SENDCRYPT

Antoni Soldado Campos (ETIS)

Antoni Martinez Balleste

14/01/2008

Títol: Sendcrypt
Autor: Antoni Soldado Campos
Consultor: Antoni Martinez Balleste
Data: 14 de gener de 2008

Resum:

En aquest projecte he intentat crear una aplicació universal per qualsevol telèfon mòbil que compleixi uns mínims requisits que permeti a l'usuari enviar i rebre missatges SMS (Short Message Service) encriptats i així poder mantenir la privacitat d'aquests de la mateixa manera que ho fa un usuari de PGP (Pretty Good Privacy) o OpenPGP.

No hi ha cap operador de telefonia ni cap fabricant de terminals mòbils que ofereixi l'encriptació dels missatges dels usuaris pel que penso que aquest projecte permet solucionar un buit existent i que ateny al privacitat de l'usuari.

M'agradaria continuar amb aquest projecte i que aquest es convertís en una interfície criptogràfica que no només permetés a l'usuari enviar i rebre missatges SMS sinó que s'estengués a altres tipus de missatgeria on s'utilitzés el mòbil o mes generalment, Internet. Així doncs m'agradaria crear una interfície criptogràfica única on l'usuari pogués encriptar les seves comunicacions habituals: SMS, MMS (Multimedia Messaging Service), correus electrònics, converses de xat i compartició d'arxius.

Alhora, voldria que aquest projecte desenvolupat en Java per la seva universalitat en el camp de la telefonia mòbil, s'estengués a altres dispositius per tal de fer-ho el mes estàndard possible, i que qualsevol usuari de Apple Iphone, Google Android, Symbian OS Phone o Windows Mobile Phone Edition pogués beneficiar-se d'aquestes noves funcionalitats.

1. Cos de la memòria

1.1. Introducció

1.1.1. Justificació del TFC i context en el qual es desenvolupa

Són ja a Espanya 49 milions de línies mòbils. El que representa que estadísticament a cada habitant de l'estat li correspon més d'un número de mòbil (la població d'Espanya és de 45 milions de persones), a part, hi ha quatre operadors majoritaris (Movistar, Vodafone, Orange i Yoigo), i més de 100 sol·licituds de llicències d'OMV (Operador de Mòbil Virtual) de les quals, 30 ja poden operar (algunes d'elles ja ho fan: Carrefour, Eroski, Ono IO, Happy, ...).

Han passat ja més de quinze anys des d'aquell primer missatge enviat el tres de desembre de 1992 i que va fer que aquest servei passés a ser un mitjà de comunicació al igual que ho eren les trucades telefòniques, enlloc d'un servei per comunicar incidències utilitzat únicament per les operadores de telefonia mòbil.

S'estima que durant el 2007 s'han enviat arreu del mon 1,9 bilions de missatges (quasi un bilió més que l'any anterior). Només aquest cap d'any Movistar va gestionar 128 milions mitjançant la seva infraestructura.

Aquestes dades demostren que la telefonia mòbil ha arrelat en la societat actual, i que un dels serveis que ofereix, els missatges SMS, s'ha convertit en una part molt important de la facturació total de les empreses de telefonia.

Cada vegada més, la utilització dels missatges SMS ha anat evolucionant, inicialment com un mitjà per transmetre missatges puntuals ("Quedem a les 3" "He arribat" "Arribaré tard" ...) fins a un sistema de comunicació on s'estableixen veritables diàlegs al igual que es faria amb una trucada telefònica convencional (fins i tot amb un llenguatge propi).

No oblidem l'accés que mitjançant altres sistemes (pàgines web, missatgeria instantània, telefonia fixa, cabines telefòniques, ...) permeten accedir a aquest nou mitjà de comunicació.

Tot això ha fet que apareguessin infinitat de serveis que utilitzen els missatges SMS com a únic sistema de comunicació. D'aquests podríem destacar les compres electròniques (des de tons, vídeos i jocs pel telèfon mòbil com

empreses especialitzades en cobrament de productes mitjançant el terminal mòbil, per exemple Mobipay SA), els sistemes de posicionament GPS on s'envien les coordenades en missatges independents, els sistemes d'informació tant oficial (per exemple Hisenda) com comercial (empreses privades), els sistemes d'alertes (alarmes, events, noticies, ...), la gestió dels comptes bancaris personals i el control remot d'equipaments remots (ordinadors, maquinaria, ...).

Com es pot veure, els missatges electrònics contenen sovint dades confidencials. Dades privades que només haurien de esser llegides per l'emissor i el corresponent receptor.

Aquest és l'esquema principal per la utilització d'un sistema criptogràfic, però després de tantes paraules escrites, encara no ha aparegut cap paraula que parli sobre seguretat i confidencialitat, i molt menys sobre criptografia ni encriptació: No hi ha ningú, persona o entitat, que garanteixi a l'usuari que el seu missatge només el podrà llegir el seu destinatari legítim.

Excepcionalment, val a dir, que aquest gener, començarà a funcionar un OMV participada per KPN (empresa de telefonia mòbil a Holanda) i Bankinter, que inclourà en les targetes SIM opcions criptogràfiques i firma digital.

1.1.2. Objectius del TFC

Aquest projecte, SendCrypt, intenta oferir als usuaris de telefonia mòbil una manera de poder enviar i rebre missatges encriptats, garantint així que la informació viatja segura i fins i tot protegida davant de robatoris de terminals i d'intrusions en el canal de comunicació, i que arriba a la persona a la que va dirigida.

Inicialment consta d'una aplicació desenvolupada pels terminals mòbils que un cop instal·lada ofereix a l'usuari una nova interfície per enviar i rebre missatges SMS de forma encriptada. Aquesta nova aplicació no substitueix el sistema natiu que cada terminal incorpora per gestionar els missatges SMS, pel que l'usuari no perd en cap moment la comunicació tradicional que fins aleshores venia utilitzant.

Al igual que la nova aplicació manté una base de dades de missatges enviats, rebuts i eliminats, paral·lela a la pròpia de l'equip, també ho fa amb els contactes. L'aplicació manté una base de dades pròpia de contactes independent de la originaria del mòbil. Aquests dos punts poden semblar punts febles en l'aplicació, però també poden mantenir una independència entre els dos sistemes d missatgeries que en més d'una ocasió podria ser d'agrair. Això no treu que la millor prestació sigui sempre donar la possibilitat a l'usuari a

mantenir o no un doble sistema de missatgeria SMS així com una doble gestió de contactes.

En poc temps (sobretot amb l'implantació de les xarxes UMTS - Universal Mobile Telecommunications System -) s'ha fet una aproximació dels terminals mòbils cap a la tecnologia pròpia d'Internet. Exemples d'això son les aplicacions de missatgeria instantània (per exemple Microsoft Messenger) que ofereixen els proveïdors de servei, els serveis de correu en mobilitat en temps real (per exemple BlackBerry i RealMail), els serveis de VeuIP (per exemple Guizmo i Skype) i diverses aplicacions web adaptades al terminal mòbil (per exemple Google, Gmail i YouTube).

Aquest fet ha obligat als fabricants a oferir terminals mòbils amb prestacions que fins ara eren impensables en un telèfon mòbil, i no em refereixo només a prestacions hardware (com per exemple càmeres de quatre megapixels, connexions inalàmbriques 802.11g o receptors GPS - Global Positioning System - d'alta sensibilitat) sinó també a prestacions software (com per exemple sistemes operatius – Symbian, Windows Mobile o Android – o la possibilitat de desenvolupar aplicacions que funcionin en cadascun dels equips mitjançant diversos entorns de programació específics).

Tota aquesta continua i ràpida evolució que està prenent la telefonia mòbil és la que ha motivat que vegi aquest treball com el primer pas per crear un projecte més ambiciós que ofereixi a l'usuari l'encriptació de tota la comunicació escrita que utilitzi. M'agradaria poder continuar treballant per ampliar les prestacions de SendCrypt amb les millores que mes endavant relacionaré, així com exportant-lo a noves plataformes (per exemple terminals mòbils amb sistema operatiu Symbian o Android) i crear un portal web on l'usuari de SendCrypt pogués enviar i rebre els seus missatges SMS encriptats, i no només això, sino oferir-li de la mateixa manera l'enviament i la recepció de missatges MMS, l'enviament i recepció de missatges de correus electrònics, la possibilitat de mantenir converses escrites (missatgeria instantània) i la possibilitat de compartir arxius amb d'altres usuaris, tot dintre d'un marc segur, protegit i encriptat.

1.1.3. Enfocament i mètode seguit

Crear una aplicació que es pugui executar en el major nombre de terminals mòbils és parlar de Java Micro Edition. Hi ha altres llenguatges per altres sistemes minoritaris (Iphone-SDK pels terminals IPhone d'Apple, Llenguatge C o Java pels terminals Symbian, Android-SDK pels futurs terminals amb sistema operatiu Android o Microsoft .NET pels terminals Microsoft Windows Mobile, entre d'altres) però no són majoritaris en el mercat actual de la telefonia mòbil.

J2ME és una versió reduïda del llenguatge de programació Java de SUN Microsystems. Com la versió estandard d'aquest llenguatge, J2ME es

caracteritza per ser un llenguatge pseudo compilat que accedeix als recursos de cada dispositiu mitjançant API (Application Programming Interface – Interfície de programació d'aplicacions) d'alt nivell independents del hardware de la maquina que executa l'aplicació, ja que precisa d'una màquina virtual especifica per cada tipus de terminal que s'encarrega d'executar l'aplicació compilada en pseudocodi (bytecodi).

J2ME és un llenguatge no només utilitzat en telèfons mòbils, sinó que hi ha infinitat de dispositius que internament suporten l'execució d'aplicacions escrites en aquest llenguatge. Alguns dispositius que implementen aquest llenguatge de forma nativa podrien ser dispositius PDA, mòdems, descodificadors digitals de TV, receptors de satèl·lit, targetes intel·ligents d'accés i microprocessadors d'aplicacions universals.

He escollit doncs aquest llenguatge, J2ME, per escriure l'aplicació SendCrypt. En cap cas l'elecció s'ha fet valorant la dificultat, ni les prestacions, ni les limitacions, que ens ofereix J2ME (val a dir, que si hagués contemplat qualsevol d'aquestes tres opcions, no hauria escollit J2ME). He valorat únicament el nombre de potencials usuaris de l'aplicació, creient que un projecte d'aquestes característiques s'ha d'oferir inicialment al mercat majoritari, deixant per una segona ampliació els mercats minoritaris.

Concretament he desenvolupat SendCrypt per terminals mòbils compatibles amb l'especificació J2ME MIDP2 – JSR118 (Java Specification Request). MIDP (Mobile Information Device Profile) és una especificació implementada en la majoria de telèfons mòbils del mercat, sigui en la seva primera revisió (MIDP1 – JSR37) o en la segona i més actual (MIDP2 – JSR118). Aquest any passat és va publicar l'especificació MIDP3 – JSR271.

MIDP1 va aparèixer l'any 2001. Va ser la primera especificació i ara actualment està en desús degut a que no permet accedir als nous recursos integrats en els telèfons mòbils: blueTooth, missatgeria, sistema d'arxius, pantalla gràfica, pantalla complerta, àudio, ...

Com que la finalitat de SendCrypt és enviar i rebre missatges, he hagut de partir de la segona especificació (MIDP2) ja que la seva antecessora (MIDP1) no permet accedir al sistema de missatgeria dels terminals mòbils, i per tant, no permet ni rebre ni enviar missatges.

Un cop escollit el llenguatge a utilitzar vaig escollir l'entorn de programació idoni per tal de escriure el programa. Existeixen actualment diversos entorns de programació per desenvolupar aplicacions amb llenguatge Java; des de solucions comercials com JBuilder de l'empresa CodeGear (filial de l'empresa Borland) o Visual J++ de l'empresa Microsoft, fins a solucions lliures sota llicencies GPL (General Public License) i CDDL (Common Development and Distribution License) com Netbeans de SUN o llicències EPL (Eclipse Public License) com Eclipse.

Inicialment vaig estudiar les solucions lliures en les seves ultimes revisions (Eclipse 3.3.1.1 i Netbeans 5.5). Val a dir que Eclipse oferia un millor rendiment, era més flexible i ràpid però la seva estructura d'extensions no m'acabava de convèncer. No és la solució integral que a priori buscava per realitzar el projecte.

Netbeans 5.5 és un bon entorn però li mancava l'agilitat i flexibilitat que jo considero que havia de tenir un bon entorn de programació, a part, que tampoc era una solució integra sinó que precisava d'extensions per poder realitzar aplicacions per terminals mòbils, com la solució estudiada anteriorment.

El problema de les solucions comercials, a part del cost econòmic, és que no respectaven l'estàndard del llenguatge Java i cadascuna d'elles incloïa petites variacions del llenguatge que no el feien portable (una de les principals característiques d'aquest llenguatge), sobretot el producte de Microsoft.

Tot va canviar el tres de desembre de l'any passat al aparèixer la versió 6.0 de Netbeans. Aquest nou producte es distribueix mitjançant paquets ja creats que incorporen les diferents extensions que antigament s'afegien a part. I aquesta és el la interfície de desenvolupament escollida per aquest projecte: Netbeans 6.0 Mobility, que inclou l' interfície estàndard Netbeans per desenvolupar aplicacions Java i tots els complements necessaris per desenvolupar aplicacions per terminals mòbils, i sobretot, emuladors de les diferents variacions de J2ME per tal de veure el comportament que tindria l'aplicació en cadascun dels terminals existents.

Netbeans necessita que l'ordinador que l'executi tingui instal·lat l'aplicació JDK (Java Developers Kit) per tal de poder compilar les aplicacions que desenvolupem, però alhora, precisa de l'aplicació JRE (Java Runtime Environtment) ja que Netbeans està desenvolupat alhora en Java, i com hem comentat abans, Java és un llenguatge pseudocompilat que tradueix el codi font a un codi intermig anomenat ByteCode que precisa d'una màquina virtual especifica per cada processador per poder-lo executar.

En aquest cas, he optat per utilitzar la última versió, 1.6u3, tant de JDK com de JRE.

1.1.4. Planificació del projecte

Un cop definit el llenguatge i l'entorn de programació calia fer una recerca per tal de trobar llibreries auxiliars que em permetessin arribar a desenvolupar l'aplicació d'acord amb les funcionalitats previstes inicialment.

En primer lloc calia trobar una llibreria criptogràfica adient per l'aplicació: compatible amb l'entorn escollit, ràpida en l'execució en terminals mòbils i no lligada a cap llicència propietària que delimités la utilització d'aquesta per aquest projecte.

Un cop escollida aquesta part, calia cercar un sistema per emmagatzemar la informació que generaria el programa dintre dels terminals mòbils, de manera no volàtil, que no s'inicialitzés cada vegada que s'executés el programa.

I, per últim, cercar la possibilitat d'utilitzar alguna llibreria que facilités la generació de la interfície d'usuari i millorés la visibilitat en les pantalles dels terminals mòbils.

El projecte està construït de manera modular per tal de facilitar les tasques d'ampliació i modificació. Aquesta divisió s'ha realitzat depenent de les funcions de cadascuna de les parts, deixant doncs un projecte net i fàcil d'entendre, podent seguir l'execució i/o depuració d'aquest en tot moment.

Dintre del projecte Java, cadascuna de les parts està posada dintre d'un paquet (package), contenint aquest les diverses classes necessàries per tal de realitzar la tasca assignada.

En el següent esquema detallo cadascuna d'aquestes parts amb el nom del paquet assignat i les classes que el composen:

Tasca	Nom del paquet	Classes Compreses
Interfície	sendcrypt	SendCryptMIDlet
Comunicacions	comm	SMSComm SMSReciever SMSSender
Beans	beans	Contacte Missatge MissatgeEliminat MissatgeEnviat MissatgeRebut PrivateKey PublicKey
Criptografia	crypt	RSA
BBDD	bbdd	BBDDManager

1.1.5. Productes obtinguts

Com en totes les aplicacions desenvolupades en J2ME els programes preparats per instal·lar consten de dos arxius.

El primer, l'arxiu que té extensió JAD (Java Application Descriptor), és un arxiu de text que conté dades descriptives de l'aplicació per instal·lar. Les dades més importants que apareixen en aquest arxiu són el nom de l'aplicació, el tamany d'aquesta, la URL (Uniform Resource Locator) d'Internet des de on es pot descarregar i la variant de J2ME que ha de tenir el terminal mòbil per poder-la executar. A continuació poso un exemple genèric d'aquest tipus de fitxer:

MIDlet-1: <Application name>, <icon path> , <midlet class>
MIDlet-Jar-Size: <Size in bytes>
MIDlet-Jar-URL: <Associated JAR file>
MIDlet-Name: <Application name>
MIDlet-Vendor: <Company>
MIDlet-Version: 1.0 <Version number>
MicroEdition-Configuration: CLDC-1.1 <CLDC version>
MicroEdition-Profile: MIDP-2.0 <MIDP version>

El segon, el que té l'extensió JAR (Java ARchive), és un arxiu binari que conté comprimides totes les classes Java que necessita l'aplicació per poder-se executar junt amb d'altres arxius auxiliars com podrien ser els icones gràfics, els fons de pantalla o els arxius de configuració. A part, inclou també un arxiu de text amb les mateixes dades que el vist anteriorment. Aquest arxiu s'anomena MANIFEST.MF i està sempre d'una carpeta META-INF.

Actualment la majoria de terminals mòbils poden executar directament el fitxer binari, el d'extensió JAR, però encara n'hi ha d'altres que necessiten llegir el fitxer de text, el fitxer JAD i descarregar el fitxer binari JAR de la ruta especificada en el fitxer de configuració.

La generació d'aquests fitxers és realitza automàticament mitjançant l'entorn de desenvolupament, el Netbeans, malgrat que es pot generar amb les pròpies eines que Sun Microsystems facilita junt amb el JDK.

Junt amb aquesta memòria, adjuntaré els dos arxius amb l'aplicació preparada per ser executada en tots els terminals mòbils compatibles amb la variant J2ME MIDP2.

Habilitaré també una pàgina web (http://www.sendcrypt.es) per facilitar-vos la descarrega d'aquest document, de la presentació i dels arxius JAD i JAR.

La instal·lació de l'aplicació en cadascun dels terminals es pot fer de diverses maneres, des de enviar-li via BlueTooth o via infraroigs el fitxer JAR, fins a connectar-nos amb el navegador del terminal a la URL especificada anteriorment i descarregar l'arxiu JAD.

1.1.6. Breu descripció dels altres capítols de la memòria

En els següents capítols descriuré cadascun dels mòduls en que funcionalment he dividit l'aplicació, segons he comentat anteriorment en l'apartat 5.1.4 (Planificació del projecte).

Per cadascuna d'aquestes parts, explicaré la funció que realitza i els recursos que utilitza, així com les complicacions que he tingut alhora de desenvolupar-les i com les he intentat resoldre.

Com a introducció a cada capítol hi he afegit un diagrama UML (Unified Modeling Language) de les classes corresponents per tal de centrar l'explicació i veure en tot moment l'herència i les relacions entre elles.

La primera part, que he anomenat interfície, és la encarregada d'executar l'aplicació indicant-li al terminal mòbil els permisos necessaris que aquesta necessita. També és la responsable de generar els menús que veu l'usuari a treves de la pantalla així com de gestionar les seves respostes.

La segona part, que he anomenat comunicacions, és la encarregada d'enviar i rebre els missatges SMS amb que l'usuari es comunica. A aquest apartat pertany la classe que s'executa automàticament quan el terminal mòbil rep un missatge encriptat, estigui o no estigui l'aplicació activada.

La tercera part, que he anomenat beans, conté únicament estructures de dades per facilitar la tasca de manipulació d'informació en les classes de les altres parts.

La quarta part, que he anomenat criptografia, és la encarregada tant d'encriptar i de desencriptar com de crear la parella de clau privada i clau pública necessàries per el correcte funcionament de l'aplicació.

La cinquena i última part, que he anomenat BBDD, és la encarregada de gravar i recuperar la informació que l'aplicació va generant mitjançant la seva continua utilització.

1.2. Mòdul: Interfície

1.2.1. Descripció general

Aquest mòdul hereta de la classe Midlet del J2ME. Conté la classe que s'executa tan sols iniciar l'aplicació en el terminal mòbil, SendCrypeMIDlet. Aquesta classe conté també el fil principal d'execució de l'aplicació gestionant així l' interacció amb l'usuari (enviant les dades sensibles de ser impreses a la pantalla i recollint les opcions escollides per l'usuari per tal de executar les accions corresponents) i amb les altres classes i mòduls.

Val a dir que J2ME no genera un entorn visualment modern, sino més aviat, tot el contrari. Els entorns d'usuari creats amb J2ME són simples, senzills, poc flexibles i desfasats en el temps, sense cap tipus de disseny ni evolució que es pugui comparar amb entorns desenvolupats amb altres plataformes. I per acabar-ho d'empitjorar, no és un entorn estàtic, sinó que varia notablement depenent del terminal mòbil que l'executa.

Vaig intentar buscar alternatives visuals a aquesta anacrònica plantilla que ofereix J2ME i vaig trobar una alternativa molt interessant, J2ME Polish.

La vaig provar, però la complicació que afegia a la programació no es compensava amb els resultats obtinguts. Visualment millorava però la carrega de dificultat que comportava em va fer desistir en la seva utilització. No descarto però, que si el projecte SendCrypt continua amb noves ampliacions i prestacions, torni a provar d'utilitzar aquesta llibreria ja que l'entorn visual d'una aplicació és sovint quasi tant important com les prestacions d'aquesta.

1.3. Mòdul: Comunicacions

1.3.1. Descripció general

Aquest mòdul s'encarrega de gestionar l'enviament i la recepció dels missatges.

Consta de tres classes, dos d'elles hereten de la mateixa, SMSComm. Aquesta classe estableix els paràmetres necessaris per tal que es puguin enviar i rebre SMS. Com he comentat anteriorment, aquesta aplicació no utilitza ni el motor d'enviament de missatges SMS ni la gestió de contactes natius dels telèfons mòbils. Per poder aconseguir-ho, J2ME obliga a especificar un port en cadascuna de les aplicacions per tal de que puguin enviar i rebre missatges. Els missatges s'envien, doncs, no només a un número de telèfon (com és fa en seguint el procediment habitual) sinó també a un número de port.

Quan el terminal mòbil rep un missatge SMS en un port determinat el que fa és executar una classe que prèviament s'hagi predeterminat.

Les altres dues classes, que hereten de SMSComm són SMSReceiver i SMSSender.

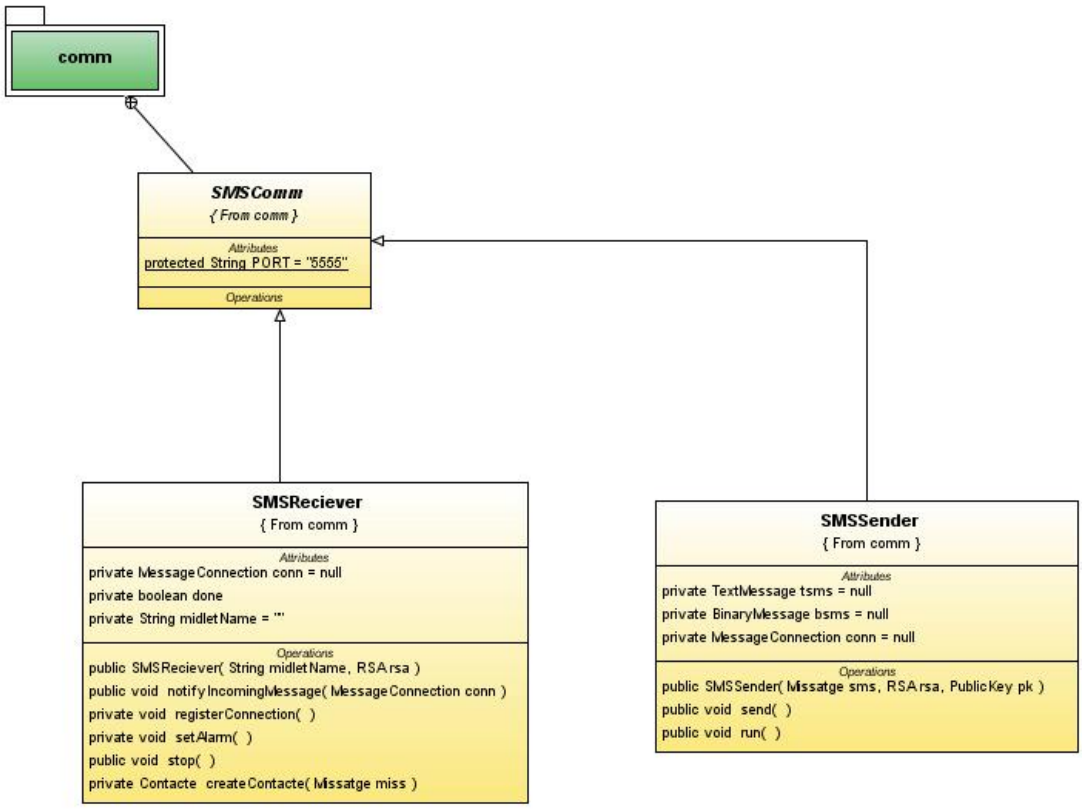
SMSReceiver gestiona la recepció dels missatges SMS, cridant les corresponents classes per guardar-lo en la base de dades. Si el missatge comença amb el literal "@CP@" el guarda en la taula de contactes, ja que és la marca literal que he escollit per marcar que el missatge és una clau pública que ens han enviat seguint l'opció corresponent del menú principal. Si per contra el missatge comença amb el literal "@NESMS@", l'aplicació li diu al midlet que és un missatge no encriptat i sense procesar-lo el copia a la safata d'entrada. En qualsevol altre cas, la classe suposa que es tracta d''un missatge encriptat i procedeix a processar-lo.

Com és pot suposar, aquesta classe és la que s'executa quan es rep un missatge pel port especificat en l'aplicació. Aquesta propietat d'executar una classe a partir de que hagi succeït un event s'anomena push registry.

SMSSender gestiona en canvi l'enviament dels missatges SMS.

Degut a que el contingut dels missatges encriptats pot contenir caràcters ASCII de control (no imprimibles en pantalla), l'enviament i la recepció dels missatges és farà en binari, pel que tant la classe SMSReceiver com la classe SMSSender faran la conversió després de rebre o abans d'enviar.

1.3.2. Diagrama de classes UML



1.4. Mòdul: Beans

1.4.1. Descripció general

Aquest mòdul l'utilitzo només per poder manipular les dades que utilitzo al llarg de l'aplicació. Això fa més senzilla la comprensió del codi al igual que la depuració d'aquest. Però sobretot em permet mantenir una independència total entre les dades que gestiona l'aplicació i el mitjà no volàtil en que es guarden. Per exemple, podem tenir la mateixa aplicació funcionant en dos entorns diferents, un d'ells guardant les dades en arxius XML i l'altre en una base de dades relacional AS400, només canviant un dels mòduls enlloc de canviar quasi tot el codi font.

És per això que les classes que composen aquest mòdul es corresponen amb les unitats de informació amb que treballa el programa.

La classe Contacte conté tots els atributs i mètodes que necessitem per manipular els contactes.

La classe Missatge conté tots els atributs i mètodes comuns que necessitem per representar un missatge malgrat que cal especificar el tipus de missatge mitjançant una de les següents classes que hereten d'aquesta.

MissatgeEliminat la utilitzaré per representar un missatge que ha estat eliminat.

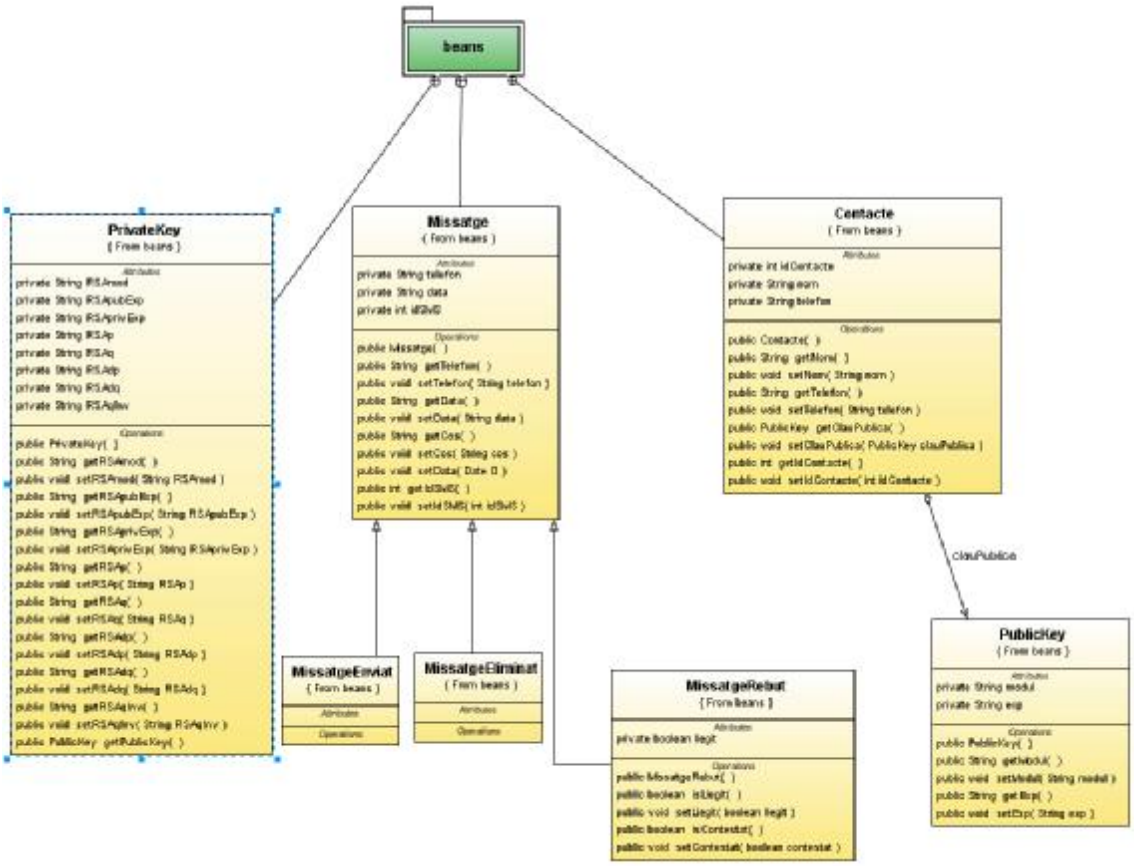
MissatgeEnviat la utilitzaré per representar un missatge que s'ha enviat.

MissatgeRebut la utilitzaré per representar un missatge que hem rebut.

PrivateKey és la classe que em permet guardar tots els paràmetres necessaris per calcular en qualsevol moment la clau privada que utilitzaré per encriptar els missatges sortints.

La classe PublicKey en canvi és la classe utilitzo per guardar la clau pública d'un contacte per tal de desencriptar un missatge.

1.4.2. Diagrama de classes UML



1.5. Mòdul: Criptografia

1.5.1. Descripció general

Aquest mòdul s'encarrega d'encriptar i desencriptar els missatges i de generar una clau publica i una clau privada en cas de que aquesta no existeixi.

Consta tan sols d'una classe, RSA, amb tots els atributs i mètodes necessaris per dur a terme les tasques descrites anteriorment.

Inicialment volia utilitzar el PGP / OpenPGP amb la idea de futur de utilitzar les mateixes claus creades amb el programa SendCrypt per encriptar els correus electrònics enlloc de tenir-ne dues de diferents.

Vaig buscar per Internet implementacions de l'algorisme OpenPGP per entorns J2ME, però no vaig trobar-ne cap. Vaig provar però, implementacions per Java convencional (per exemple PGPJava) i com era d'esperar no es podien importar a entorns de mobilitat. Al final però, vaig trobar una projecte similar a SendCrypt però desenvolupat per entorns Symbian que utilitzava una llibreria amb l'algorisme OpenPGP en entorns Java reduïts. Aquest projecte s'anomena mPGP i la llibreria que utilitza Cryptix.

Al final però no vaig aconseguir fer que cap implementació OpenPGP funcionés en l'entorn que estava treballant, el J2ME.

Deixant la idea inicial d'utilitzar OpenPGP, vaig mirar altres algorismes i seguint el consell del meu consultor, vaig escollir utilitzar un sistema d'encriptació RSA de clau pública. Per fer-ho, he optat per les llibreries open source BouncyCastle, que malgrat no ser les millors, si que són les úniques en que el seu ús no està limitat per cap patent comercial.

BouncyCastle ofereix API's d'encriptació de la majoria d'algorismes existents per totes les versions de llenguatge Java, entre d'elles J2ME.

Val a dir que no és ni de bon tros la més fàcil de treballar ja que per exemple no té un sistema d'exportar les claus d'una manera directa, sino que obliga a guardar cadascun dels paràmetres (mòdul, exponent públic, exponent privat, p, q, dp, dq i qinv) que generen aquesta clau per separat.

Un altre problema que he tingut amb aquesta llibreria és que utilitza classes del Java convencional (per exemple java.math.BigInteger) que no existeixen en la versió J2ME i que malgrat les han incorporat en el seu codi font, obliguen a ofuscar el codi per tal de que els hi canviïn el nom.

Ofuscar és un procés en que es manipula un programa escrit en Java per tal de que no se li pugui extreure el codi font un cop estigui compilat. Els programes

escrits en J2ME que utilitzin aquesta llibreria d'encriptació s'han d'ofuscar per tal de que les maquines virtuals Java dels terminals mòbils no rebutgin l'aplicació al entendre que utilitza classes només existents en el llenguatge Java convencional.

Ofuscar aquest mòdul va interferir en el funcionament de tots els altres mòduls fins que vaig adjuntar les següents linies com a parametres en el programa ofuscador:

```
-keep public class * extends javax.microedition.midlet.MIDlet

-keep public class crypt.* {
        public *;
}
-keep public class bbdd.* {
        public *;
}
-keep public class beans.* {
        public *;
}

-keep public class edu.upc.J2MESDLIB.* {
        public *;
}
-keep public class comm.* {
        public *;
}

-defaultpackage
-dontusemixedcaseclassnames
-libraryjars "C:\SendCrypt\src\lib\J2MEMicroDBv10.jar"
```

1.5.2. Diagrama de classes UML

1.6. Mòdul: BBDD

1.6.1. Descripció general

Per últim, aquest mòdul s'encarrega de guardar en un suport no volàtil tota la informació que genera l'aplicació en el seu funcionament habitual.

Aquest, ha estat potser el pitjor dels problemes als que m'he hagut d'enfrontar realitzant aquest projecte.

La idea inicial era guardar en simples arxius de text els missatges enviats, els missatges rebuts, els missatges eliminats, els contactes de l'agenda i la clau pública, és a dir, tenir cinc arxius de text, amb la informació gravada seqüencialment sense tenir possibilitat de fer recerques directes o ordenacions seguint criteris determinats.

Veient que aquesta idea no era tot lo adient que creia que havia de ser, vaig buscar de nou per Internet solucions alternatives, que em facilitessin un accés més precís i ràpid a la informació, i la vaig trobar.

XMLXML és una llibreria per utilitzar des de J2ME que permet manipular arxius XML. Els arxius XML, malgrat esser arxius de text, tenen una estructura que permet a uns algorismes determinats accedir ràpida i directa a la informació buscada.

Un cop estava tot el desenvolupament fet i verificat amb l'emulador del Netbeans, la meva sorpresa va ser majúscula quan vaig veure que malgrat tenir terminals mòbils que complien amb les especificacions J2ME per accedir al sistema de fitxers, aquest no es produïa correctament: en alguns terminals no es podia, en altres nomes es podia accedir a les targetes de memòria externes tenien insertades, d'altres accedien en modes equivocats d'escriptura i lectura.

De nou, estava en el punt de partida, pel que vaig buscar de nou alternatives.

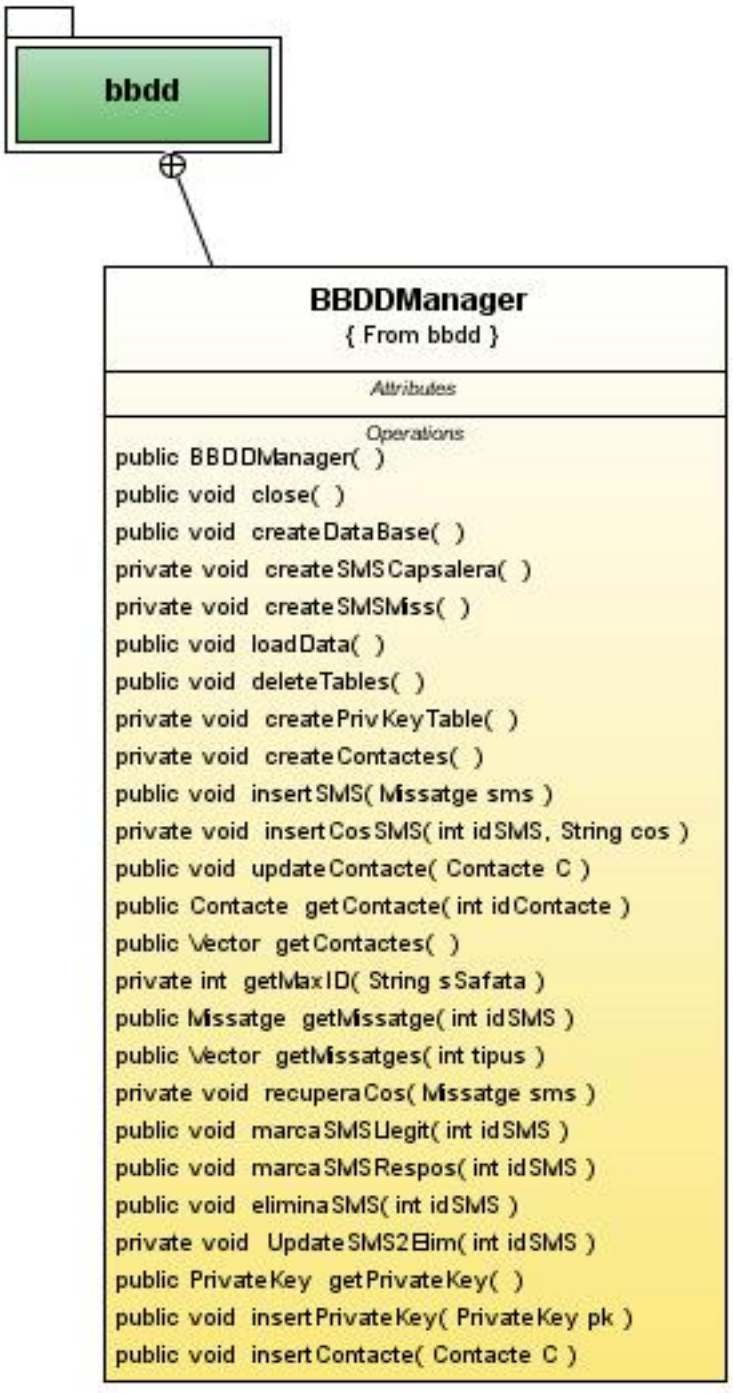
Aquest cop, l'alternativa va ser un desenvolupament de la UPC anomenat J2MicroDB. És un sistema gestor de base de dades per entorn J2ME que guarda la informació en els RMS (Record Management System) enlloc de fer-ho en arxius externs del sistema d'arxius del terminal mòbil.

J2MicroDB és un gran producte al que encara li resten hores de depuració, només permet un tipus de dades en els camps (String), amb una longitud màxima de 254 caràcters, a part, tampoc té controlat la gestió dels errors, però funciona perfectament i ara per ara és l'únic producte d'aquestes característiques (està a punt de sortir un altre producte OpenSource que també promet molt. S'anomena OpenBaseMovil).

La classe BBDDManager conté tots els atributs i registres necessaris per permetre el traspàs d'informació entre la aplicació i la base de dades.

L'estructura que he escollit per guardar la informació en cadascuna de les taules de la base de dades és la següent:

1.6.2. Diagrama de classes UML



1.7. Conclusions

Al llarg d'aquesta memòria he explicat totes les traves que m'he trobat alhora de desenvolupar aquest projecte. Un projecte que m'atrau i m'anima a continuar, que em motiva, però que si ara hagués de tornar a començar, potser el canviaria per no tornar a passar pel que he passat.

J2ME és encara un llenguatge massa jove i massa depenent dels fabricants de terminals mòbils, per l'experiència que he adquirit en aquest treball puc dir que si bé la filosofia de tenir un llenguatge únic per tots els terminals mòbils és fantàstica, actualment és utòpica.

Sinó, perquè després d'executar un mateix programa en tres terminals mòbils diferents, l'interficie apareix diferent? Perquè les opcions del menú principal i les funcions dels botons de control dret i esquerre que jo he programat d'una manera apareixen de diferent manera en els diferents terminals mòbils? Perquè un menú principal que té cinc opcions, amb un terminal Motorola en té sis? Perquè amb uns terminals un missatge es veu dins un requadre de diverses línies i en altres terminals la mateixa aplicació mostra els missatges en una línia amb un scroll cap a la dreta i cap a l'esquerra?

I no s'acaba tot amb la visualització. Perquè si en les especificacions del fabricant apareix que integra l'accés al sistema d'arxius javax.microedition.io.file (JSR75), aquest no es realitza correctament?

Però no s'acaben aquí els problemes, ja que encara no hem parlat de la seguretat. En tots els terminals, fins i tot en l'emulador, m'he trobat que cada vegada que vull accedir a algun recurs em demana permís (per exemple al intentar enviar un SMS) trencant totalment la fluïdesa d'execució de l'aplicació.

Aquest últim punt té una solució que passa per signar l'aplicació amb un certificat d'una entitat certificadora reconeguda en la maquina virtual J2ME del dispositiu mòbil. Per fer això, cal comprar un certificat digital amb una duració limitada. Una altre possibilitat és enviar el programa a http://javaverified.com per tal de que l'estudiïn i el signin de forma gratuïta, però amb la condició que l'aplicació resti disponible al públic.

Un altre punt important son els problemes que m'han aparegut intentant aplicar solucions provisionals a cadascun dels problemes detallats anteriorment. Quan ofuscava el codi per tal de que el sistema criptogràfic funcionés, deixava de funcionar la base de dades, i, quan deixava d'ofuscar el codi per tal de que funciones la base de dades, deixava de funcionar el sistema criptogràfic. Però, alhora de signar l'aplicació amb un certificat emes per VeriSign, deixaven de funcionar tant el sistema criptogràfic com l'accés a la base de dades.

I per últim, veure com després de que l'empresa on treballo comprès un certificat a VeriSign per signar l'aplicació, alguns terminals mòbils son incapaços

de llegir la firma correctament pel que segueix demanant confirmació per cada operació d'entrada i sortida que el programa necessita realitzar.

En resum, crec en aquest projecte i crec en tenir en un futur un portal unificat a Internet en que l'usuari podrà comunicar-se de manera privada, codificada i segura. Ara, he aconseguit un programa força estable, que compleix perfectament amb les funcions per les quals ha estat creat. No hi ha dubte que li manquen moltíssimes opcions per tal de facilitar la seva utilització i fer-lo mes atractiu a l'usuari però com he explicat en els punts anteriors, el llenguatge utilitzat no és el més amè per escriure facilitats i el temps per la realització del TFC és força limitat.

2. Glossari

A continuació relacionaré tot un seguit de termes que han aparegut al llarg de la memòria per tal de facilitar el seu seguiment i la seva comprensió.

Android: Sistema operatiu basat en Linux desenvolupar per Google per dispositius mòbils.

API (Application Programming Interface): Interficie de Programació d'Aplicacions. Conjunt de funcions, mètodes i procediments recollits en una llibreria que es pot utilitzar des de qualsevol aplicació.

Apple IPhone: Telèfon mòbil multimèdia desenvolupat per Apple que incorpora el sistema operatiu Mac OSX adaptat a processadors ARM.

ARM: Família de processadors RISC dissenyats per l'empresa Acorn Computers que s'utilitzen habitualment en dispositius mòbils (PDA, telèfons mòbils, ...).

BlackBerry: Terminals inalàmbrics desenvolupats per Research in Motion coneguts principalment per la integració del correu electrònic.

Bluetooth: Nom comú que rep la especificació IEEE 802.15.1 que defineix un estandard global de comunicacions per radiofrecuencia que permet la transmisio de dades i veu entre dispositius de forma segura.

ByteCode: Codi intermig més abstarcte que el codi màquina. Java utilitza aquest tipus de codificació per tal de poder executar el mateix programa en plataformes diferents.

GPS (Global Positioning System): Sistema de Posicionament Global. Sistema de navegació per satèl·lit que permet determinar la posició d'un objecte o persona amb pocs centímetres d'error.

IETF (Internet Engineering Task Force): Grup de Treball en Enginyeria d'Internet. Organització internacional oberta creada per la normalització l'any 1986 a Estats Units.

J2ME (Java 2 Micro Edition): Col·lecció d'APIs Java orientades a productes de consum (telèfons mòbils, PDA o electrodomèstics).

J2MEPolish: Colecció d'eines pel desenvolupament d'aplicacions J2ME

JAD (Java Application Descriptor): Arxiu de text que conté dades descriptives de l'aplicació per instal·lar el Midlet en el terminal mòbil.

JAR (Java ARchive): Arxiu binari que conté comprimides totes les classes Java que necessita el Midlet per poder-se executar junt amb d'altres arxius auxiliars com podrien ser els icones gràfics, els fons de pantalla o els arxius de configuració.

Java: Llenguatge de programació orientat a objectes desenvolupat per Sun Microsystems l'any 1990.

JSR (Java Specification Request): Conjunt de documents que descriuen les especificacions per implementar l'accés a una determinada tecnologia.

Midlet: Aplicació desenvolupada en J2ME per esser utilitzada en terminals mòbils. Consta normalment de dos fitxers, d'igual nom però d'extensions JAD i JAR.

MIDP: Versió de J2ME integrada en els telèfons mòbils. Hi ha tres versions MIDP1, MIDP2 i MIDP3.

MMS (Multimedia Messaging Service): Sistema de Missatgeria Multimèdia. Servei disponible en la telefonia mòbil que permet enviar missatges amb contingut multimèdia (sò, imatges i vídeos) entre terminals mòbils.

OMV: Operador de Mòbil Virtual. Operador de telefonia mòbil que no te xarxa ni cobertura pròpia sinó que la contracte a altres operadors que si que en tenen.

OpenPGP: Éstandard d'Internet basat el PGP proposat per la IETF.

Palm OS: Sistema operatiu desenvolupat per l'empresa PalmSource Inc. pels seus propis dispositius PDA.

PDA (Personal Digital Assistant): Assistent Digital Personal. Ordinador de mà dissenyat inicialment com a agenda electrònica. Com tot ordinador, funciona mitjançant un sistema operatiu que gestiona tots els components físics junt amb l'interacció de l'usuari.

PGP (Pretty Good Privacy): Privacitat bastant bona. Aplicació desenvolupada per Phil Zimmermann per tal d'encriptar la informació mitjançant algorismes d'encriptació de clau pública.

PushRegistry: Implementació que permet a un midlet executar-se automàticament sense l'intervenció de l'usuari, a traves d'un event determinat. Existeix només en MIDP2 i MIDP3.

RMS (Record Management System): Sistema de Gestió de Registres. Implementació d'un mecanisme basat en registres que perment guardar informació no volatil a les aplicacions desenvolupades en J2ME.

RSA: Algorisme asimètric de codificació que utilitza una clau pública i una clau privada.

SMS (Short Message Service): Servei de missatges curts. Servei disponible en la telefonia mòbil que permet enviar missatges curts de text entre terminals mòbils i actualment, entre telèfons fixes.

Symbian OS: Sistema operatiu per terminals mòbils creat mitjançant una aliança entre diverses empreses de telefonia mòbil.

UML (Unified Modeling Language): Llenguatge Unificat de Modelat. Llenguatge gràfic per visualitzar, especificar, construir i documentar una aplicació.

UMTS (Universal Mobile Telecommunications System): Sistema Universal de Telecomunicacions Mòbils. Tecnologia utilitzada pels mòbils de tercera generació que permet velocitats de fins a 2Mbps per cada usuari mòbil.

Verisign: Empresa reconeguda internacionalment al ser una Autoritat de Certificació, emisora de tot tipus de ceritficats SSL.

VeuIP: Utilització del protocol IP d'Internet per transmetre la veu.

Windows Mobile: Sistema operatiu desenvolupat per Microsoft per diversos terminals mòbils.

XML (Extended Markup Language): Llenguatge de marques extensible. es un metallenguatge extensible d'etiquetas que es proposa com un estàndard per l'intercanvi d'informació estructurada.

3. Bibliografia

3.1. Relació de pàgines web consultades

[1] http://java.sun.com
[2] http://www.javahispano.org/
[3] http://es.wikipedia.org
[4] http://www.j2mepolish.org/
[5] http://www.javaverified.com/
[6] http://www.verisign.com
[7] http://www.bitelia.com
[8] http://www.xatakamovil.com
[9] http://www.symbian.com/
[10] http://www.microsoft.com/spain/windowsmobile/default.mspx
[11] http://www.bouncycastle.org
[12] http://www.cryptix.org/
[13] http://sourceforge.net/projects/mpgp/
[14] http://www.openpgp.org/
[15] http://www.pgpi.org/
[16] http://morfeo.upc.es/crom/course/view.php?id=5
[17] http://kxml.sourceforge.net/
[18] http://www.codegear.com/products/jbuilder
[19] http://www.netbeans.org
[20] http://www.eclipse.org/
[21] http://msdn.microsoft.com/vjsharp/
[22] http://www.javaworld.com/javaworld/jw-04-2006/jw-0417-push.html
[23] http://www.it.uc3m.es/celeste/docencia/j2me/tutoriales/
[24] http://193.146.58.131/riicu/numeros/1/38-cicuSierraC.pdf
[25] http://developers.sun.com/mobility/midp/articles/pushreg/
[26] http://developers.sun.com/mobility/midp/questions/pushregistry/
[27] http://developers.sun.com/mobility/midp/articles/sms/
[28] http://www.lcc.uma.es/~galvez/J2ME.html
[29] http://www.quands.cat/wp/
[30] http://www.514.es/
[31] http://www.argo.es/~jcea/artic/cripto.htm
[32] http://www.segu-info.com.ar/criptologia/criptologia.htm
[33] http://www.kriptopolis.org/
[34] http://www.criptored.upm.es/
[35] http://www.iec.csic.es/criptonomicon/

3.2. Relació de llibres consultats

[36] GALVEZ ROJAS, SERGIO Y ORTEGA DIAZ, LUCAS (2003): *Java a tope: J2ME*. Universidad de Malaga
[37] KNUDSEN, JONATHANK Y LI, SING (2005): *Beginning J2ME: From Novice to Professional, Third Edition.* Apress
[38] DREAMTECH SOFTWARE INDIA, INC., TEAM (1999): *Cracking the code. Wireless programming with J2ME.* Hungry Minds
[39] MORRISON, MICHAEL (2001): *Sams Teach Yourself Wireless Java with J2ME in 21 Days.* Sams
[40] PIROUMIAN, VARTAN (2002): *Wireless J2ME Platform Programming.* Prentice Hall PTR

4. Annexos

4.1. Manual d'instal·lació

Hi ha dues maneres d'instal·lar l'aplicació en un dispositiu mòbil: mitjançant una connexió de dades amb l'ordinador o mitjançant la connexió GPRS / 3G del terminal mòbil.

4.1.1. Connexió de dades amb l'ordinador

Aquesta opció només es pot utilitzar en aquells terminals mòbils que no necessitin l'arxiu de descripcions del midlet (JAD), sinó que siguin capaços d'instal·lar i executar l'aplicació només amb el fitxer binari (JAR).

La connexió amb l'ordinador es pot realitzar de tres maneres diferents, mitjançant un cable específic de cada terminal, mitjançant una connexió inalàmbrica Bleutooth o mitjançant una connexió inalàmbrica amb infraroigs.

Dependrà del camí escollit la manera exacte d'enviar el fitxer del programa. Si tot ha funcionat correctament, quan el mòbil rebi el midlet ens preguntarà si el volem instal·lar, i un cop fet això, ens demanarà di el volem executar.

4.1.2. Connexió mitjançant GPRS / 3G

Aquesta opció és la mes recomanable i la més senzilla d'utilitzar, malgrat que suposa un cost addicional ja que genera tràfic de dades que el nostre proveïdor de telefonia mòbil ens facturarà. L'únic requisit és tenir el terminal correctament configurat per navegar per Internet.

El procés és més senzill que l'anterior, només cal que en el navegador del nostre mòbil hi posem la següent URL:

http://www.sendcrypt.es/SendCrypt.jad

Això farà que ens descarregui l'arxiu descriptor del midlet i tot seguit ens pregunti si volem descarregar i instal·lar l'arxiu binari del midlet (recordem que uns dels paràmetres de l'arxiu descriptor era la URL on podia descarregar l'arxiu binari).

4.2. Ús de l'aplicació

4.2.1. Consideracions inicials

L'aplicació genera automàticament la clau publica i privada el primer cop que s'executa en un terminal mòbil i les guarda en la base de dades pel que en properes execucions, ja no les crea sinó que simplement les recupera de nou.

L'agenda incorporada només pot tenir contactes amb clau publica, no he fet un manteniment complert d'aquesta taula que permetés afegir contactes sense aquesta.

L'aplicació permet enviar missatges encriptats i sense encriptar. Tots els missatges que s'enviïn utilitzant l'agenda s'encriptaran automàticament, per altre banda, els missatges que s'enviïn escrivint el numero directament s'enviaran sense encriptar.

Com he comentat al llarg d'aquest redactat, un dels problemes que més se m'han repetit durant aquest desenvolupament ha estat la compatibilitat amb els terminals mòbils amb que he treballat, pel que relaciono a continuació els terminals que executen l'aplicació i els que no.

Terminals que executen l'aplicació: Motorola V360, Mototola V3, HTC TyTN, HTC Trinity
Terminals que no executen l'aplicació: Nokia 6021

4.2.2. Operativa bàsica de gestió de una safata de missatges

A continuació incloc la operativa per tal de gestionar la safata de missatges explicant a cada pas el que vaig fent. He preferit prendre com a exemple la safata d'entrada, però podria utilitzar-se qualsevol de les altres dues de la mateixa manera:

Aquesta és la pantalla principal, on el mòbil ens pregunta si estem segurs que volem executar l'aplicació.

Un cop li hem dit que si que volem executar el midlet ens apareix la finestra principal.

Obrim el menú principal amb el botó d'accés directe de la dreta per tal de desplegar les opcions del menú.

Un cop fet això ens apareix la relació de misstges de la safata.

Si apretem el botó central sobre qualsevol d'ells se'ns obre una finestra amb el cos del missatge i la data.

Un cop aquí tenim dues opcions, respondre el missatge o eliminar-lo, i en el botó d'accés directe de l'esquerra, tenim l'opció de tornar enrere.

El missatge que hem respos apareix en la relació de missatges de la safata com a contestat.

Si en el menú anterior escollim l'opció d'eliminar el missatge ens apareix la relació de missatges actualitzada, en aquest cas amb tan sols un missatge.

I en la safata d'elements esborrats apareix el missatge que acabem d'eliminar.

4.2.3. Operativa bàsica de gestió de l'agenda

A continuació incloc la operativa que l'aplicació ens ofereix per treballar amb l'agenda:

Aquesta és la pantalla principal, on el mòbil ens pregunta si estem segurs que volem executar l'aplicació.

Un cop li hem dit que si que volem executar el midlet ens apareix la finestra principal.

Obrim el menú principal amb el botó d'accés directe de la dreta per tal de desplegar les opcions del menú.

Si escollim ara l'opció de l'agenda, automàticament ens apareixen relacionats els nostres contactes. Recordem que tots els contactes que apareixen aquí són perquè ens han enviat les seves corresponents claus públiques.

En aquesta ocasió només tenim dues opcions, tornar enrera mitjançant el botó d'accés directe de l'esquerra i enviar-li un sms al contacte seleccionat mitjançant el botó d'accés directe de la dreta.

4.2.4. Operativa bàsica de les opcions del menú principal

A continuació incloc la operativa que l'aplicació ens ofereix com a opcions del menú principal. Actualment, en aquesta versió nomes tenim una possibilitat que és la d'enviar la nostra clau publica al mòbil que nosaltres escollim:

Aquesta és la pantalla principal, on el mòbil ens pregunta si estem segurs que volem executar l'aplicació.

Un cop li hem dit que si que volem executar el midlet ens apareix la finestra principal.

Obrim el menú principal amb el botó d'accés directe de la dreta per tal de desplegar les opcions del menú.

Ens apareix només una possibilitat, la d'enviar la nostra clau pública al número de mòbil que escollim.

Al seleccionar-la, ens apareix el formulari d'escriure els missatges amb la nostra clau pública el el cos del missatge i amb el cursor en el número de telèfon per tal que li introduïm el nombre.