ha implementat un mòdul de servidor (bluetooth-server) i el mòdul de client, en els següents apartats veurem la implementació d'ambdós mòduls amb més detall.

### 6.2.1. Mòdul de servidor (bluetooth)

El mòdul de servidor consisteix en un programari el qual a través d'un dispositiu bluetooth (hardware), llegeix un fitxer, la ruta del qual és passada com a paràmetre a traves de la línia d'arguments, posteriorment espera a que un dispositiu es connecti per enviar-li el fitxer llegit anteriorment.
Val a dir, que en tota comunicació Bluetooth en primer lloc abans d'establir la connexió es produeix un emparellament dels dispositius servidor i client, sol·licitant-se un PIN, per posteriorment poder establir la connexió.
Per implementar el servidor, s'han utilitzat les llibreries de BlueCove, i s'ha provat el funcionament en Linux Fedora 18 x64, així com, en Windows 7. En el cas de Linux cal tenir instal·lades prèviament les llibreries de BlueZ (paquet bluez-utils; actualment la majoria de sistemes Linux porten aquestes instal·lades de forma predeterminada). En el cas de Windows, cal assegurar-se que el mòdul de servidor s'executa amb la versió de 32 bits de la màquina virtual de Java.
En quant a la implementació, s'ha implementat per una banda la classe `BluetoothServer` que hereta de `Thread`, la qual mitjançant el mètode `run(..)` inicia el servidor esperant a que un dispositiu es connecti, per trametre el fitxer quan s'estableix la comunicació. La comunicació establerta es realitza mitjançant el protocol `btspp` (Bluetooth Serial Port Profile), el servei publicat s'identifica per un UUID, necessari per identificar el servei de forma unívoca.
El punt d'entrada a l'aplicació l'implementa la classe `Main`, que rep com a arguments la ruta en el sistema de fitxers del fitxer a trametre. Veiem el diagrama de classes:

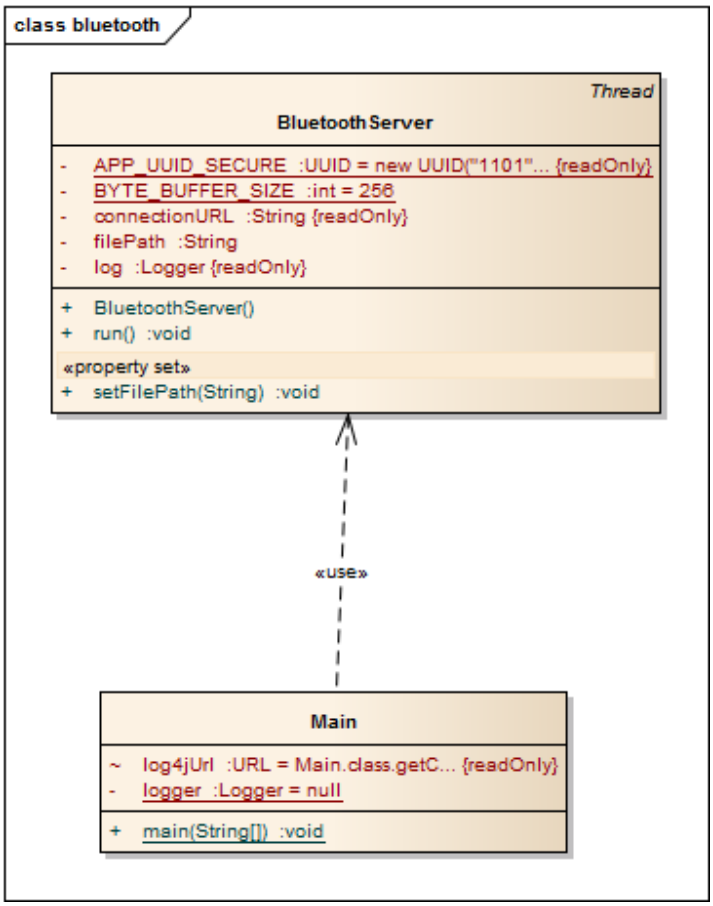


**Figura 12. Diagrama de classes del servidor bluetooth**

Per executar l'aplicació en un sistema Linux suposant que el fitxer a trametre és /home/user/invoices/invoice1.xml tindríem que executar la següent comanda:

```
java -cp    ./bluetooth-server-0.0.1-SNAPSHOT.jar:
            ./libraries/*.jar
             edu.mistic.minvoice.bluetooth.Main
            /home/user/invoices/invoice1.xml
```

On `./libraries` seria el directori on es troben totes les dependències/ llibreries necessàries per executar l'aplicació (p.e: bluetooth o log4j). També, es pot executar el mòdul des del mateix Eclipse.

Per finalitzar en lo que respecta a aquesta secció, comentar que al igual que en la resta de mòduls s'ha utilitzat Maven, per tal d'afegir les llibreries pertinents o dependències:

```
<dependencies>
      <!-- BlueCove is JSR-82 J2SE implementation that currently
       interfaces with the Mac OS X, WIDCOMM, BlueSoleil and Microsoft
      Bluetooth stack -->
      <dependency>
            <groupId>net.sf.bluecove</groupId>
            <artifactId>bluecove</artifactId>
```

```
            <version>2.1.0</version>
        </dependency>
        <dependency>
            <groupId>net.sf.bluecove</groupId>
            <artifactId>bluecove-gpl</artifactId>
            <version>2.1.0</version>
        </dependency>
        <!-- Log4j logging -->
        <dependency>
            <groupId>log4j</groupId>
            <artifactId>log4j</artifactId>
            <version>1.2.17</version>
        </dependency>
        <!-- Commons IO -->
        <dependency>
            <groupId>commons-io</groupId>
            <artifactId>commons-io</artifactId>
            <version>2.4</version>
        </dependency>
</dependencies>
```

### 6.2.2. Client (bluetooth)

La implementació del client es una mica més complexa, per dur-la a terme s'han utilitzat les API d'Android, les quals porten incorporades les llibreries pertinents de cara a establir la connexió via bluetooth: `android.bluetooth.*`.

Els passos a seguir per tal de descobrir una factura són:

1. Disposar del dispositiu de servidor en estat iniciat i preparat per emetre la factura.
2. Quan l'usuari pren l'opció de menú "descobrir factura", i l'usuari selecciona comunicació per "bluetooth", en cas d'estar des-habilitada la connectivitat bluetooth del dispositiu mòbil, s'inicia l'adaptador bluetooth del dispositiu.
3. El dispositiu mòbil cerca els dispositius bluetooth en l'espectre electromagnètic, i es mostra una llista amb aquests a l'usuari.
4. L'usuari selecciona el dispositiu de servidor a conectar, es sol·licita la introducció de PIN.
5. S'introdueix el PIN a dues bandes, produint-se l'emparellament.
6. Per finalitzar es produeix la connexió entre emissor (el servidor) i el client, i s'envia la factura al receptor, el dispositiu mòbil, el qual la mostra per pantalla.

Amb més detall es mostra el diagrama de seqüencia que descriu els passos seguits per rebre la factura, així

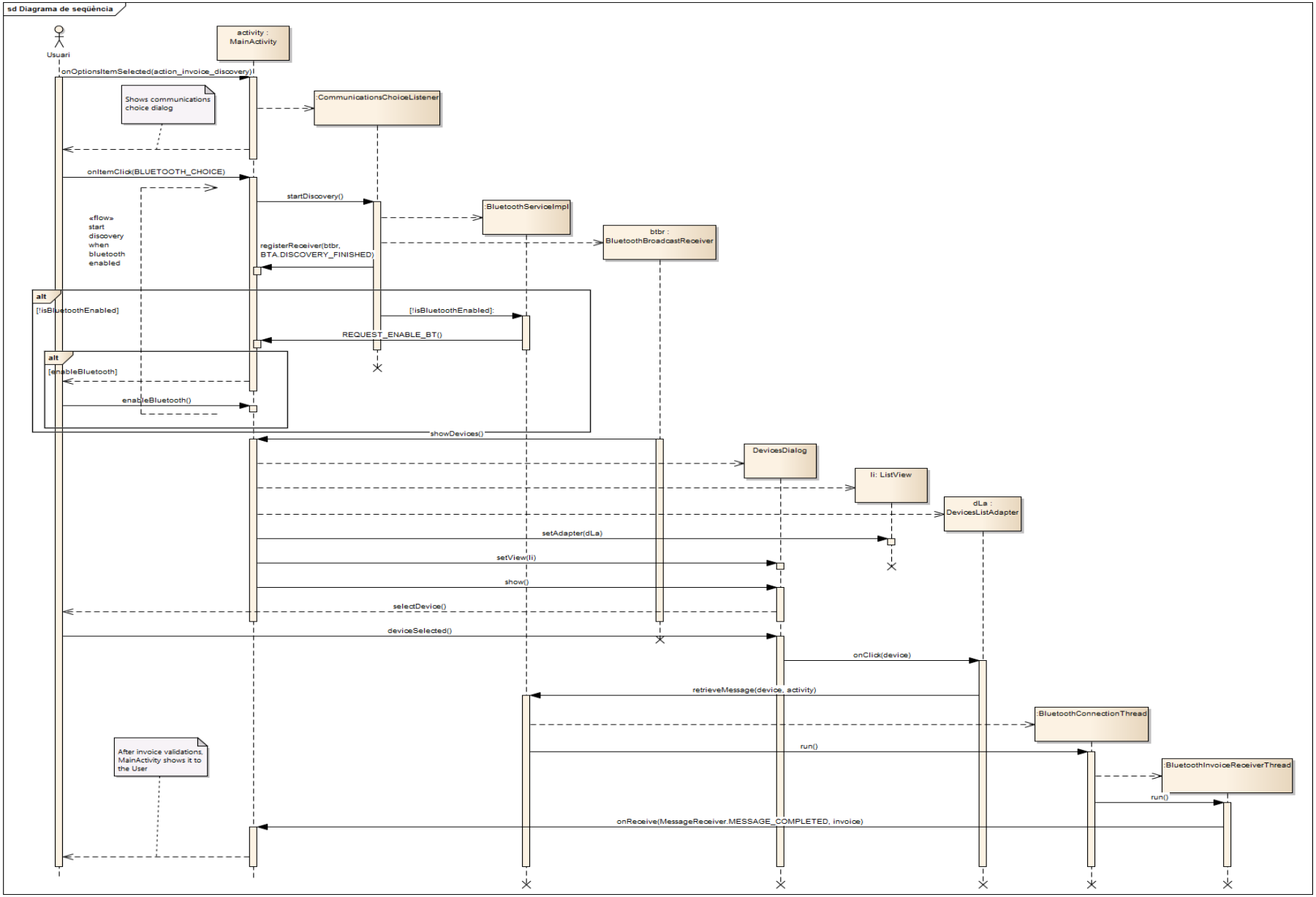


**Figura 13. Diagrama de seqüència del procés de descobriment d'una factura**

A continuació es proporciona una breu descripció de les classes que intervenen en aquest diagrama, amb una breu descripció del paper que juguen en aquest procés:

| Nom de la classe | Descripció |
|---|---|
| *MainActivity* | És l'activitat principal de l'aplicació, la que construeix i mostra els objectes de la UI amb la que interacciona l'usuari. Al mateix temps implementa la interfície *MessageReceiver* lo qual li permetrà rebre com a missatge, la factura rebuda per el subsistema de descobriment de factures i presentant-la finalment a l'usuari. |
| *CommunicationChoiceListener* | Es tracta d'un *listener* que rebrà la selecció del tipus de comunicació triada per l'usuari, la qual potser NFC o Bluetooth. |
| *BluetoothServiceImpl* | El servei de Bluetooth que implementa la interfície *BluetoothService*. Interacciona amb les classes *BluetoothConnectionThread* i *BluetoothReceiverThread*. Es a través del mètode *retrieveMessage(..)*, que desencadena el procés de recepció de la factura. |
| *BluetoothBroadcastReceiver* | Es tracta d'un *Listener* que escolta els events de descobriments de dispositius bluetooth. Lo qual permetrà confeccionar la llista de dispositius bluetooth disponibles. El procés de descobriment de dispositius finalitza amb una notificació a l'activitat principal, per tal de que siguin mostrats a l'usuari. |
| *AlertDialog.Builder (DevicesDialog)* | És el diàleg visual que es mostra a l'usuari per tal de que seleccioni un dispositiu amb el que connectar. |
| *ListView* | Component gràfic que en aquest cas s'utilitza per mostrar la llista de dispositius bluetooth descoberts. |
| *DevicesListAdapter* | Adaptador que per permetrà mostrar cadascun dels dispositius en la llista. A més a més, reaccionarà als events de clic que es facin sobre els elements. Desencadenant la connexió contra el dispositiu en el que s'ha fet clic. |
| *BluetoothConnectionThread* | Classe que hereta de *java.lang.Thread*, fil que s'executa per establir mitjançant el seu mètode *run(),* la connexió contra el dispositiu (servidor). |
| *BluetoothInvoiceReceiverThread* | És cridat per la classe esmentada tot just en el punt anterior. Es tracta d'un fil, que hereta de java.lang.Thread, i que es l'encarregat de rebre la factura per el dispositiu (servidor) amb el que prèviament ha establert connexió. |

**Taula 6. Descripció de les classes que intervenen en el descobriment de la factura**

Per últim en lo que respecta aquesta secció, cal comentar que s'han donat els permisos adients a nivell de programari per tal de que l'aplicació pugui accedir a les funcionalitats bluetooth del dispositiu mòbil:

```
<uses-permission android:name="android.permission.BLUETOOTH"/>
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
```

# 7. L'aplicació

En aquesta secció es mostraran les funcionalitats implementades en l'aplicació, es divideix en tres parts: la primera relativa a l'autenticació i registre a l'aplicació, la segona relativa a la gestió de categories que ens permetran classificar les factures, i la tercera i última relativa a la gestió de factures, que va des del procés de recepció de la factura, fins la cerca o consulta de factures.

## 7.1. Autenticació i registre

Quan l'usuari accedeix per primer cop a l'aplicació, se li mostra la pantalla d'autenticació on l'usuari pot introduir les seves credencials o fer clic a l'opció de registrar-se a l'aplicació. A continuació es mostra el mapa de navegació amb els diferents casos d'us possibles:

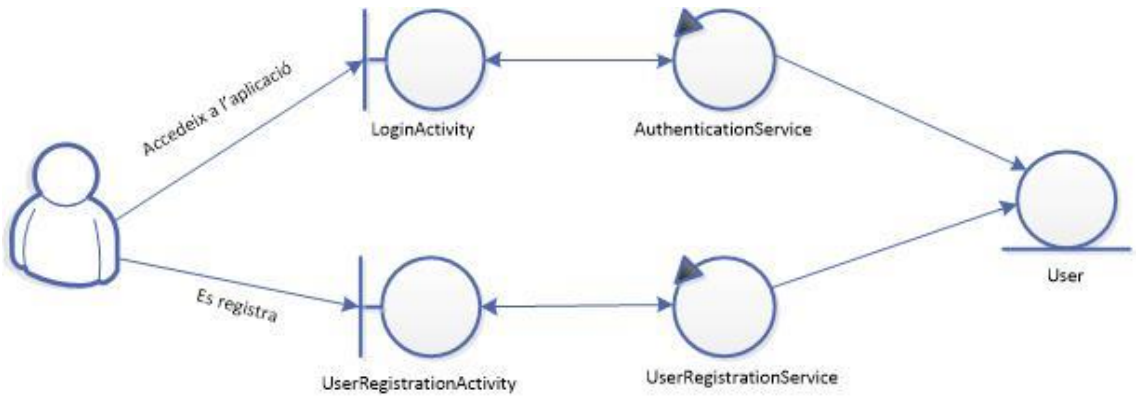


**Figura 14. Mapa de navegació: Registre i autenticació de l'usuari**

### 7.1.1. Autenticació (CU01)

La pantalla d'autenticació la qual es visualitzada a través del component `LoginActivity`, presenta el següent aspecte:

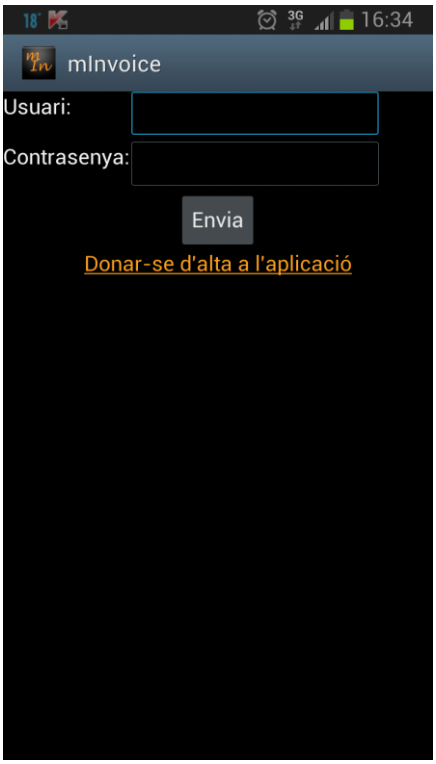


**Figura 15. Pantalla d'autenticació**

Mitjançant aquesta pantalla l'usuari pot introduir les seves credencials (usuari i contrasenya), a continuació el servei `AuthenticationService`, valida les credencials introduïdes i en cas de que siguin correctes, li permet l'entrada a l'aplicació, visualitzant-se la pantalla de categories (figura 18) a través de la qual es pot accedir a les factures d'una determinada categoria.
Per altra banda, la pantalla presenta un enllaç a la pantalla de registre de l'usuari, per tal, de que l'usuari es pugui registrar en aquesta, en el cas de no haver accedit abans a l'aplicació.

### 7.1.2. Registre de l'usuari (CU02)

La pantalla de registre permet que l'usuari en el cas de que no estigui donat d'alta pugui donar-se d'alta a l'aplicació, per tal de poder accedir-hi posteriorment. Aquesta pantalla es visualitza per mitjà de l'activitat `UserRegistrationActivity`, la qual utilitza el servei `UserRegistrationService`, per tal de donar d'alta l'usuari. A continuació es presenta l'esmentada pantalla:

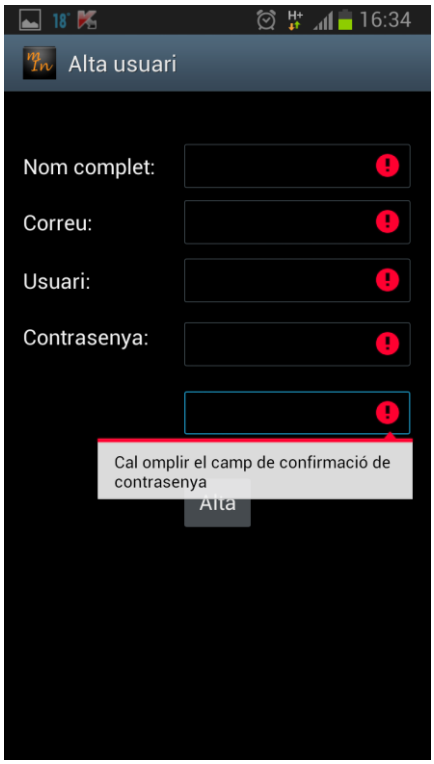


**Figura 16. Pantalla de registre d'usuari**

En aquesta pantalla es sol·liciten les dades de l'usuari, per tal de donar-lo d'alta, aquestes són: el nom complert, l'adreça de correu, l'identificador d'usuari, la contrasenya i el camp de confirmació de contrasenya. Tots els camps esmentats són obligatoris i es realitzen les oportunes validacions, garantint a més a més, que la contrasenya triada tingui com a mínim 8 caràcters, i els camps de confirmació de contrasenya i de contrasenya coincideixin.
Per altra banda, es recaven el nom i adreça de correu, de cara a enviar-li un correu de recordatori de contrasenya, en el cas de que aquesta hagi estat oblidada.

## 7.2. Gestió de categories

En aquesta secció es descriuen els diferents casos d'ús relatius a la gestió de categories, des de la consulta com el manteniment de categories. El mapa de navegació amb els casos d'ús relatius a la gestió és el següent:

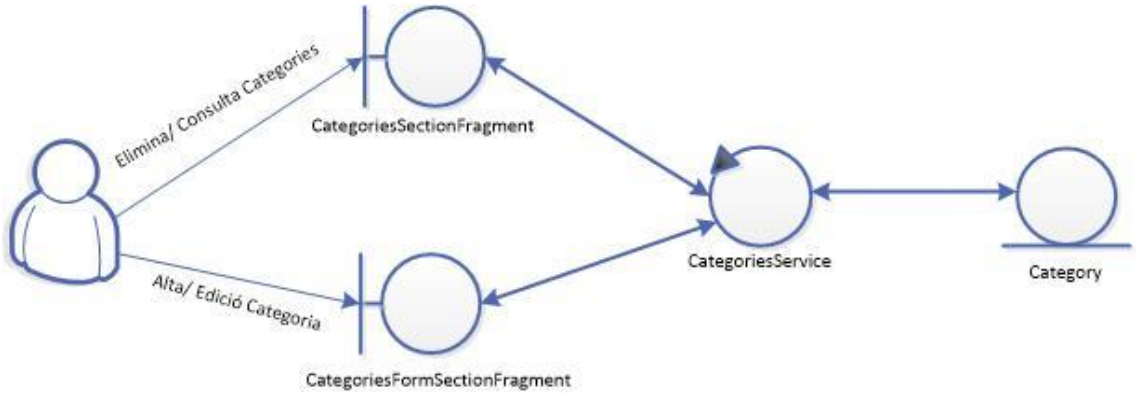


**Figura 17. Mapa de navegació: Gestió de categories**

En la figura anterior podem veure els components de software que intervenen, els fragments esmentats formen part de la `MainActivity`, i a l'hora utilitzen el servei `CategoriesService` per tal de realitzar les operacions CRUD, l'objecte del model manipulat pertany a la classe `Category`.

### 7.2.1. Consulta categories (CU03)

La consulta de categories és la primera pantalla que es visualitza quan s'accedeix a l'aplicació, en ella es mostren les diferents categories donades d'alta a l'aplicació, i fent clic en una d'elles es poden visualitzar les factures associades a la categoria seleccionada. A continuació es presenta l'aspecte de l'esmentada pantalla:


**Figura 18. Pantalla amb llistat de categories**

Un incís, es pot veure en la captura anterior que la navegació per l'aplicació, es pot

realitzar a través de pestanyes (Categories, Factures, Consums...).
Per altra banda, si es fa prem una categoria durant uns segons, apareixen les opcions d'eliminació o edició de la categoria, a través de les quals es podrien dur a terme les respectives accions (veure casos d'ús següents).


**Figura 19. Selecció "de durada llarga" sobre categoria**

### 7.2.2. Eliminació d'una categoria (CU04)

Per tal d'eliminar una categoria, cal seleccionar o prémer durant uns segons a sobre de la categoria en qüestió (veure la figura 19), i a continuació seleccionar l'opció "Elimina la categoria", a continuació a apareixerà un diàleg de confirmació, i si confirmem, la categoria serà esborrada.

### 7.2.3. Alta i edició d'una categoria (CU05)

Per implementar els casos d'ús relatius a l'alta i edició de categories, s'ha utilitzat la mateixa pantalla. En el cas de l'alta no es mostren dades, i en el cas de l'edició es mostren les dades de la categoria en qüestió.
Per altra banda, a l'edició s'accedeix premem durant uns segons sobre la categoria del llistat que volem editar (veure la figura 19), a continuació accedirem a la pantalla d'alta o edició, la qual té un aspecte com el que es mostra a continuació:

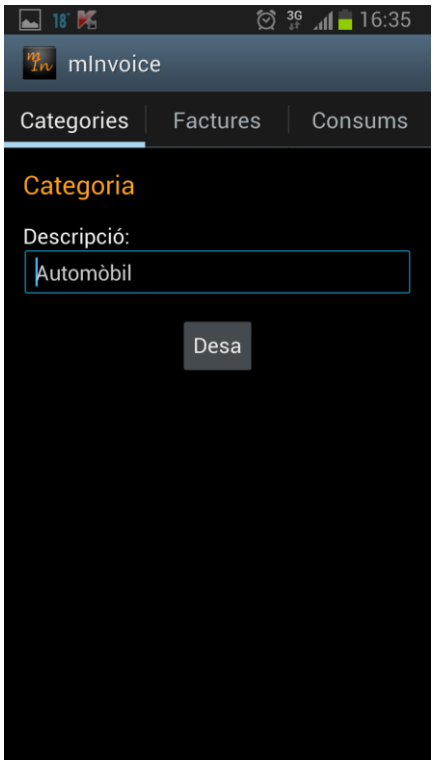


**Figura 20. Pantalla d'alta o edició de la categoria**

En aquesta pantalla es presenta una capsa de text per introduir la descripció de la categoria en qüestió.
Cal comentar també, que s'accedeix a l'alta de categories, des de la pestanya de categories, fent clic al botó d'opcions de menú del dispositiu mòbil, vegis la següent pantalla:


**Figura 21. Accés a l'opció de creació de la categoria**

## 7.3. Gestió de factures

En aquest apartat, es descriuen de forma breu els diferents casos d'ús relatius a la gestió de factures. Des del descobriment d'aquestes, fins a la visualització de la factura o la cerca. A continuació es presenta el mapa de navegació::

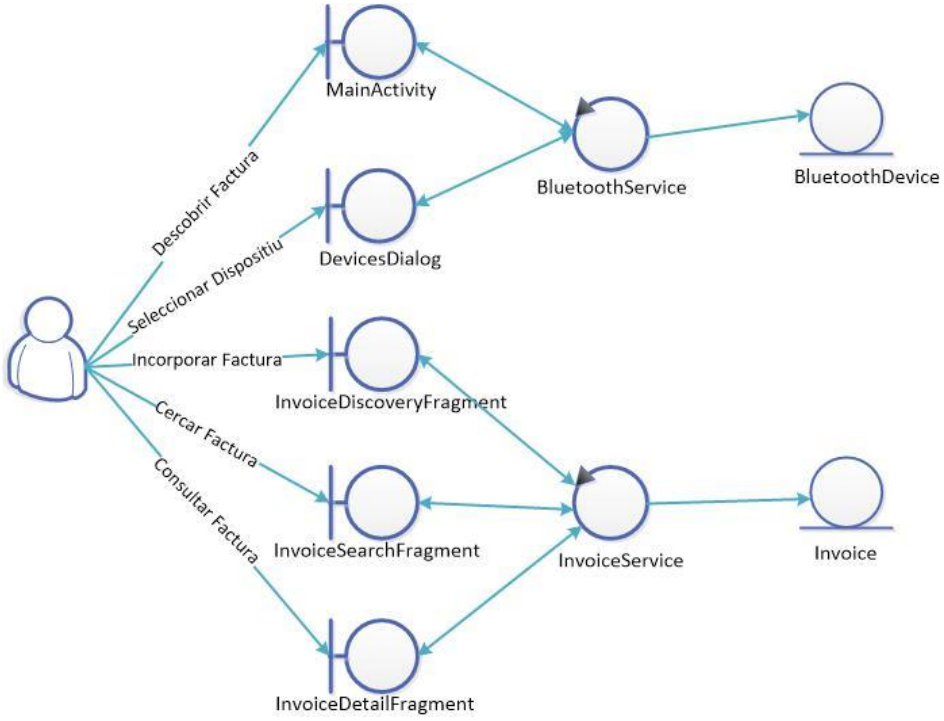


**Figura 22. Mapa de navegació: Gestió de factures**

En aquest diagrama es descriuen els diferents escenaris: el procés de descobriment de la factura, que passa per diferents fases (descobriment, selecció del dispositiu bluetooth; emissor, i la incorporació al sistema), la cerca de factures, i la consulta de les dades de la factura. Per dur a terme la implementació dels diferents components de la UI s'han utilitzat com a components, fragments, els quals formen part a la vegada de l'activitat principal. Per altra banda, els serveis que proveeixen o implementen la funcionalitat en qüestió, són el `BluetoothService` i l'`InvoiceService`, els quals a la vegada utilitzen les entitats `BluetoothDevice` (el dispositiu emissor contra el que ens connectarem) i `Invoice` (objecte del model).

### 7.3.1. Descobriment de la factura (CU06)

Aquest cas d'us es descriu la funcionalitat necessària per incorporar les factures al dispositiu mòbil, que implica:

1. Seleccionar l'opció de descobriment de factura.
2. Seleccionar el tipus de comunicació a realitzar (actualment només està suportat bluetooth).
3. L'activació de bluetooth al terminal mòbil en el cas de que no estigui habilitat.

4. La cerca i mostra per pantalla dels dispositius bluetooth disponibles, així com, l'opció de connectar-se a un dispositiu en concret.
5. L'establiment de la connexió, i recepció de la factura al mòbil, mostrant-se per pantalla les dades de la factura en qüestió, i l'opció d'incorporar la factura al terminal mòbil.

En primer lloc com s'ha dit cal seleccionar l'opció de descobriment de la factura:


**Figura 23. Selecció de l'opció de descobriment de la factura**

Aquesta opció de selecció hi apareix disponible a través del menú d'opcions del terminal mòbil, a qualsevol de les pestanyes (categories, factures, consums...).

A continuació apareix un diàleg en que se'ns interroga per el tipus de comunicació que volem establir per rebre la factura:

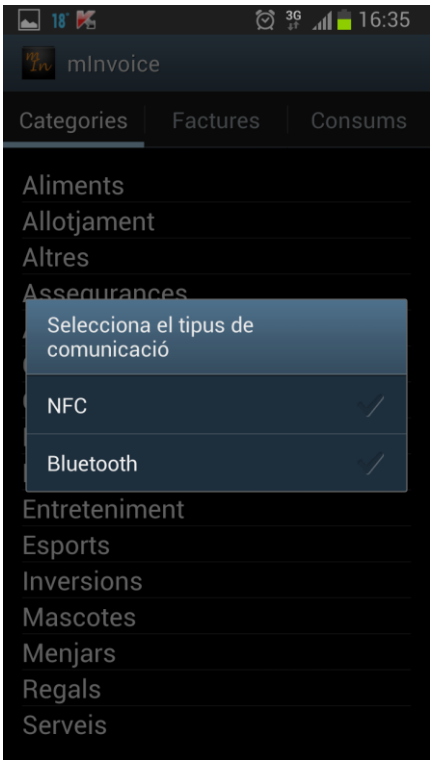


**Figura 24. Diàleg de selecció del tipus de comunicació.**

Apareix la possibilitat de connectar-se per NFC o per bluetooth, no obstant, com s'ha comentat l'única opció implementada és la darrera.
En seleccionar bluetooth, en el cas de que no estigui activat ens apareixerà un diàleg on es demanarà permís per habilitar bluetooth:

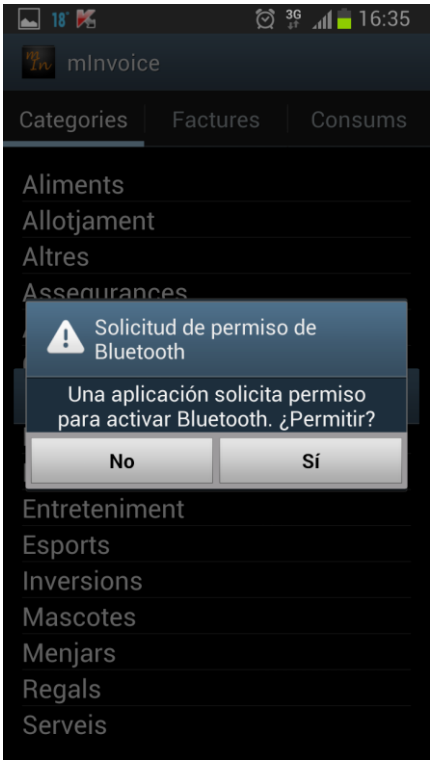

**Figura 25. Diàleg d'activació de bluetooth**

Acte seguit apareixerà la llista de dispositius disponibles per connectar, seleccionarem com és lògic el servidor de bluetooth emissor de factures:

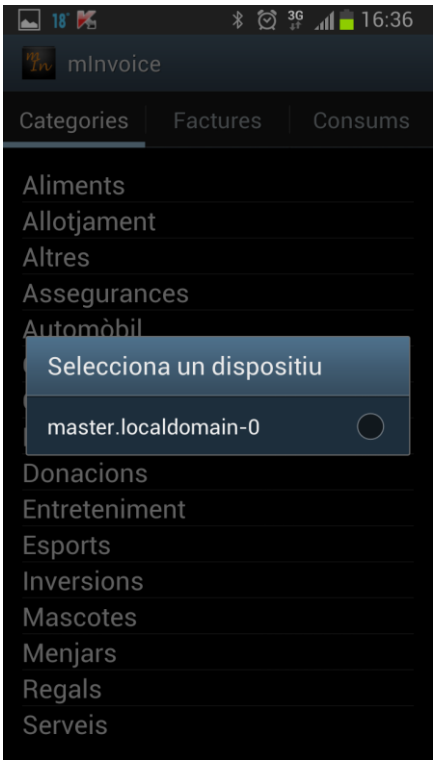


**Figura 26. Llistat de dispositiu bluetooth disponibles**

Al seleccionar el dispositiu en qüestió, en primer lloc es produirà l'emparellament, i se'ns sol·licitarà el codi PIN que s'utilitzarà per connectar contra el servidor, s'introduirà el mateix codi PIN al servidor, per a continuació, produir-se la connexió i rebre la factura al dispositiu mòbil, on es mostrarà en el dispositiu mòbil, les dades del signant de la factura i les dades de la factura en qüestió. En la següent captura veurem la pantalla que es mostra tres rebre la factura i que demana confirmació a l'usuari per incorporar-la al dispositiu mòbil:

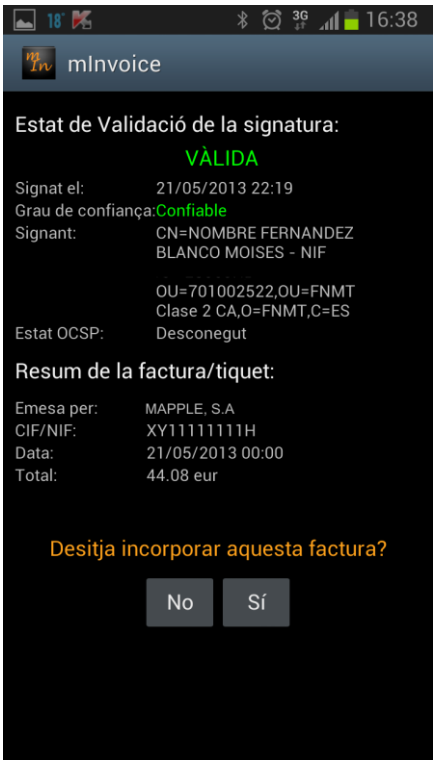


**Figura 27. Pantalla de pre-visualització i incorporació de la factura rebuda**

En la figura o pantalla anterior podem veure una factura rebuda a través del mòdul de descobriment de factures, la qual ha estat signada amb un certificat de la FNMT vàlid.

A continuació es descriuen els diferents camps que es mostren a la pantalla:

| Camp | Descripció |
|---|---|
| Estat de validació de la signatura | Mostrat el resultat de la validació de la signatura XADES del document Factura-e. Els valors que pot prendre són: VÀLIDA, INVÀLIDA o SENSE SIGNATURA. A més, aquests valors es mostren amb un codi de colors: VERD => VÀLIDA, VERMELL => INVÀLIDA i GRIS CLAR => SENSE SIGNATURA |
| Signat el | Es mostra la data en que es va realitzar la signatura. |
| Grau de confiança | Mostra el grau de confiança en el certificat. Els valors que pot prendre aquest camp són: Sense Revisar, Sense confiança, Confiable i CA Confiable. També es mostren amb un codi de colors: si es confiable color verd, si no hi ha confiança color vermell, i gris clar en la resta de casos. |
| Signant | Es mostra la informació relativa al certificat utilitzat per dur a terme la signatura. |
| Estat OCSP | Mostra l'estat de la validació OCSP, en el cas de que s'hagi dut a terme, en cas contrari mostra N/A. Els valors que pot prendre el camp: desconegut, confiable i revocat. S'utilitza també un codi de colors per mostrar el camp: verd si és confiable, vermell si està revocat i |

| | |
|---|---|
| | gris clar si l'estat és desconegut. |
| Emesa per | Es mostra la raó social de l'emissor de la factura, ja és tracti d'una persona física o jurídica. |
| Data | Es mostra la data d'emissió de la factura. |
| Total | Es mostra l'import total de la factura, aplicades taxes i descomptes. |

**Taula 7. Descripció dels camps de la pantalla de previsualització i incorporació de la factura**

Per últim comentar, que la decisió final de incorporar la factura al sistema és de l'usuari, i per aquest motiu se li pregunta si desitja incorporar o no, la factura.

Si l'usuari, decideix incorporar-la l'últim pas és vincular la categoria a una de les categories existents, per tal després de poder visualitzar les factures que pertanyen a una categoria determinada, la pantalla que es mostra és la següent:


**Figura 28. Pantalla de selecció de categoria durant la incorporació**

### 7.3.2. Cerca i llistat de factures (CU07)

La cerca i consulta de factures es pot fer mitjançant el cercador de factures al qual es pot accedir a través de la pestanya factures, o bé, fent clic a una categoria (veure figura 18), en el primer dels casos apareixerà un llistat amb les factures que coincideixen amb els criteris de cerca i en el segon dels casos un llistat de les factures associades a la categoria. Tant però ambdós llistats són la mateixa pantalla que rep dades de cerca diferents, en un cas per identificador de categoria i en un altre cas per criteris de cerca, tal i com, s'ha comentat.
Veiem les diferents pantalles que intervenen en aquest cas d'ús:


**Figura 29. Pantalla de cerca de factures**

La pantalla de cerca de factures, permet la cerca per:

- Un literal inclòs en les dades de l'emissor de la factura, dit d'un altre forma, es cerca que el literal aparegui en la raó social (nom de la companyia o nom de la persona en el cas de tractar-se d'una persona física), o bé, en el nom de la marca.
- Es pot, simultàniament, cercar per la categoria a la que està vinculada la factura.
- També es poden cercar factures que estiguin compreses entre la data "Des de" i la data "Fins a".
- Per últim, es pot especificar si es tracta d'una factura emesa per una factura física o jurídica, per defecte està seleccionada aquesta darrera.

Totes les cerques impliquen operacions AND entre els diferents camps de cerca, per exemple, podríem cercar factures que pertanyen a la categoria "Serveis" continguin el literal "Restaurant" i estiguin compreses entre una data X i una data Y.

Per altra banda, la pantalla presenta dos botons, un per executar la cerca ("Cerca") i un altre per netejar les dades del filtre de cerca ("Neteja").

Per últim en lo que respecta a aquesta pantalla, cal comentar, que les dates s'introdueixen fent clic al botó "Calendari", a continuació apareix un component gràfic (spinner) que permet triar la data de cerca:

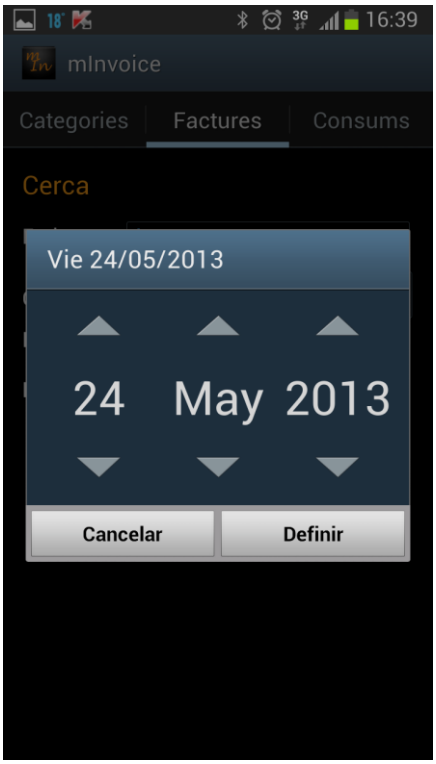


**Figura 30. Spinner per introduir les dates**

Quan premem el botó de cerca, el resultat es mostra en un llistat on apareixen les diferents factures que coincideixen amb els criteris de cerca:


**Figura 31. Llistat de factures**

En el llistat de factures i per cadascuna de les factures trobades, es mostra l'emissor

de la factura, la data d'emissió de la factura i l'import total en la moneda que sigui pertinent (es mostra l'acrònim de la moneda).

Seleccionant una factura en qüestió, es pot consultar el detall de la factura seleccionada (vegis el següent cas d'ús).

### 7.3.3. Consulta del detall d'una factura (CU08)

Com s'ha comentat al detall d'una factura s'accedeix a partir del llistat de factures, en la pantalla de detall es mostren les dades de l'emissor de la factura, les dades del receptor, els imports ... veiem-ho amb més detall:


**Figura 32. Pantalla de detall de la factura**

| Camp | Descripció |
|---|---|
| Data | Conté la data d'emissió de la factura. |
| Número | Conté el número de la factura |
| Sèrie | Número de sèrie de la factura si en té. |
| Secció empresa | Es mostra la raó social de l'empresa conjuntament amb el seu CIF/NIF (dependent de si es tracta de persona jurídica o física) |
| Secció comprador | Es mostren les dades del comprador, tant si es tracta d'una persona física com jurídica. Concretament es mostra el nom i cognoms o raó social, i el CIF o NIF segons apliqui. |

| | |
|---|---|
| Secció Imports | Es mostren els imports de la factura:<br>• Import brut (import abans descomptes i taxes)<br>• Descompte (l'import total de descomptes aplicats, si n'hi han).<br>• Base imposable(import resultat d'aplicar a l'import brut els descomptes)<br>• Taxes (import total suma de totes les taxes aplicades del tipus que correspongui: IVA, IGIC...)<br>• A pagar (import total, suma de les taxes i de la base imposable). |
| Desglossament | Es mostren les línies de detall o conceptes de la factura (aquesta secció conté un scroll). Per cada línia de desglossament es mostra:<br>• El número d'unitats.<br>• El concepte<br>• El preu unitari<br>• El tipus d'impost aplicat (IVA, IGIC... segons el percentatge que apliqui).<br>• El total resultat de: Nº d'unitats * Preu unitari + taxes. |

**Taula 8: descripció dels camps del detall d'una factura**

Cal comentar que la moneda utilitzada per presentar els imports depèn de la moneda utilitzada en l'emissió de la factura, podent ésser: euros, dòlars, yens ...

# 8. Joc de proves

Al llarg del desenvolupament s'han anat fent proves unitàries i proves funcionals diverses, comprovant tots els casos d'ús que presenta l'aplicació. No obstant les proves més exhaustives s'han dut a terme en la validació de les factures rebudes (tant de la signatura com del processat del XML). En aquesta secció, no es presenta la bateria de proves realitzada, sinó la metodologia seguida per fer les proves i com s'han generat els jocs de proves.
Per altra banda, el desenvolupament s'ha realitzat seguint un cicle evolutiu, és a dir, s'implementava un cas d'ús, a continuació es realitzaven proves unitàries i funcionals, detectant els punts de fallada, i a continuació es corregien, per tornar a començar, fins a obtenir un cas d'ús estable. A continuació es passava a la implementació del següent cas d'ús i es seguia el mateix procediment.

## 8.1. Validació de la signatura dels documents

L'aplicació està implementada per validar qualsevol tipus de document signat XADES, no tan sols una factura. El mòdul de descobriment permet rebre documents diversos, no obstant, sinó es tracta de factures no es podran incorporar a l'aplicació i lògicament no es podran pre-visualitzar les dades del document, a excepció de les corresponents a la signatura. En aquest sentit fent "googling" ens hem baixat documents diversos signats amb XADES, així com factures de proves i documents signats del MINETUR. A continuació s'han cobert diversos escenaris de prova, encara que no s'ha pogut fer la prova de signatura de certificats revocats, ni d'escenaris complexes de testejar com potser l'estat del certificat desconegut. Les proves realitzades es presenten a

continuació a la Taula 9:

| Nº | Fitxer | Tipus Signatura | Tipus de Certificat | Resultat de la validació | OCSP | Dades addicionals | Observacions |
|---|---|---|---|---|---|---|---|
| T01 | t01_sense_signatura | N/A | N/A | FirmaXMLError: Signatura invàlida. No s'ha pogut trobar el node signatura. | N/A | | |
| T02 | t02_xades_bes_caducat | XADES-BES | DNIe | ResultadoEnum.INVALID (isValid=false) &ConfianzaEnum.SIN_CONFIANZA ----> [Veure objecte: datosFirma.politicas] | N/A | | |
| T03 | t03_xades_sense_confiança | XADES-BES | DNIe | ResultadoEnum.VALID (Signatura Vàlida) & ConfianzaEnum.SIN_CONFIANZA | UNKNOWN | | |
| T04 | t04_xades_alterat | N/A | N/A | ResultadoEnum.INVALID (isValid=false)& ConfianzaEnum.SIN_CONFIANZA | N/A | Signatura invàlida (signatura i/o certificats alterats) | |
| T05 | t05_xades_bes_caducat | XADES-BES | DNIe | ResultadoEnum.INVALID (isValid=false) &ConfianzaEnum.SIN_CONFIANZA ----> [[Veure objecte: datosFirma.politicas] | N/A | current time: Fri May 10 22:36:01 CEST 2013, expiration time: Tue Mar 02 18:44:08 CET 2010 | |
| T06 | t06_xades_epes_caducat | XADES-EPES | FNMT | ResultadoEnum.INVALID (isValid=false) &ConfianzaEnum.CON_CONFIANZA | N/A | current time: Fri May 10 23:16:47 CEST 2013, expiration time: Thu Nov 27 10:12:47 CET 2008 | |
| T07 | t07_xades_bes_valid (múltiples signatures) | XADES-BES | DNIe | ResultadoEnum.VALID (Signatura Vàlida) & ConfianzaEnum.CON_CONFIANZA | GOOD | | Tots els certificats inclús els de la cadena són vàlids |
| T08 | t08_xades_epes_caducat | XADES-EPES | DNIe | ResultadoEnum.INVALID (isValid=false) &ConfianzaEnum.SIN_CONFIANZA ----> [[Veure objecte: datosFirma.politicas] | N/A | current time: Fri May 10 23:27:07 CEST 2013, expiration time: Wed Dec 10 17:46:25 CET 2008 | |
| T09 | t09_xades_epes_valid | XADES-EPES | FNMT | ResultadoEnum.VALID (Signatura Vàlida) & ConfianzaEnum.CON_CONFIANZA | UNKNOWN | | |
| T10 | t10_xades_epes_valid | XADES-EPES | FNMT | ResultadoEnum.VALID (Signatura Vàlida) & ConfianzaEnum.CON_CONFIANZA | UNKNOWN | | |
| T11 | t11_xades_epes_valid | XADES-EPES | FNMT | ResultadoEnum.VALID (Signatura Vàlida) & ConfianzaEnum.CON_CONFIANZA | UNKNOWN | | |

**Taula 9. Proves de validació de signatura**

Aquests tests s'han realitzat accedint a les variables internes del programa, concretament s'utilitza l'enumeració `ResultadoEnum` i `ConfianzaEnum` (de les API del MINETUR) per emmagatzemar la validesa i el grau de confiança de la signatura.

`ResultadoEnum` pot prendre els següents valors: UNKNOWN, VALID, i INVALID, i `ConfianzaEnum` els valors NO_REVISADO, SIN_CONFIANZA, CA_CONFIANZA, CON_CONFIANZA.

Per altra banda, la descripció de les columnes de la taula és la següent:

- Nº: indica el nombre de test executat.
- Fitxer: nom del fitxer testejat, cadascun dels fitxers té un nom descriptiu, que es correspon amb el resultat de validació esperat.
- Tipus de signatura: el tipus de signatura XADES amb el que el document ha estat signat, o N/A si el document ha estat alterat o és sense signatura.
- Tipus de certificat: el tipus de certificat emprat per la signatura, només es suporten DNIe i els expedits per la FNMT.
- Resultat de la validació: en base al valor de les enumeracions mencionades en el paràgraf anterior.
- OCSP: resultat de la validació OCSP, potser GOOD, REVOKED o UNKNOWN.
- Dades addicionals: bàsicament s'indiquen els motius per els quals es rebutja la signatura.
- Observacions: consideracions addicionals.

Bàsicament les proves que s'han fet han estat de:

- Fitxer sense signatura, nom de la prova: T01.
- Fitxer caducat (sense confiança o amb confiança), nom de les proves: T02, T05, T06 i T08.
- Fitxer alterat després de la signatura, nom de la prova: T04.
- Fitxer amb signatura vàlida però sense confiança, nom de la prova: T03.
- Fitxers amb signatura vàlida i amb confiança, nom de les proves: T07, T09, T10 i T11.

Val a dir que la comprovació OCSP només s'ha fet per els certificats vàlids, obtenint els resultats GOOD i UNKNOWN.

Per altra banda, les proves T09, T10 i T11 corresponen a factures amb format Factura-2 v3.2.

Per últim, s'ha contrastat que la informació que es mostra per la pantalla de pre-visualització de la factura (veure figura 27).

## 8.2. Validació de format de les factures

El primer pas per validar les factures es poder generar factures vàlides d'acord l'especificació de format de Factura-e versió 3.2. El MINETUR té una aplicació lliure que permet emetré factures d'acord amb aquesta especificació, aquesta es pot descarregar de la seva web.
A partir d'aquí s'ha procedit a la descarrega de l'aplicació, i a emetre factures fictícies signades amb un certificat de la FNMT. L'aplicació del MINETUR té el següent

aspecte:

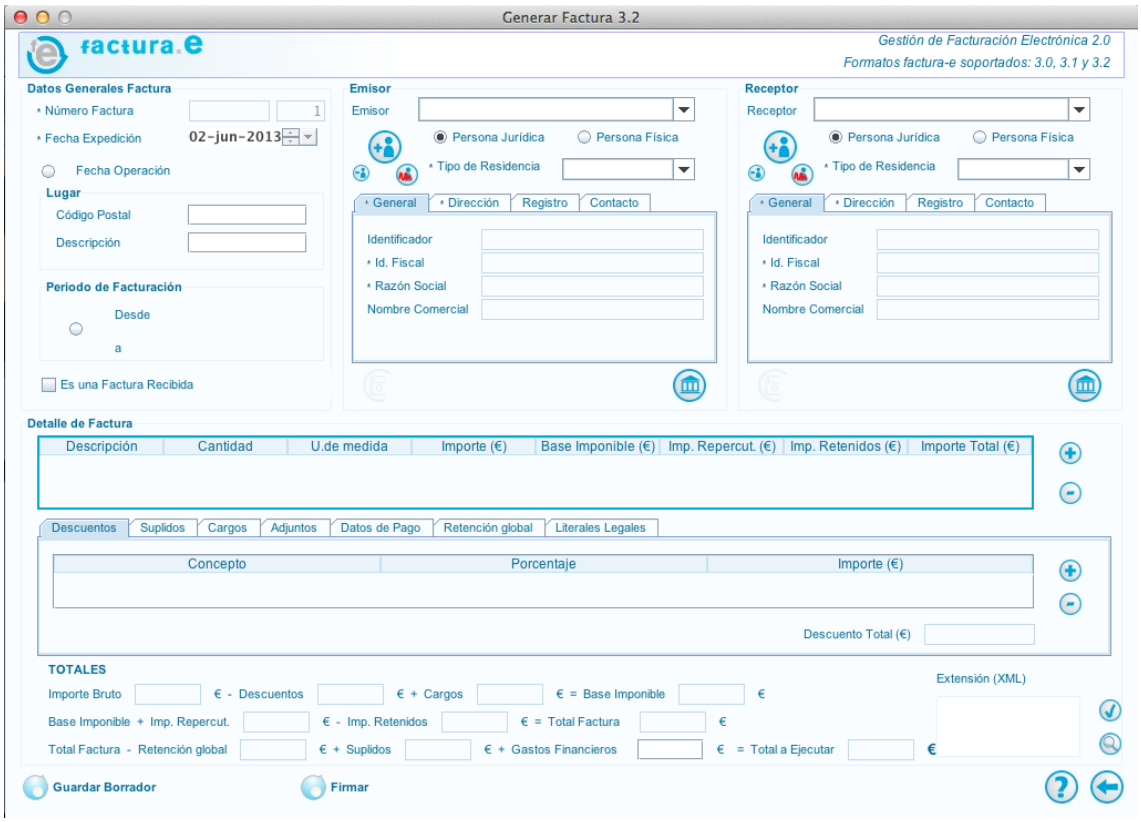


**Figura 33. Captura de pantalla del programa Factura-e del MINETUR**

Generades diverses factures, signades i carregades a l'aplicació, posteriorment s'han contrastat les dades carregades i que es visualitzaven a l'aplicació contra les del fitxer electrònic o Factura-e, gràcies a aquestes verificacions s'han pogut detectar les següents incidències entre d'altres:

1. Les factures es creaven dos cops.
2. No es desaven les dades de l'emissor quan aquest era una persona física.
3. Els totals de les factures es perdien degut a una errada en la classe `PropertyUtils` utilitzada per copiar propietats d'un objecte a d'altres.
4. A la pantalla de pre-visualització no apareixia la raó social.
5. L'import brut i la base imposable apareixien intercanviats.
6. Quan una factura per l'emissor o receptor no tenia dades de contacte es produïa una errada que provocava que l'aplicació es tanqués.

Com a observació, dir que encara que en aquest punt no es presenta un pla de proves exhaustiu, cal dir que les proves realitzades ho han estat.

## 8.3. Verificació de la correcta creació de la BBDD

Per tal de verificar la correcta creació de la base de dades s'han habilitat les traces de l'aplicació, podent determinar possibles falles durant la creació d'aquesta, es pot veure el registre de creació a l'annex 10.7.
Per altra banda, s'ha verificat que la BBDD estigués realment xifrada i que la informació no es mostrés en text clar. Per fer-ho s'ha utilitzat l'emulador d'Android per tal de recuperar la BBDD creada i descarregar-la a l'amfitrió, a partir d'aquí amb l'eina "hexdump" (veure annex) de UNIX s'ha bolcat el contingut per comprovar que realment era així.

# 9. Conclusions

L'objectiu principal d'aquest projecte era desenvolupar una aplicació de tiqueting per a dispositius mòbils, dins de l'àmbit del comerç electrònic, amb les garanties adients a nivell de seguretat, així com, dotar-la funcionalment dels requisits jurídics i legals adients en un sistema d'aquestes característiques, i es pot ben dir, que els objectius han estat assolits.

S'ha obtingut un producte final que garanteix per una banda la transferència de tiquets electrònics entre el dispositiu de servidor, corresponent al venedor, i el dispositiu mòbil del comprador, d'una forma segura mitjançant la tecnologia Bluetooth (i en un futur NFC). I per altre banda, els requeriments de seguretat: confidencialitat i integritat, s'han complert doncs s'ha assolit implementar un sistema de emmagatzematge segur amb la informació xifrada, així com, s'ha pogut aconseguir també la validació de la signatura digital i del document XML o factura en el propi dispositiu mòbil. Aquestes funcionalitats es plantejaven molt complexes, fins i tot, tant com per qüestionar la viabilitat del projecte, lo qual ha estat un gran èxit. Aquest darrer fet garanteix que la informació viatgi lo mínim necessari a través de xarxes de comunicacions, amb la conseqüent millora del nivell de seguretat. A més a més seguint amb lo relatiu a la seguretat, és garanteix l'accés a l'aplicació, única i exclusivament, per el propietari del dispositiu mòbil, gràcies al sistema d'autenticació.

En lo que respecta als requisits jurídics i legals, s'ha assolit implementar els requeriments previstos, fent servir un format estàndard de factura electrònica (del MINETUR) que garanteix la validesa dels tiquets o factures en qüestió, i gràcies a la signatura, l'autenticitat i el no repudio els requeriments pertinents.

Cal dir, que aquest projecte, tot i encara, que en termes pràctics es tracta d'una "petita aplicació", la feina que ha comportat ha estat considerable, pel fet de la complexitat de portar un sistema de validació d'aquestes característiques a la plataforma mòbil (Android).

## 9.1. Opinió personal

Ja en un bon principi quan vaig conèixer el projecte em va semblar molt interessant, i una molt bona idea, força avantguardista. El futur o el present, està en les tecnologies innovadores, i muntar un sistema de tiqueting que utilitzin els propers sistemes de pagament electrònic amb dispositius mòbils amb les mateixes garanties que un sistema de pagament "físic" o tradicional, sota el meu punt de vista, és sense cap mena de dubte una idea innovadora.
Poder implementar un sistema d'aquestes característiques, es participar en la innovació, i es força gratificant, més encara quan veus que assoleixes els objectius marcats en un principi.

## 9.2. Treball Futur

Tot i encara que els requeriments principals han estat assolits, encara resta molta feina per fer. Primer de tot, es vol publicar aquest projecte en un repositori de codi obert amb llicència LGPL (cal estudiar quin i la forma), podria servir en un futur proper per nous projectes tecnològics.
Tant però, restarien per implementar tota una sèrie de funcionalitats addicionals, aquestes es citen a continuació:

- Possibilitat d'utilitzar NFC per les comunicacions entre emissor i receptor.
- Poder assignar colors a les categories.
- Poder ordenar els llistats de factures per diferents criteris.
- Implementar bloqueig temporal de l'aplicació quan es detecti un nombre determinat d'intents d'entrar a l'aplicació sense èxit.
- Implementar funcionalitat de recordatori de la paraula de pas en cas de que aquesta sigui oblidada, es podria gestionar la recuperació per correu electrònic.
- Implementar tota la part de consums, en la que es mostrin estadístiques i llistats de consums agrupats per diferents criteris.
- Implementar la secció de preferències de l'aplicació, en la que l'usuari pugui configurar certs aspectes, com per exemple: l'adreça dels servidors OCSP a utilitzar, si es vol realitzar aquest tipus de validació o no, la configuració del compte de correu electrònic...
- Poder instal·lar certificats de noves CAs, des de la interfície d'usuari sense tenir que modificar el codi de l'aplicació.
- Crear aplicació en el núvol amb la que poder sincronitzar de forma segura la informació que conté el dispositiu mòbil.

Algunes d'aquestes funcionalitats deurien ser implementades abans de publicar el projecte com a programari lliure, per exemple: les comunicacions NFC, la consulta de consums i les preferències.

# 10. Annexes

## 10.1. Glossari

**ACID (Atomitcity, Consistency, Isolation, Durability):** són un conjunt de propietats que garanteixen que les transaccions realitzades sobre una BBDD son realitzades de forma correcta/ consistent.

**Activity**: activitat en el context de Android, és la classe base utilitzada per implementar qualsevol aplicació Android que tingui interfície gràfica.

**ADT (Android Development Tools)**: acrònim de les eines de desenvolupament d'Android.

**AES (Advanced Encryption Standard):** és un algorisme de xifrat per blocs adoptat com a estàndard de xifrat per al govern dels Estats Units.

**API (Application Programming Interace)**: Interfície de Programació d'Aplicació, conjunt de mètodes i funcions que proveeixen un conjunt determinat de llibreries, i que aporta una

capa d'abstracció a les funcionalitats que les llibreries ofereixen.

**APK (Application PacKage File):** és un acrònim que fa referència als tipus d'arxius instal·lables per la plataforma Android.

**Amfitrió**: l'amfitrió d'un sistema virtual és la pròpia màquina física.

**BO (Business Object)**: patró de disseny del programari utilitzat per encapsular la lògica de negoci de les unitats funcionals, desacoblant-la d'altres capes, com per exemple la capa de presentació.

**BBDD:** acrònim de base de dades.

**BKS (BouncyCastle KeyStore)**: magatzem de claus/ certificats de Bouncy Castle.

**Bytecode**: codi intermig generat per un compilador, el contingut del qual es executable per una màquina virtual o una màquina.

**CA (Certification Authority):** és tracta d'una entitat certificadora de confiança capaç d'emetre o revocar certificats digitals, utilitzats en la signatura electrónica.

**Classpath**: variable d'entorn que indica a la màquina virtual de Java on cercar les llibreries o classes de l'aplicació.

**COMMIT**: o confirmació d'una transacció, és l'operació que permet que els canvis realitzats es facin permanents a la base de dades.

**CRUD (Create, Read, Update and Delete):** acrònim/ sigles de les operacions d'accés i manipulació de dades de Creació, Lectura, Actualització i Esborrat.

**DAO (Data Object Access)**: patró de disseny que abstrau la forma d'accés a la BBDD de les operacions que es realitzen sobre aquesta.

**DDL (Data Definition Language)**: es tracta del llenguatge que permet a través de sentències, típicament SQL manipular l'estructura de la base de dades. Mitjançant aquest llenguatge es podran crear, modificar i esborrar: taules, vistes... i en general tota mena d'objectes de la BBDD.

**DML (Data Manipulation Language):** es tracta del llenguatge que permet la manipulació de la informació que està continguda en la BBDD.

**Des-serialització**: procés invers a la serialització.

**Fragment**: fragment en el context d'Android, és un component de la interfície gràfica que permet definir pantalles adaptables a diversos dispositius mòbils (telèfons, tablets ...)

**IDE (Integrated Development Environment)**: acrònim de entorn integrat de programació. És el programari utilitzat per al desenvolupament, inclou eines d'edició, compilació, i altres tipus d'utilitats encaminades a poder desenvolupar d'una forma més àgil.

**IMEI (International Mobile Identity):** és tracta d'un codi gravat en el terminal mòbil, que identifica unívocament el terminal a nivell mundial. Aquest codi es tramés a la xarxa per l'aparell quan aquest és connecta a aquesta.

**JKS (Java Keystore):** magatzem de certificats utilitzat per Java. Emmagatzema certificats d'entitats certificadores de confiança o claus públiques de certificats.

**JVM (Java Virtual Machine):** acrònim de la Màquina Virtual de Java.

**Layout**: paraula anglesa que en el context de l'aplicació context de l'aplicació fa referència a un component de la interfície gràfica que permet distribuir els elements gràfics en la pantalla del dispositiu.

**Listener**: o escoltador, es tracta de classes que reaccionen o reben events, generalment provinents de la interfície gràfica, com a conseqüència d'accions que realitza l'usuari. Aquests events són processats i desencadenen altres accions o funcionalitats com a resposta de l'esmentada interacció.

**MINETUR**: sigles del Ministeri d'Indústria, Energia i Turisme del Govern Espanyol.

**MODEL E/R**: el model Entitat/Relació permet definir el disseny i modelat d'un sistema de informació, aquest model servirà a posteriori per crear la base de dades relacional que permetrà emmagatzemar la informació del sistema.

**OCSP (Online Certificate Status Protocol)**: protocol utilitzat per la comprovació de l'estat del certificat, bàsicament permetrà comprovar si el certificat es revocat o per al contrari, és confia plenament en aquest.

**Plugin:** component de programari que aporta unes funcionalitats específiques.

**Rollback**: en una transacció es l'operació de desfer els canvis en cas de detectar una errada, per tal de que aquests no es facin persistents a la BBDD.

**Serialització**: procés de transformació de les dades d'un objecte (p.e: Java) a fitxer (p.e: XML).

**Singleton:** patró de disseny de programari, utilitzar per restringir el número de instàncies d'una classe, de forma que existeixi un únic objecte pertanyent a la classe especificada.

**SQL92 (Structured Query Language)**: es tracta d'un llenguatge estàndard utilitzat per fer operacions sobre una base de dades de tipus relacional.

**Transfer Object o DTO (Data Transfer Object)**: es un patró de disseny de programari utilitzar per tal de passar dades entre uns subsistemes i d'altres.

**UI (User Interface):** acrònim de interfície d'usuari, fa referència a la interfície gràfica amb la que interactua l'usuari.

**UUID (Universally Unique Identifier):** número de 128 bits, que identifica un servei bluetooth.

**XADES**: en anglès XML Advanced Electronic Signature, es un conjunt d'extensions a les

recomanacions del format XML-DSig per adaptar-les a la signatura electrònica avançada.

**XADES-EPES**: tipus de signatura XADES que conté a més a més dades sobre la política de signatura.

**XML (eXtensible Markup Language)**: llenguatge de marques desenvolupat per World Wide Web Consortium (W3C), i que permet la definició de pseudo-llenguatges propers al llenguatge humà, definint una gramàtica específica per a uns tipus de documents específics.

**XML-Dsig**: signatura XML, conjunt de recomanacions publicades per W3C que defineixen una sintaxi XML per la signatura digital.

**XSD (XML Schema):** llenguatge utilitzat per especificar i descriure l'estructura d'un document XML de forma precisa, més enllà de les regles sintàctiques que especifica el propi XML.

## 10.2. Construcció del projecte amb Maven

Com ja s'ha comentat al llarg del projecte, de cara a la construcció del projecte s'utilitza Maven, es recomana fer la construcció per mitjà del plugin d'Eclipse: m2eclipse, en cas d'utilitzar aquest, en cas contrari es tindria que realitzar la descarrega de Maven i fer la construcció des de la consola. Per tal d'instal·lar el plugin es pot seguir la següent guia: http://eclipse.org/m2e/download/.

Una vegada instal·lat el plugin sobre Eclipse, caldrà definir una ubicació al sistema de fitxers per al repositori local de Maven, així com, definir un fitxer de configuració de Maven, típicament anomenat, *settings.xml*, on es configuren les adreces dels repositoris públics d'Internet, així com, la ubicació del repositori local. A continuació es mostra un exemple:

```
<settings xmlns="http://maven.apache.org/SETTINGS/1.1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.1.0
http://maven.apache.org/xsd/settings-1.1.0.xsd">
   <localRepository>/Users/mike/UOC/mvn/repository</localRepository>
   <profiles>
    <profile>
    <id>maven-repository</id>
    <activation>
        <activeByDefault>true</activeByDefault>
    </activation>
    <repositories>
        <repository>
            <id>ibiblio</id>
            <name>ibiblio</name>
            <url>http://www.ibiblio.org/maven/</url>
        </repository>
         <repository>
            <id>maven2.java.net</id>
            <name>maven2</name>
            <url> http://download.java.net/maven/2/</url>
        </repository>
```

```
        <repository>
            <id>central</id>
            <name>maven2</name>
            <url>http://repo1.maven.org/maven2</url>
        </repository>
        <repository>
            <id>apache.snapshots</id>
            <name>snapshots</name>
            <url>http://people.apache.org/repo/m2-snapshot-repository</url>
        </repository>
    </repositories>
    </profile>
</profiles>
</settings>
```

Vegis al principi, marcat en groc, la ubicació del repositori local en el sistema de fitxers.

Una vegada realitzada la configuració anterior, configurarem Eclipse per tal de que el plugin de m2e agafi l'esmentada configuració, per això anirem a les preferències de Eclipse, i sota la opció de menú Maven > User Settings, definirem la ubicació del fitxer anterior i del repositori local. Vegis la següent figura:

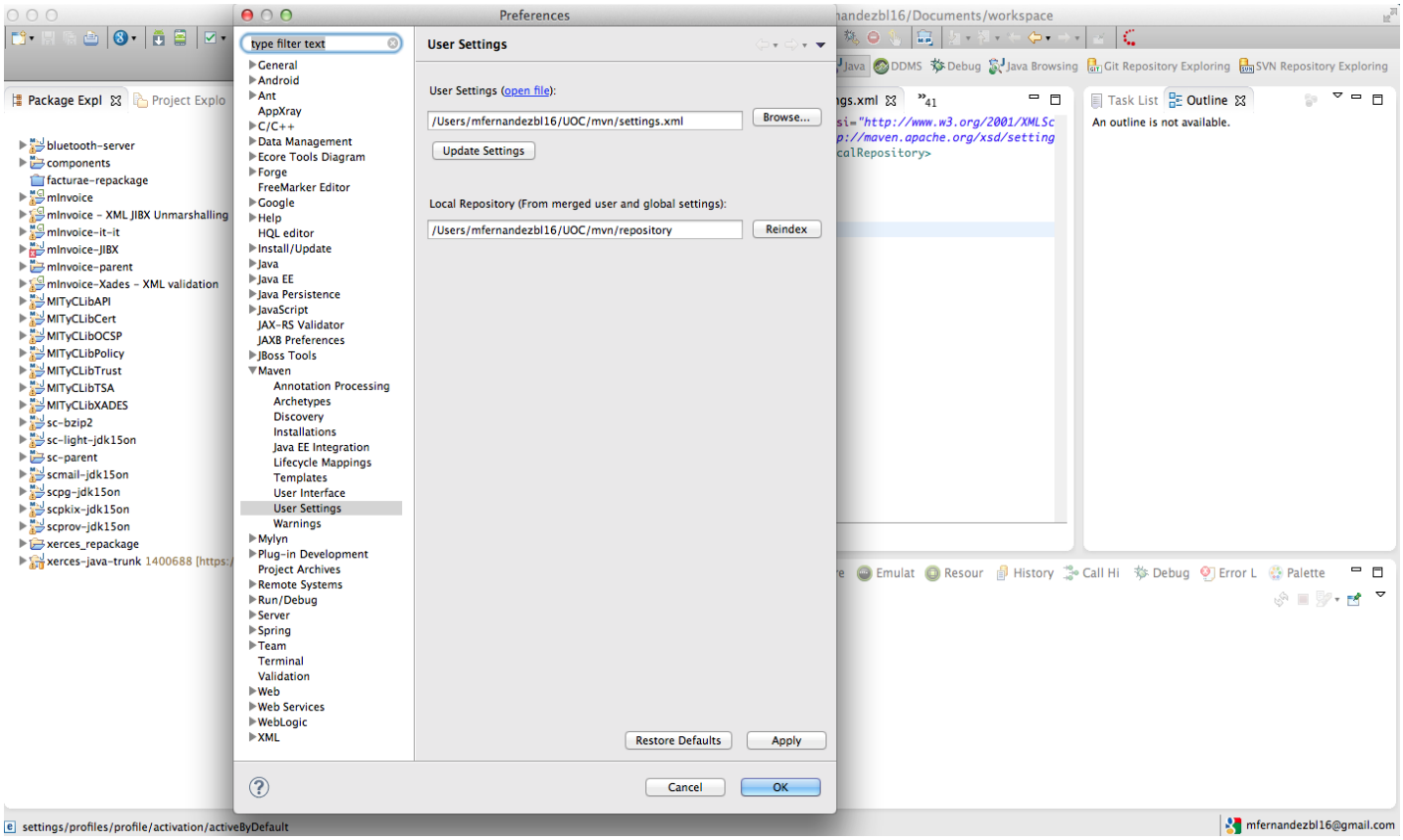


**Figura 34. Especificant les preferències de Maven**

Una vegada especificades les preferències es pot fer la construcció del projecte, col·locant-nos sobre el mòdul mInvoice o mInvoice-parent segons desitgem, i prement el botó dret del ratolí sobre aquest, per tal de a continuació i seleccionar "Run as" i a continuació "Maven install" (si volguéssim esborrar una versió prèviament generada faríem "Maven Clean" i després "Maven Install"), vegis la següent figura:

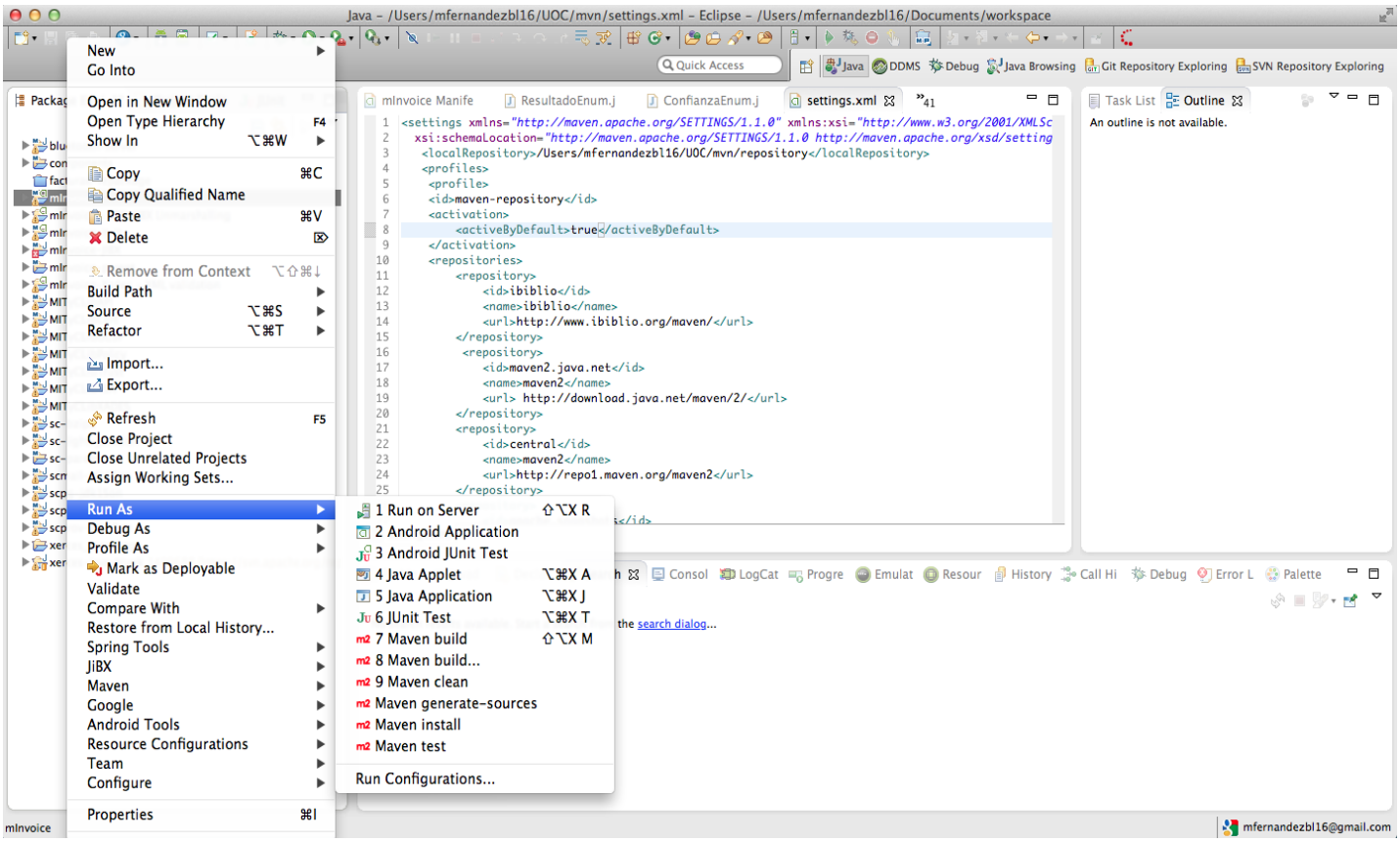


**Figura 35. Fent la construcció amb Maven**

## 10.3. Incorporació de llibreries amb Maven

En alguns casos les llibreries que utilitzem poden no trobar-se en repositoris públics, com és el cas de les llibreries de SQLCipher, en aquest cas tindrem que realitzar la instal·lació manual al repositori local, per després afegir la dependència al nostre projecte.
Per instal·lar les llibreries de forma manual tindrem que tenir instal·lat Maven, a continuació obrirem un terminal, en sistemes Unix (p.e: bash), o la consola de DOS de Microsoft, en sistemes Windows, a continuació en la variable d'entorn PATH apuntarem a la instal·lació de Maven, i definirem la variable d'entorn JAVA_HOME apuntant al directori de instal·lació del JDK (Java Development Kit).

Posteriorment llençarem la comanda "mvn install:install-file" sobre la llibreria en qüestió.
Per finalitzar aquest apartat, a continuació es mostra un exemple de comanda llençada per instal·lar les llibreries de SQLCipher:

```
localhost:libs mfernandezbl16$ mvn install:install-file -Dfile=sqlcipher.jar -
DgroupId=sqlcipher -DartifactId=sqlcipher -Dversion=2.1.1 -DgeneratePom=true -
Dpackaging=jar --settings /Users/mfernandezbl16/UOC/mvn/settings.xml

[INFO] Scanning for projects...
```

```
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building Maven Stub Project (No POM) 1
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-install-plugin:2.3.1:install-file (default-cli) @ standalone-pom ---
[INFO] Installing /Users/mfernandezbl16/UOC/TFM/recursos/SQLCipher for Android 2.1.1/libs/sqlcipher.jar to /Users/mfernandezbl16/UOC/mvn/repository/sqlcipher/sqlcipher/2.1.1/sqlcipher-2.1.1.jar
[INFO] Installing /var/folders/7_/tf0m782n3k72l1xytpxk3lth0000gn/T/mvninstall5786253491432744860.pom to /Users/mfernandezbl16/UOC/mvn/repository/sqlcipher/sqlcipher/2.1.1/sqlcipher-2.1.1.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 0.479s
[INFO] Finished at: Sun Apr 07 14:18:50 CEST 2013
[INFO] Final Memory: 5M/84M
[INFO] ------------------------------------------------------------------------
```

## 10.4. Re-empaquetat de llibreries per Android

En alguns casos s'han hagut de re-empaquetar llibreries per la plataforma Android, com és el cas de les llibreries de Xerces, per assolir aquest objectiu s'ha seguit el següent document: https://code.google.com/p/dalvik/wiki/JavaxPackages

En el cas de Xerces, s'ha creat un mòdul "xerces_repackage", en el directori "libs" es col·loquen les llibreries a re-empaquetar, i posteriorment es re-empaquetaran, per fer-ho s'utilitzarà la tecnologia Ant, sobre el fitxer de construcció "build.xml", en el qual especificarem el nom que prendran els nous paquets re-empaquetats:

```
<!-- Converts this project's .class files into .dex files -->
<target name="-jarjar" depends="-compile">
    <taskdef name="jarjar" classname="com.tonicsystems.jarjar.JarJarTask"
             classpath="buildtools/jarjar-1.4.jar"/>
    <jarjar jarfile="${out.absolute.dir}/xercesImpl-2.9.0-ae.jar">
        <fileset dir="${out.classes.absolute.dir}" />
        <zipgroupfileset dir="${external.libs.absolute.dir}"
                         includes="*.jar" />
        <rule pattern="org.apache.**" result="edu.mistic.org.apache.@1"/>
    </jarjar>
</target>
```

## 10.5. Creació de magatzem de certificats BKS

Com s'ha comentat anteriorment ha estat necessari definir un magatzem de certificats BKS (BouncyCastle KeyStore) per emmagatzemar els certificats de confiança de les CA.
Per fer-ho ens haurem d'haver descarregat les llibreries bcprov-jdk15-1.43.jar, posteriorment tindrem que utilitzar l'eina "keytool" del JDK. Veiem la generació en

qüestió del magatzem de certificats:

```
mfernandezbl16$ keytool -genkey -alias minvoice -keystore minvoice -storepass
mistic -storetype BKS -provider
org.bouncycastle.jce.provider.BouncyCastleProvider -providerpath ./bcprov-
jdk15-1.43.jar

¿Cuáles son su nombre y su apellido?
  [Unknown]:  Moisés Fernández Blanco
¿Cuál es el nombre de su unidad de organización?
  [Unknown]:  Seguretat en Aplicacions Web
¿Cuál es el nombre de su organización?
  [Unknown]:  MISTIC
¿Cuál es el nombre de su ciudad o localidad?
  [Unknown]:  BCN
¿Cuál es el nombre de su estado o provincia?
  [Unknown]:  BCN
¿Cuál es el código de país de dos letras de la unidad?
  [Unknown]:  ES
¿Es correcto CN=Moisés Fernández Blanco, OU=Seguretat en Aplicacions Web,
O=MISTIC, L=BCN, ST=BCN, C=ES?
  [no]:  si

Introduzca la contraseña de clave para <minvoice>
    (INTRO si es la misma contraseña que la del almacén de claves):
```

Durant la creació se'ns demanaran tota una sèrie de dades necessàries per dur-la terme, i finalment és sol·licitarà la contrasenya per al magatzem de claus.

## 10.6. Registre de traces (Logging)

Acostuma a ser molt útil el registre de traces d'informació, depuració i/o error en un fitxer de test, de cara a poder resoldre incidències que es detectin a l'aplicació. En aquest cas i com s'ha dit anteriorment, s'utilitzen les llibreries de "Androlog", per incorporar-les al projecte ha estat necessari afegir la següent dependència en el fitxer "pom.xml":

```
<dependency>
      <groupId>de.akquinet.android.androlog</groupId>
      <artifactId>androlog</artifactId>
</dependency>
```

Una vegada afegida la dependència, s'ha creat el fitxer "androlog.properties" (ubicat al directori "androlog" del mòdul mInvoice), en aquest fitxer s'ha definit el nivell de traces:

```
androlog.active = true
androlog.default.level=VERBOSE
```

Els nivells de traces suportats són: VERBOSE, DEBUG, INFO, WARN, i ERROR. A partir d'aquí les traces a l'aplicació es registren de forma molt senzilla, veiem un exemple:

```
Log.d("Determinant si cal crear el truststore");
```

A continuació, s'haurà de copiar el fitxer "androlog.properties" al directori "/sdcard" del dispositiu mòbil. Això es pot fer amb l'eina "adb" del ADT:

```
adb push androlog.properties /sdcard/
```

Per últim, comentar que es pot consultar un detallat manual a l'adreça:

http://stand.spree.de/wiki_details_androlog

## 10.7. Registre de creació de la Base de dades

Com s'ha comentat a la secció de "Verificació de la correcta creació de la BBDD" s'han habilitat les traces per registrar la creació de la BBDD mitjançant ORMLite, a continuació es mostren aquestes traces:

```
05-18 12:03:02.301: I/TableUtils(20765): creating table 'category'
05-18 12:03:02.306: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `category`
(`DESCRIPTION` VARCHAR , `ID` INTEGER PRIMARY KEY AUTOINCREMENT )

05-18 12:03:02.311: I/TableUtils(20765): creating table
'invoicedocumenttype'
05-18 12:03:02.311: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS
`invoicedocumenttype` (`ACRONYM` VARCHAR , `DESCRIPTION` VARCHAR ,
`ID` INTEGER PRIMARY KEY AUTOINCREMENT )

05-18 12:03:02.316: I/TableUtils(20765): creating table 'invoiceclass'
05-18 12:03:02.316: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `invoiceclass`
(`ACRONYM` VARCHAR , `DESCRIPTION` VARCHAR , `ID` INTEGER PRIMARY KEY
AUTOINCREMENT )

05-18 12:03:02.321: I/TableUtils(20765): creating table 'currencycode'
05-18 12:03:02.321: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `currencycode`
(`CURRENCY_CODE` VARCHAR , `ID` INTEGER PRIMARY KEY AUTOINCREMENT )

05-18 12:03:02.326: I/TableUtils(20765): creating table 'countrycode'
05-18 12:03:02.326: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `countrycode`
(`COUNTRY_CODE` VARCHAR , `ID` INTEGER PRIMARY KEY AUTOINCREMENT )

05-18 12:03:02.331: I/TableUtils(20765): creating table
'taxoutputtype'
05-18 12:03:02.331: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `taxoutputtype`
(`CODE_TYPE` VARCHAR , `DESCRIPTION` VARCHAR , `ID` INTEGER PRIMARY
KEY AUTOINCREMENT )

05-18 12:03:02.336: I/TableUtils(20765): creating table 'address'
05-18 12:03:02.341: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `address`
(`ADDRESS` VARCHAR , `ID` INTEGER PRIMARY KEY AUTOINCREMENT ,
`m_CountryCode_id` INTEGER NOT NULL , `POST_CODE` VARCHAR , `PROVINCE`
VARCHAR , `TOWN` VARCHAR )

05-18 12:03:02.346: I/TableUtils(20765): creating table
```

```
'contactdetails'
05-18 12:03:02.346: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `contactdetails`
(`ADDITIONAL_DETAILS` VARCHAR , `CNO_CNAE` VARCHAR , `CONTACT_PERSONS`
VARCHAR , `EMAIL` VARCHAR , `FAX` VARCHAR , `ID` INTEGER PRIMARY KEY
AUTOINCREMENT , `INE_CODE` VARCHAR , `TELEPHONE` VARCHAR ,
`WEB_ADDRESS` VARCHAR )

05-18 12:03:02.351: I/TableUtils(20765): creating table 'individual'
05-18 12:03:02.356: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `individual`
(`FIRST_SURNAME` VARCHAR NOT NULL , `ID` INTEGER PRIMARY KEY
AUTOINCREMENT , `m_Address_id` INTEGER NOT NULL ,
`m_ContactDetails_id` INTEGER , `NAME` VARCHAR NOT NULL ,
`SECOND_SURNAME` VARCHAR )

05-18 12:03:02.361: I/TableUtils(20765): creating table
'businessparty'
05-18 12:03:02.361: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `businessparty`
(`CORPORATE_NAME` VARCHAR , `ID` INTEGER PRIMARY KEY AUTOINCREMENT ,
`m_Address_id` INTEGER NOT NULL , `m_ContactDetails_id` INTEGER ,
`TRADE_NAME` VARCHAR )

05-18 12:03:02.416: I/TableUtils(20765): creating table 'taxoutput'
05-18 12:03:02.416: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `taxoutput` (`ID`
INTEGER PRIMARY KEY AUTOINCREMENT , `m_InvoiceLine_id` INTEGER ,
`m_TaxOutputType_id` INTEGER NOT NULL , `TAX_RATE` FLOAT ,
`TAX_AMOUNT` DOUBLE PRECISION )

05-19 12:03:02.416: I/TableUtils(20765): creating table 'buyerparty'
05-19 12:03:02.416: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE `buyerparty`
(`IS_BUSINESS_PARTY` SMALLINT , `CORPORATE_NAME` VARCHAR ,
`FIRST_SURNAME` VARCHAR NOT NULL , `ID` INTEGER PRIMARY KEY
AUTOINCREMENT , `m_Address_id` INTEGER NOT NULL ,
`m_ContactDetails_id` INTEGER , `NAME` VARCHAR NOT NULL ,
`SECOND_SURNAME` VARCHAR , `IDENTIFIER_NUMBER` VARCHAR , `TRADE_NAME`
VARCHAR )

05-18 12:03:02.416: I/TableUtils(20765): creating table 'xmlinvoice'
05-18 12:03:02.421: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `xmlinvoice`
(`XML_DATA` BLOB , `ID` INTEGER PRIMARY KEY AUTOINCREMENT )

05-18 12:03:02.421: I/TableUtils(20765): creating table 'invoiceline'
05-18 12:03:02.421: D/dalvikvm(20765): GC_CONCURRENT freed 413K, 8%
free 12945K/13959K, paused 12ms+15ms, total 50ms
05-18 12:03:02.426: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `invoiceline`
(`ARTICLE_CODE` VARCHAR , `ID` INTEGER PRIMARY KEY AUTOINCREMENT ,
`ITEM_DESCRIPTION` VARCHAR , `m_Invoice_id` INTEGER NOT NULL ,
`QUANTITY` SMALLINT , `TOTAL_COST` DOUBLE PRECISION ,
`UNIT_PRICE_WITHOUT_TAX` DOUBLE PRECISION )

05-18 12:03:02.426: I/TableUtils(20765): creating table 'invoice'
```

```
05-19 12:03:02.426: I/TableUtils(20765): executed create table
statement changed 1 rows: CREATE TABLE IF NOT EXISTS `invoice`
(`xmlInvoice_id` INTEGER NOT NULL , `ID` INTEGER PRIMARY KEY
AUTOINCREMENT , `INVOICE_NUMBER` VARCHAR , `INVOICE_SERIES_CODE`
VARCHAR , `ISSUE_DATE` VARCHAR , `m_BusinessParty_id` INTEGER ,
`m_BuyerParty_id` INTEGER NOT NULL , `m_Category_id` INTEGER ,
`m_CurrencyCode_id` INTEGER , `m_Individual_id` INTEGER ,
`m_InvoiceClass_id` INTEGER NOT NULL , `m_InvoiceDocumentType_id`
INTEGER NOT NULL , `OPERATION_DATE` VARCHAR , `OUTSTANDING_AMOUNT`
DOUBLE PRECISION , `TOTAL_AMOUNT` DOUBLE PRECISION ,
`TOTAL_AMOUNT_BF_TAXES` DOUBLE PRECISION , `TOTAL_TAX_OUTPUTS` DOUBLE
PRECISION , `GENERAL_DISCOUNTS` DOUBLE PRECISION )
```

## 10.8. Utilització de hexdump

La eina “hexdump” ve incorporada en sistemes operatius Unix, en aquest cas s’ha utilitzat per verificar que el contingut de la BBDD és xifrat, la utilització de la comanda és molt senzilla, a continuació es pot veure un exemple on “dbname.db” és el nom de la BBDD a “bolcar”:

```
$ hexdump –C dbname.db
```

La sortida sobre una BBDD xifrada seria un conjunt de caràcters illegibles, no passaria lo mateix si la BBDD no estigués xifrada, ja que es podria visualitzar la informació que conté aquesta.

Per consultar, la documentació de “hexdump” es pot executar la comanda:

```
$ man hexdump
```