XAT SEGUR

Autor: Josep Lluís Calabuig Font (ETIG)

Consultor: Antoni Martínez Ballesté

18 de Juny del 2004

“Aprendre sense pensar és inútil, pensar sense aprendre, perillós.”
Confuci (551 ANE – 479 ANE)

Índex

1. Continguts del Document

En aquesta memòria s'especifica tota la informació que s'ha cregut que seria rellevant per a l'enteniment del funcionament i objectius del Treball Final de Carrera al que es refereix. Així, trobem estructurat el document segons la següent forma:

Inicialment es fa una introducció a la missatgeria instantània, tenint en compte com funciona, a grans trets, i quins problemes té aquest servei. Aprofitant aquesta introducció també es comenten els objectius generals del projecte, que de forma resumida podríem dir que són el desenvolupament d'una aplicació de missatgeria instantània deslligada de cap servidor i amb sistema de xifrat de la informació i negociació de claus.

Com que el projecte es centra molt intensament en els aspectes de seguretat, s'ha cregut oportú incloure també un apartat on s'expliquen de forma molt bàsica alguns conceptes de seguretat, així com els algorismes relacionats amb seguretat que farà servir l'aplicació desenvolupada.

En el següent punt es donen algunes especificacions generals de l'aplicació, així com els requeriments per al seu ús i altres aspectes de la implementació pràctica, com per exemple quin llenguatge de programació s'ha escollit i perquè s'ha escollit aquest i no un altre.

Posteriorment es comenta com funciona l'aplicació internament, és a dir, com està dissenyada i implementada, però sense arribar a mirar el codi font.

A continuació es presenta un manual de l'aplicació, en el qual s'expliquen totes les opcions que té l'aplicació i els passos a seguir per al seu bon ús, així com el procés a reproduir per a la seva instal·lació i inicialització.

Finalment, trobem un apartat de conclusions i de línies futures, que ens permetrà veure les possibilitats i problemes d'aquesta aplicació i les possibles millores a introduir en el futur.

2. Introducció

Dins el conjunt de les aplicacions de comunicacions que actualment podem trobar en el mon de la informàtica, existeix un elevat i variat nombre d'aplicacions de missatgeria instantània (entre d'altres productes).

Aquestes aplicacions de missatgeria instantània permeten a una gran quantitat d'usuaris de la xarxa Internet establir comunicacions en temps real entre ells, tradicionalment de forma escrita, però que en l'actualitat sovint també s'inclou la possibilitat de fer-ho per veu o fins i tot per videoconferència, així com altres serveis com els jocs on-line, recepció d'informació financera, compartició de fitxers, telefonia IP, etc.

2.1 Què és la missatgeria instantània

La missatgeria instantània és molt sovint definida com un punt intermig entre els programes de chat i els programes de correu electrònic. Aquesta manera de definir el que realment és actualment un programa de missatgeria instantània queda curt (ja que actualment aquestes aplicacions donen moltes altres facilitats o serveis), però sí que és cert que en recull l'aspecte fonamental i l'essència del que eren aquestes aplicacions en un principi.

Per al cas que ens pertoca ens és suficient de saber que la missatgeria instantània és un servei que proporcionen determinades aplicacions, i que permet, de forma més o menys adequada, la comunicació per escrit i en temps real de varis usuaris d'Internet (generalment entre 2 usuaris, però sovint també es permeten converses de varis usuaris, és a dir, reunions de grup).

En els últims anys aquests tipus de serveis, principalment popularitzats per aplicacions com l'ICQ, el Microsoft MSN Messenger, l'AOL Instant Messenger o el Yahoo Messenger, han arribat a quotes d'ús realment impressionants. Actualment moltes de les comunicacions personals o empresarials que s'estableixen diàriament es fonamenten en aquests serveis de missatgeria instantània, sent per a moltes persones de vital importància el seu ús i sovint una manera molt econòmica d'establir comunicacions de llarga distància.

2.2 Funcionament i problemes de la missatgeria instantània

Per norma general aquests tipus d'aplicacions funcionen en el ja molt conegut esquema client - servidor, on els usuaris, i mitjançant aquesta aplicació, es connecten a un servidor per tal de poder establir algun tipus de comunicació amb la resta d'usuaris.

Aquest tipus de funcionament fa que la càrrega recaigui principalment en el servidor, el qual ha de controlar tots els usuaris (clients) que té connectats, adonant-se sempre de si la comunicació es talla o bé si un nou usuari es connecta, així com gestiona quins usuaris veuen a quins altres usuaris, qui es comunica amb qui, etc.

El problema que representa aquest sistema és que els usuaris depenen del bon funcionament del servidor per a poder utilitzar el servei. És a dir, la caiguda del servidor o el mal funcionament d'aquest (sigui accidental, per atacs de Denegació de Servei, per excés de tràfic, per problemes de xarxa, etc.) provocarà que l'usuari rebi un servei deteriorat o que simplement no tingui accés a dit servei.

Per altra banda, també cal tenir en compte l'aspecte de la confidencialitat de les comunicacions i la autenticitat de les fonts, que sovint amb aquests serveis centralitzats en un servidor s'escapa del nostre control. Per exemple, si entenem confidencialitat com el fet que ningú excepte els interlocutors pugui saber què s'està dient en la conversa, aleshores aquesta no es pot assegurar amb un servei centralitzat, ja que amb un servei centralitzat en un servidor mai podrem estar segurs de que això sigui així, ja que el servidor fa d'intermediari i de negociador en l'establiment de la comunicació (en canvi amb un sistema Peer-to-Peer, punt a punt, serà més fàcil assegurar aquesta confidencialitat).

Un altre problema que hi ha en els serveis de missatgeria instantània és la falta d'un estàndard, doncs cada aplicació utilitza un sistema propietari propi i incompatible amb el de la resta d'aplicacions. Actualment però, estan començant a aparèixer aplicacions, generalment GNU, que permeten concentrar en una única aplicació tots els contactes dels diferents programes de missatgeria que poguéssim estar utilitzant (s'entén per *tots* com els més populars) i establir converses amb tots ells simultàniament, així com reunir-los en les mateixes converses de grup.

2.3 Objectius del Treball Final de Carrera

L'objectiu d'aquest Treball Final de Carrera és crear un programari de missatgeria instantània deslligada de cap servidor, de manera que cada usuari disposi d'un programa client/servidor amb el qual pugui rebre "trucades" o realitzar "trucades", en aquest últim cas indicant prèviament l'adreça IP del destinatari.

Quan es diu "una trucada", s'entén com a que una trucada és un flux continu d'intercanvi de missatges de text entre els diferents interlocutors (que escriuran a través del seu teclat i veuran la recepció d'aquests missatges a través de la pantalla).

És a dir, volem aconseguir una aplicació punt a punt de missatgeria instantània, la qual no dependrà ni farà ús de cap servidor intermig, alhora que s'implementarà un sistema de xifrat i negociació de claus que garantirà la confidencialitat de les converses.

Així, l'usuari del programari haurà de poder mantenir vàries "trucades" simultànies amb diferents usuaris, alhora que les comunicacions hauran d'ésser xifrades mitjançant una clau de sessió (diferent per a cada "trucada") que s'assignarà al iniciar la comunicació.

Evidentment, el programari disposarà d'una interfície gràfica adequada que facilitarà l'ús de l'aplicació a l'usuari i la gestió eficient de les diferents "trucades" simultànies que pugui establir. Així com també disposarà d'algunes opcions extres que milloraran la funcionalitat de l'aplicació (com per exemple una agenda).

3. Seguretat

En aquest apartat ens centrarem en la seguretat en les comunicacions a través de xarxes, consistents en prevenir, impedir, detectar i corregir violacions de seguretat durant la transmissió d'informació. No es tractarà la part de seguretat orientada a les pròpies màquines, és a dir, a la seguretat de sistemes operatius i bases de dades. A més a més, tindrem en compte que la seguretat es realitzarà per mitjans lògics i no pas físics. En concret ens centrarem en la seguretat lligada d'alguna manera amb l'aplicació de missatgeria instantània del projecte, però la gran majoria de conceptes seran aplicables a altres sistemes de transmissió de dades.

Cal tenir en compte que quan no es vol que hi hagi accés a certa informació o a certes transmissions per part de qui no hi estigui autoritzat, hi ha dos possibilitats bàsiques: O s'impedeix l'accés a la xarxa als estranys o s'utilitzen sistemes de xifrat de la informació de manera que es pugui enviar per entorns potencialment no segurs. La solució més segura sempre passa per combinar les dues opcions, però no sempre és possible. En el cas que ens pertoca només podrem fer ús de la segona opció, el xifrat de la informació, per tan, ens centrarem en aquest aspecte.

3.1 Conceptes de criptografia

La criptografia és l'art o la ciència encarregada de donar una capa de seguretat a la informació, és a dir, transforma les dades (encriptació o xifrat) de manera que siguin intel·ligibles per a usuaris no autoritzats, i generalment (però no sempre), dona mecanismes per a fer aquesta transformació inversa (desencriptació o desxifrat) a partir d'una o vàries claus només conegudes pels usuaris autoritzats.

Si considerem que tota informació es pot interpretar com cadenes de bits que es poden emmagatzemar o enviar, llavors també podem considerar que aquestes cadenes de bits no són més que una representació binària d'un gran nombres (o de varis nombres). A aquest nombre hi podem aplicar una funció matemàtica de manera que el resultat sigui un nombre diferent, una cadena de bits diferent i per tant una informació diferent, o el que és el mateix, la informació la podem canviar, la podem xifrar.

Actualment s'utilitzen de forma generalitzada dos tipus de xifrats: el xifrat per clau simètrica i el xifrat per parelles de claus pública i privada (també conegut com xifrat per clau pública).

Si anomenem **x** al nombre que representa la informació, **f(·)** la funció matemàtica que farà la transformació i **k** la clau a utilitzar en el xifrat (que de fet serà un altre nombre), llavors al fer **f(x, k)**, tal i com es mostra a la figura 3.1, obtindrem un altre nombre resultat, **y**, que serà la informació xifrada mitjançant la clau **k**. Si s'ha escollit una **f(·)** que sigui difícil d'invertir, llavors serà molt complicat obtenir **x** a partir de **y** si no es coneix **k**. Si aconseguim trobar un sistema per donar **k** als usuaris autoritzats, llavors ja tenim una manera vàlida de xifrar de forma segura la informació.

Quan la clau **k** utilitzada per al xifrat i per al desxifrat és la mateixa, es diu que l'algorisme criptogràfic utilitzat és simètric. Aplicant-ho a l'exemple anterior, tindrem que la font enviarà la informació **x** xifrada per **k** mitjançant **f(·)**, mentre que el receptor agafarà la informació rebuda **y** i la desxifrarà utilitzant **k** i una funció **g(·)** determinada que li donarà com a resultat la informació que es tenia abans del xifrat, és a dir, **x**.

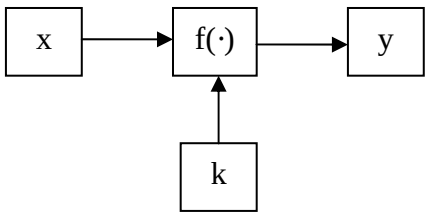

Figura 3.1 – Esquema bàsic de xifrat –

Com ja s'ha dit, existeix un altre tipus d'algorisme de xifrat, el xifrat per clau pública. Aquest es basa en parelles de claus i amb una funció que compleix unes certes propietats. Si les claus fossin **A** i **B**, la funció **f(·)** hauria de complir:

$$f(f(text, A), B) = text$$
$$f(f(text, A), A) \neq text$$
$$f(f(text, B), A) = text$$
$$f(f(text, B), B) \neq text$$

Si es trien les claus de manera que no es pugui deduir **A** de **B** ni a la inversa, llavors fent pública una clau (coneguda per tothom) i privada l'altra, es poden establir uns mecanismes que permetin aplicar aquest sistema a aspectes com l'autentificació, la integritat i la confidencialitat (Més endavant es veurà com funciona un d'aquests sistemes, l'algorisme de Diffie-Hellman).

El problema que té aquest sistema de xifrat per clau pública és que el xifrat és molt lent en comparació amb el xifrat per clau simètrica.

3.2 Criptografia aplicada a aquest TFC

En l'aplicació implementada en aquest Treball Final de Carrera s'ha fet ús de dos algorismes de xifrat. Per una banda tenim l'algorisme de xifrat simètric DES, i per altra banda tenim l'algorisme per clau pública de negociació de claus Diffie-Hellman.

El sistema DES és un dels estàndards d'algorisme de xifrat per clau simètrica més utilitzats en l'actualitat, el qual es realitza mitjançant desplaçament de bits i xors, per tan, permet fer una implementació per hardware o software molt ràpida i eficient. Típicament les claus són de 64 bits i amb una longitud efectiva de 56 bits, un valor potser massa petit per als nivells de seguretat exigibles avui en dia (problema que es resol mitjançant l'aplicació de la variant de l'algorisme anomenada triple DES), tot i que per al nostre cas serà suficient.

Aquest algorisme, força ràpid en el xifrat, serà utilitzat per a xifrar els missatges que s'enviaran els interlocutors. Tot i així ens cal un sistema per posar d'acord als dos interlocutors sobre quina clau ha de fer servir aquest algorisme DES. Amb aquest propòsit es farà ús prèviament de l'algorisme Diffie-Hellman, el qual permetrà a ambdues parts posar-se d'acord, i de forma segura, en l'ús d'una mateixa clau per al xifrat DES (es diu que l'algorisme Diffie-Hellman és un algorisme de negociació de claus).

Així doncs, per al nostre cas, l'algorisme Diffie-Hellman treballarà sobre dos usuaris **U1** i **U2**, els quals tindran cada un una clau pública i una clau privada, **U1pub**, **U1priv**, **U2pub**, **U2priv**. A partir de les clau públiques l'usuari **U1** calcularà:

U1tx = (U2pub ^ U1priv) % U1pub

i li enviarà el resultat a **U2**, al mateix temps, l'usuari **U2** calcularà:

U2tx = (U2pub ^ U2priv) % U1pub

i li enviarà el resultat a **U1**.

Així, els usuaris **U1** i **U2** podran calcular una clau coneguda per ambdós mitjançant els càlculs:

U1clau = (U2tx ^ U1priv) % U1pub
U2clau = (U1tx ^ U2priv) % U1pub

donat que **U1clau** = **U2clau** = clau coneguda pels dos = **Uclau**

[Nota: **%** és la funció també coneguda com **mod**.]

A partir d'aquesta clau **Uclau** coneguda per ambdues parts s'aplicarà una funció per obtenir una clau DES comuna (també s'hagués pogut fer que una de les parts escollís la clau DES i la envies xifrada amb **Uclau** a l'altra part, però no s'ha fet ja que s'ha considerat més segura la opció ja comentada).

4. Aspectes de la implementació pràctica

Cal tenir en compte que l'aplicació que es vol dissenyar i implementar és una aplicació de servei a l'usuari final (que no té perquè ser un professional de la informàtica), concretament per a connectar diferents usuaris, per tan, alhora d'implementar l'aplicació s'haurà de tenir en compte que aquesta pugui disposar d'una interfície gràfica senzilla i fàcil d'utilitzar.

Evidentment l'aplicació es fonamenta en les comunicacions via Internet, per tan haurà de realitzar-se tenint en compte que ha de fer ús dels protocols TCP/IP i no pas de cap altre protocol de comunicacions.

Un aspecte que s'ha remarcat molt en el plantejament d'aquest projecte és en el de la seguretat, per tan, l'aplicació haurà d'utilitzar mètodes de xifrat i negociació de claus ja consolidats, testats i validats per la comunitat internacional.

També caldrà tenir en compte que l'aplicació haurà de permetre vàries comunicacions simultànies, per tan caldrà fer ús de tècniques que permetin aquest aspecte.

Un aspecte desitjat també seria que l'aplicació fos compatible amb vàries plataformes, donat que l'usuari final pugui comunicar-se amb altres usuaris indistintament de la plataforma sobre la que estiguin treballant les seves màquines.

4.1 Llenguatge de programació Java de Sun Microsystems

La implementació pràctica del projecte s'ha fet íntegrament en llenguatge Java, concretament s'ha treballat sobre la Java 2 SDK (Software Development Kit) de Sun Microsystems, en la seva versió 1.4.1 i 1.4.2 indistintament.

El fet d'escollir el llenguatge Java com a eina de programació no ha estat un mer formalisme, sinó que s'ha tingut en comptes totes les avantatges i inconvenients que oferia enfront d'altres llenguatges i posteriorment s'ha vist que justament el llenguatge Java era el millor candidat per a la implementació del projecte.

Els aspectes bàsics que han motivat l'ús de la plataforma de programació Java es poden resumir en dos: Java és un llenguatge Orientat a Objectes pur molt potent; i Java és un llenguatge Multiplataforma.

Però l'ús de Java no ha estat només motivat per ser un llenguatge Orientat a Objectes i ser multiplataforma, sinó que altres aspectes han estat de vital rellevància. Alguns d'aquests aspectes són el fet que Java és robust, té dispositius de tractament d'errors, té una gestió automàtica de la memòria, permet l'ús de threads, incorpora llibreries de seguretat (xifrat DES, etc.), té llibreries per a l'ús de sockets, té llibreries per a la creació d'interfícies gràfiques de forma còmoda, etc.

Les característiques generals de Java es presenten en el següent punt.

4.1.1 Característiques del llenguatge Java

Les principals característiques del llenguatge Java són:

Simple

Java és un llenguatge senzill i entenedor que ens permet treballar sense l'ús d'encapçalaments de fitxers, aritmètica de punters, estructures, unions, sobrecàrrega d'operadors, classes base particulars, etc.

A més a més, el llenguatge Java és petit, és a dir, el programari creat ocupa poc espai de memòria.

Orientat a Objectes

Java és un llenguatge Orientat a Objectes (OO) pur, és a dir, a diferència d'altres llenguatges com el C++, es treballa només amb classes (a excepció dels elements bàsics com són els int, double, etc.).

Un altra diferència respecte d'altres llenguatges OO com el C++ és que no permet herència múltiple directa, fet que es soluciona mitjançant l'ús d'interfícies. El resultat final és un sistema d'herència múltiple més eficient per al compilador i més robust.

Distribuït

Java disposa d'una extensa llibreria d'objectes als que es pot accedir amb els protocols TCP/IP, HTTP o FTP. Permet que aplicacions Java puguin obrir i accedir a objectes remots, com si fossin locals, a través de la xarxa mitjançant una URL (Unified Resource Locator).

Robust

El llenguatge Java disposa d'un gran assortit de mètodes per a la comprovació en temps de compilació i en temps d'execució (comprovació dinàmica) de possibles problemes o errors, així com l'eliminació d'aquests. A més a més, Java disposa d'un model de gestió de memòria que elimina la possibilitat de sobreescriure memòria i que les dades es corrompin.

Segur

Degut a que Java està pensat per permetre treballar en entorns de xarxa o distribuïts, un factor que s'ha vigilat força és el de la seguretat. Java permet treballar amb criptografia com a sistema de seguretat i fins i tot permet establir polítiques de seguretat en l'execució d'una aplicació.

Arquitectura neutra o Multiplataforma

El compilador Java genera instruccions en codi byte (bytecode) que no tenen res a veure amb l'arquitectura de cap sistema en particular, però que són fàcils d'interpretar per qualsevol màquina i fàcils de traduir al codi concret de la màquina.

El funcionament bàsic és crear una JVM (Java Virtual Machine) en la màquina on s'executa el codi compilat. Aquesta JVM és un programa que actua com a traductor del codi byte d'arquitectura neutra al codi utilitzat per la màquina en la que s'executa. Així el que es fa és una compilació inicial a codi neutre (codi byte) i posteriorment i en temps d'execució es fa una traducció o segona compilació a codi de la màquina nadiua. Amb això s'aconsegueix tenir un codi que és multiplataforma, és a dir, independent de la plataforma hardware a utilitzar.

L'inconvenient d'això és la reducció de velocitat en l'execució del programa, donat que a més d'executar les instruccions, prèviament s'han d'haver traduït per la JVM. Afortunadament, amb les noves versions de Java aquests temps s'han anat reduint gràcies a millores incorporades. A més a més, la traducció no és exactament sempre en temps d'execució, la JVM compila tot el codi que no sigui dinàmic prèviament a l'execució i només compila codi quan aquest és dinàmic o impossible de compilar amb anterioritat a l'execució.

Portable

A diferència d'altres llenguatges com el C o el C++, les implementacions Java no són dependents dels aspectes de l'especificació. S'especifiquen les mides dels tipus de dades primitius i el comportament de l'aritmètica d'aquestes dades, mentre que les llibreries del sistema ja defineixen les interfícies portables.

Interpretat

Com ja s'ha comentat, l'intèrpret de Java (JVM) pot executar codis binaris directament en qualsevol màquina.

Alt rendiment

Si amb el rendiment de l'execució del codi byte interpretat no n'hi ha prou, Java permet fer una compilació al codi màquina d'una plataforma hardware concreta, guanyant així velocitat d'execució. Amb això però es perd la propietat de Multiplataforma.

Fil múltiple o Multithread

El fil múltiple o multithread dins un mateix procés aporta millor comportament a temps real i una major sensibilitat o grau de reacció.

Dinàmic

Java permet afegir nous mètodes i atributs d'instància a les llibreries sense cap efecte en els clients, sent així un llenguatge més dinàmic en segons quines modalitats que la gran majoria de llenguatges.

4.1.2 Versió del llenguatge Java utilitzada i requeriments del sistema

El paquet JDK (Java Development Kit) o SDK (Software Development Kit), que ofereix gratuïtament Sun Microsystems per al desenvolupament d'aplicacions Java, inclou eines que permeten compilar, depurar, generar documentació i interpretar codi escrit en Java.

Històricament les versions que han marcat canvis importants en el JDK o SDK són:

- JDK 1.0: primera versió
- JDK 1.1: canvis importants en el model d'esdeveniments, events.
- JDK 1.2: també coneguda com Java 2 SDK, es considera la versió definitiva que dóna maduresa al llenguatge, incorpora canvis com: incorporació dels paquets gràfics AWT, Java2D i Swing; millora de la gestió de seguretat; optimització dels fitxers JAR per a comprimir classes; augment de la funcionalitat i suport a la creació de components Java, JavaBeans; optimització de l'ús de números de coma flotant; etc.

Així i tot, han aparegut altres noves versions, sense canvis tan importants, com són la 1.3 i la 1.4 (o la que encara està en elaboració 1.5).

Per al cas que ens interessa, s'han utilitzat les versions SDK 1.4.1 i SDK 1.4.2, les quals ja porten incorporades les llibreries de seguretat i criptografia necessàries. Tot i així, es podria fer córrer l'aplicació utilitzant una versió anterior del SDK (1.2 o 1.3), però tenint en compte que seria necessari adjuntar aquestes llibreries (conegudes com JCE).

A més a més, qualsevol de les versions a partir de la 1.2, haurien de ser compatibles amb l'aplicació, ja que totes elles incorporen l'ús de les classes Thread, Swing i totes les referides a les comunicacions via Socket. De totes maneres no es garanteix el bon funcionament si s'utilitza la versió 1.2, ja que l'aplicació no ha estat testada en aquesta versió del llenguatge.

Pel que fa al sistema necessari per al funcionament de l'aplicació, cal dir que els requeriments són mínims.

L'únic que cal és tenir instal·lat un sistema operatiu compatible amb Java (família Windows, família UNIX, família Linux, etc.) que disposi d'entorn gràfic (per exemple KDE, GNOME, X-WINDOWS, etc.). Així com cal tenir instal·lat el JDK o el SDK (o altres compatibles), a partir de la versió 1.4 o bé la 1.3 amb les llibreries JCE (la versió 1.2 també hauria de funcionar però no ha estat testada).

Es recomana que es faci servir la versió 1.4.2 del SDK (o la 1.4.1), ja que versions com la 1.3 amb les llibreries JCE donen problemes amb alguns sistemes operatius (com per exemple Windows XP).

4.2 Java en l'aplicació

Un dels aspectes més importants del Java és el fet que és de plataforma neutra o multiplataforma. Això implica que al fer l'aplicació en aquest llenguatge puguem exportar l'aplicació a múltiples plataformes com Unix, Windows, Linux, Macintosh, etc. Permetent així que els usuaris puguin comunicar-se entre ells de forma transparent a la plataforma que tinguin la resta d'usuaris.

Un altre aspecte important del Java és que està molt orientat a les aplicacions basades en Internet, implementant classes específiques per a les comunicacions TCP/IP via sockets, les quals han estat aprofitades per implementar la part de comunicacions.

També cal remarcar la possibilitat de l'ús de la tècnica de programació per fils múltiples o multithreads, que ens ha permès tenir una aplicació que pot donar suport a vàries converses simultànies en finestres diferents sense perdre la sensació de funcionar tot en temps real.

No ens podem deixar tampoc l'ús de les classes de criptografia i seguretat de Java, les quals ja incorporen mètodes específics per a implementar tot el xifrat DES o la negociació de claus Diffie-Hellman.

Pel que fa a l'aspecte gràfic Java proporciona vàries possibilitats, de manera que s'ha escollit l'opció més senzilla i fàcil d'implementar per als requisits de la nostra aplicació, és a dir, s'ha fet ús de la llibreria de classes Swing, la qual permet la fàcil implementació de finestres, menús, botons, etc.

També s'ha aprofitat el fet que Java és capaç de capturar events i programar la resposta a aquests, possibilitant que s'hagi pogut controlar l'ús de l'aplicació que fa l'usuari a través del ratolí i en alguns casos a través del teclat.

Evidentment, la possibilitat que ens proporciona Java per a la captura i tractament d'Excepcions ha estat aprofitada per al control d'errors.

5. Disseny i implementació

Tot i que seria interessant utilitzar l'estàndard UML per a definir el disseny de l'aplicació, la dimensió tan petita de l'aplicació, així com la seva senzillesa, fa que s'hagi optat per utilitzar un sistema més senzill i entenedor de mostrar el seu disseny.

La part d'implementació es descriurà paral·lelament a la de disseny, ja que tot i que sovint s'aconsella separar dites parts, no ha semblat necessari en aquest cas i dona la sensació que tot és molt més entenedor si s'ajunta.

5.1 Estructura Estàtica

L'estructura estàtica bàsica de l'aplicació és com la que es mostra en la figura 5.1:

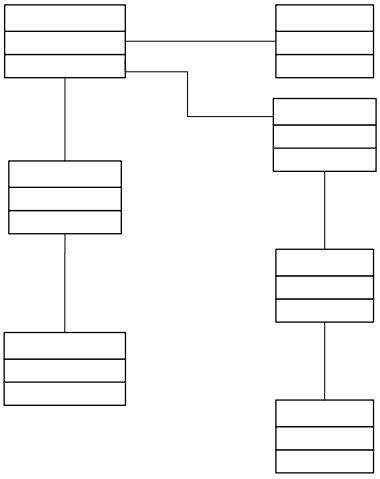

Figura 5.1 – Estructura Estàtica –

La classe *PantallaPrincipal* és la classe principal de l'aplicació, la qual inicia un thread d'una instància de la classe *tps* (encarregada de l'escolta i gestió de peticions de trucada entrants).

La *PantallaAgenda* és una finestra que apareix quan l'usuari seleccionar l'opció de l'agenda. Des d'aquesta pantalla es podran realitzar vàries accions, com crear nous contactes, eliminar-ne d'existents, etc.

Des de la *PantallaAgenda*, quan l'usuari vulgui editar un contacte o crear-ne un de nou se'ns obrirà una nova finestra, *PantallaContacte*.

Des de la *PantallaPrincipal* es podran instanciar fins a un màxim de 10 instàncies de *PantallaDialeg*, les quals representaran la interfície gràfica utilitzada per a què l'usuari pugui parlar amb el seu interlocutor (se suposa que cada *PantallaDialeg* és per a una conversa amb un usuari diferent, tot i que no hi ha cap impediment en que un usuari tingui vàries *PantallaDialeg* amb el mateix interlocutor).

La instanciació de *PantallaDialeg* corre a càrrec de *PantallaPrincipal*, com ja s'ha dit, ara bé, aquesta instanciació pot venir per ordre de l'usuari el qual inicia una trucada, o bé a partir del procés thread *tps*, el qual quan rebi una petició de trucada entrant gestionarà bona part del procés, però acabarà delegant la connexió en una *PantallaDialeg* (aquest procés de delegació recau en *PantallaPrincipal*).

Sempre que s'iniciï una instància de *PantallaDialeg* es crearà un thread des d'aquesta cap a una instància de la classe *tpc* (encarregada de la gestió de les converses i la negociació de claus, i en el cas de trucades fetes a petició de l'usuari també de la gestió de la petició de la trucada).

La classe *tpc*, un cop finalitzat el procés de negociació de claus, utilitzarà una instància de la classe *xifratDES2* per a utilitzar-la en el xifrat i desxifrat dels textos a enviar i rebre.

5.2 Utilitat de cada classe

▪ PantallaPrincipal

PantallaPrincipal és la classe principal de l'aplicació, declarada com a *static* per a què sigui única i referenciable des de qualsevol altra classe de l'aplicació. Això farà que només es pugui tenir una sola instància de l'aplicació corrent simultàniament en la màquina (en segons quins sistemes operatius si que es podria fer que hi haguessin varies instàncies, però per norma general no es podrà).

Només iniciar-se, aquesta instància de la classe principal obre un fitxer de configuració (*conf.xat*) que llegeix alguns paràmetres guardats en la sessió anterior (o el crea si el fitxer de configuració no existia). A continuació crea un thread amb una instància de la classe *tps*, la qual s'encarregarà d'obrir i escoltar el port de recepció de peticions de trucada (aquest port està definit en el fitxer de configuració i es pot canviar).

A partir d'aquí l'aplicació queda en espera d'algun event. És a dir, s'espera ha que arribi una petició de trucada entrant o bé que l'usuari realitzi alguna acció.

Les diferents opcions que pot escollir l'usuari en aquesta finestra proporcionada per *PantallaPrincipal* són:

- Canviar el nom d'usuari (obre una finestra sol·licitant el nou nom d'usuari).
- Canviar el port de recepció de trucades entrants (obre una finestra sol·licitant el nou port de recepció de trucades entrants).
- Tancar l'aplicació.
- Veure l'ajuda (mostra una finestra amb informació).
- Veure informació sobre l'autor i versió de l'aplicació (mostra una finestra amb informació).
- Iniciar una trucada sortint.
- Utilitzar l'agenda de contactes
- Canviar l'opció d'acceptació de trucades automàtiques de cert a fals i viceversa.

▪ tps

La classe *tps* és l'encarregada d'obrir un socket que actuï com a dimoni en el port de recepció de trucades entrants, i en el cas de que es rebi una petició de trucada entrant, gestionar tot el procés d'acceptació o refutació de la petició de trucada.

Quan es rep una petició de trucada entrant, la classe comprova si l'usuari té marcada l'opció d'acceptació de recepció de trucades automàtica. En cas afirmatiu crea un thread d'una instància (que queda referenciada a *PantallaPrincipal*) de la classe *PantallaDialeg*. En cas negatiu pregunta a l'usuari si vol acceptar la trucada, si l'usuari diu que sí actua tal com s'ha comentat, sinó simplement s'informa a l'altre interlocutor que no s'accepta la trucada i es tanca la comunicació. En qualsevol cas, quan aquest procés acaba, es torna a posar a escoltar en el port de recepció de peticions de trucada.

- PantallaDialeg

Aquesta classe és la que fa d'interfície gràfica per a la conversa entre l'usuari i l'interlocutor. Depèn directament de la instància de la classe *tpc* que genera la classe *tps* quan s'accepta una petició de trucada entrant, o bé que genera la classe *PantallaPrincipal* quan l'usuari sol·licita iniciar una trucada sortint.

La classe mostra, en una part de la pantalla amb Scroll de desplaçament, la conversa que es va produint entre ambdós interlocutors, així com disposa d'un casella on escriure el text que es vol enviar a l'altre interlocutor.

- tpc

La classe *tpc* s'encarrega de gestionar la negociació de claus, així com s'encarrega de la recepció de missatges de l'altre interlocutor (desxifrant-los i dipositant-los a la finestra de diàleg) i l'enviament de missatges també cap a l'altre interlocutor (xifrant-los i fent-los constar en la finestra de diàleg). Aquesta classe també s'encarrega de detectar si es produeix algun error en la comunicació o si aquesta es talla.

A part d'això, quan la instància de la classe *tpc* ha estat generada per una petició de trucada sortint, és a dir, una trucada sol·licitada per l'usuari, aleshores, prèviament a la negociació de claus, s'encarrega de tota la gestió per iniciar una petició de trucada cap al potencial interlocutor.

Per al xifrat DES dels textos enviats i el desxifrat DES dels textos rebuts, aquesta classe crea una instància de la classe *xifratDES2*, la qual permetrà fer tot el procés de xifrat i desxifrat.

- xifratDES2

Classe que permet xifrar i desxifrar textos que se li passen com a paràmetre a través de l'algorisme DES. La clau que utilitza se li pot passar com a paràmetre o pot ser generada de forma aleatòria.

- PantallaAgenda

Representa la interfície gràfica de l'agenda utilitzada per l'usuari per emmagatzemar o buscar adreces de contactes.

En aquesta agenda hi apareix una llista de contactes, que s'emmagatzema en un fitxer (*agenda.xat*). De cada contacte se'n pot veure el nom, IP i port que té assignat per a rebre peticions de trucades.

Des de aquesta finestra es podrà eliminar qualsevol contacte, crear-ne un de nou (obre una nova finestra), editar-ne un d'existent (obre una nova finestra), o bé, insertar l'adreça IP i el port en les caselles d'establiment de nova trucada de la pantalla principal.

- PantallaContacte

Mostra la interfície gràfica de la finestra que s'obre quan l'usuari sol·licita crear un nou contacte en l'agenda o bé editar-ne un d'existent.

Hi apareixen tres camps, el nom de l'usuari, la seva adreça IP i el port al qual cal enviar-li la petició de trucada. Aquests camps apareixeran buits en el cas de que s'estigui creant un nou contacte o bé amb les dades de l'usuari que s'estigui editant si s'està justament editant un contacte ja existent.

5.3 Finestres

En la creació de les finestres s'ha optat per dues opcions molt diferenciades.

Per una banda les finestres més importants han estat dissenyades i implementades heretant de les classes JFrame o JDialog segons el cas. Concretament es troben dins d'aquest cas les finestres implementades a partir de les classes PantallaPrincipal, PantallaDialeg, PantallaAgenda i PantallaContacte.

Per altra banda les finestres més senzilles han estat implementades amb un JOptionPane, que en permetia una implementació més senzilla i efectiva. Entre d'altres, trobem les següents finestres implementades amb aquest sistema: finestra de sol·licitud del nou nom d'usuari, finestra de sol·licitud del nou port de recepció de trucades entrants, finestra per veure informació d'ajuda, finestra per veure informació sobre l'autor, finestra de missatge d'error en format de dades, etc.

Per al cas de les finestres implementades a través de classes pròpies, dues hereten de JFrame, mentre que dues hereten de JDialog (per permetre obertura en mode modal). La seva representació es pot veure en la figura 5.2.

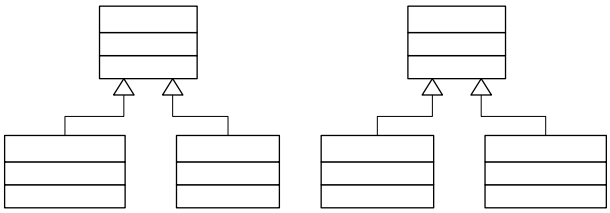

Figura 5.2 – Herència de les classes que implementen finestres –

5.4 Funcionament i estructura dinàmica

En aquest punt es pretén mostrar a grans trets el funcionament de l'aplicació, així com la manera en què es relacionen les diferents instàncies de classe de l'aplicació durant l'execució.

Com ja s'ha dit, la classe principal és la classe *PantallaPrincipal*, la qual està declarada com a *static* per tal de que la resta de instàncies hi puguin tenir accés fàcilment. Dins d'aquesta classe hi apareix un atribut en forma de vector, *vpd*, el qual s'encarrega d'emmagatzemar les referències a les instàncies de les diferents finestres de conversa (*PantallaDialeg*) que es puguin obrir (s'ha imposat un límit de 10 finestres, però s'hagués pogut posar sense límit i que es poguessin obrir tantes finestres com es volgués, però no s'ha cregut oportú).

Aquesta classe crea un thread cap a una instància de la classe t*ps*, la qual s'encarregarà de fer de dimoni, escoltant en el port de recepció de peticions de trucada i gestionant totes les possibles peticions que li arribin.

Aquesta classe t*ps* podrà crear una instància, en el moment que accepti peticions de trucada entrant, de la classe *PantallaDialeg*, la qual serà referenciada en l'atribut *vpd*. Tanmateix, la classe *PantallaPrincipal* també podrà crear una instància de la classe *PantallaDialeg* que també serà referenciada en l'atribut *vpd* (en aquest cas es realitzarà aquesta acció quan l'usuari sol·liciti iniciar una trucada sortint).

Així doncs, en el vector *vpd* hi apareixeran totes les referències a totes les finestres que hi pugui haver per a converses, és a dir, totes les instàncies de *PantallaDialeg* que hi hagi en aquell moment (fins a un màxim de 10), tan si s'han produït per trucades entrants com per trucades sortints.

També cal recordar que com ja s'havia dit, al obrir una instància de *PantallaDialeg*, aquesta crea un nou thread cap a una nova instància de la classe *tpc*. Aquesta instància de la classe *tpc* és la que completarà el procés d'establiment de la trucada, la negociació de claus i tot el procés d'enviament i recepció de missatges durant la conversa (inclòs el xifrat i desxifrat DES).

Així, quan un usuari iniciï una trucada utilitzarà una instància de la classe *tpc*, la qual es comunicarà amb una instància de la classe *tps* de l'altre interlocutor. Després d'unes certes negociacions, l'altre interlocutor tancarà la connexió establerta en la instància de la classe *tps* i obrirà una instància de la classe *tpc* perquè acabi la negociació de la connexió, així com la negociació de les claus amb la instància de la classe *tpc* de l'usuari que ha iniciat la trucada (ambdues *tpc* utilitzen un port prèviament acordat durant la negociació entre *tpc* i *tps*). Posteriorment a aquest procés, les dos instàncies de la classe *tpc* es comunicaran entre elles per a enviar i rebre els missatges de ambdós interlocutors, tot utilitzant el xifrat i desxifrat en l'enviament d'aquests missatges.

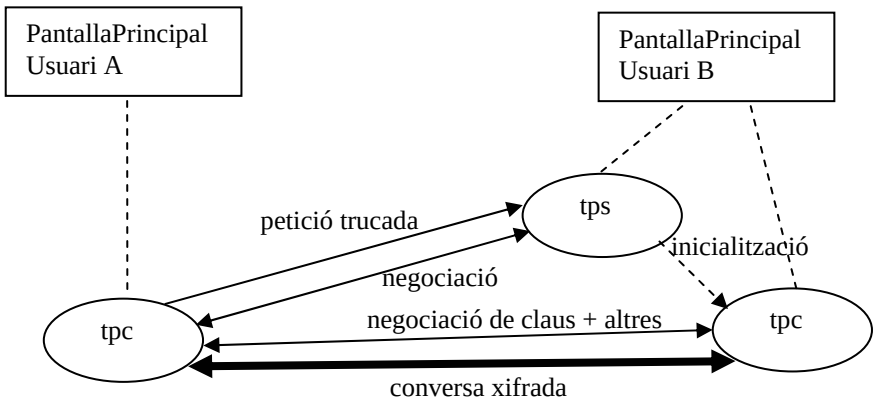

Figura 5.3 – Establiment de trucada –

Si partim de l'esquema mostrat en la figura 5.3, durant la negociació entre el *tpc* de l'usuari A i el *tps* de l'usuari B es realitzen els següents intercanvis d'informació (per a una trucada que s'estableix amb èxit):

tpc de l'usuari A	tps de l'usuari B	Descripció
	s'inicia el tps en el port portRT de B	Iniciem el dimoni de recepció de peticions de trucades de B en el portRT (per defecte el 1798).
envia "ruxat" a portRT de B		A pregunta a B si en el port hi ha el dimoni del xat (o aplicació de missatgeria instantània).
	envia "imxat"	B respon a A que SÍ és el dimoni de recepció de peticions de trucada del xat.
envia "asking for connection" a portRT de B		A sol·licita establir connexió amb B.
	envia "accepted"	B accepta la trucada amb un "accepted" (per refusar-la seria un "refused" i aquí s'acabaria la connexió).
envia "port" a portRT de B		A sol·licita el port en el que s'haurà de connectar amb el tpc de B.
	envia 2000+ID	B envia el número de port 2000+ID (van del 2000 al 2009).
tanca connexió a portRT de B	tanca connexió amb A i reinicia el dimoni en el port portRT de B.	

A partir d'aquest punt l'usuari B obre el seu *tpc* i l'usuari A connecta el seu *tpc* amb el de B pel port que ja s'havia negociat (el 2000+ID que indicava el *tps* de B). Així, l'intercanvi d'informació continuaria com s'indica a continuació:

tpc de l'usuari A	tpc de l'usuari B	Descripció
	inicia el tpc de B al port 2000+ID.	S'inicia el tpc de B en el port 2000+ID indicat per tps en la negociació anterior.
envia clau pública de A al port 2000+ID de B.		Envia la clau pública de A a B.
	envia la clau pública de B a A	Envia la clau pública de B a A.
Ja es pot enviar i rebre text xifrat	Ja es pot enviar i rebre text xifrat	Ja es pot enviar i rebre text xifrat, donat que ambdues parts han pogut completar l'algorisme Diffie-Hellman i per tan establir una clau DES comuna.

Nota: ID és el número de posició que se li assigna a la finestra *PantallaDialeg* en el vector de vpd de l'aplicació que rep la trucada.

6. Manual de l'aplicació

L'aplicació és un software de missatgeria instantània que actua tan com a servidor com a client. Permet la connexió a través d'Internet o d'una xarxa local a d'altres instàncies de l'aplicació que funcionin en altres màquines.

La gràcia d'aquesta aplicació és que no es connecta a cap servidor central per tal de poder funcionar, sinó que la connexió l'estableix directament amb l'altre interlocutor via IP. Això farà que per exemple es pugui utilitzar en empreses on hi ha xarxa local però no accés a Internet, o bé aquest està restringit per un firewall que impedeix l'ús d'altres aplicacions de missatgeria instantània basades en un servidor central.

També cal tenir en compte que tota comunicació entre dos usuaris d'aquesta aplicació serà sempre xifrada i per tan segura, és a dir, cap tercera persona podrà llegir la conversa que s'està produint entre els dos usuaris.

En els següents punts es veurà quins passos cal seguir per a la instal·lació i execució de l'aplicació, així com es veurà una explicació de totes les opcions del programa i com es realitzaria un connexió amb un altre usuari.

6.1 Instal·lació i Execució

Abans de poder iniciar l'aplicació caldrà verificar que s'han realitzat una seria de passos.

Primer de tot caldrà comprovar que s'ha instal·lat de forma correcta una versió de Java, per exemple el SDK 1.4.2. En cas de que la versió de Java instal·lada sigui anterior a la versió 1.4 caldrà instal·lar també les classes JCE de criptografia i seguretat.

Caldrà comprovar que el Java instal·lat consti en la variable d'entorn *PATH* així com que la variable d'entorn *CLASSPATH* estigui ven definida (existeixen multitud de referències per veure com realitzar això).

A partir d'aquí només cal agafar els fitxers font (fitxers *.java) o els fitxers en codi binari (fitxers *.class) de l'aplicació i copiar-los a alguna carpeta.

Si tenim els fitxers font, caldrà primer compilar-los mitjançant els següents passos:

1) Entrar a la carpeta on hi ha els fitxers font.

2) Executar l'ordre "*javac *.java*" o bé "*javac PantallaPrincipal.java*"

Si el Java està ven configurat en la màquina la compilació hauria d'acabar sense cap missatge d'error.

Si ja teníem els fitxers en codi binari, o un cop compilats els fitxers font, simplement caldrà executar el fitxer principal de l'aplicació per iniciar el programa. Per fer-ho cal introduir la següent comanda des de la carpeta on hi ha els fitxers de l'aplicació: "*java PantallaPrincipal*".

Figura 6.1 – Pantalla d'Inici –

Un cop iniciada l'aplicació hauria d'aparèixer una finestra com la que mostra la figura 6.1 i ja podríem començar a fer ús de l'aplicació així com rebre trucades d'altres usuaris de l'aplicació.

6.2 Opcions del programa

Només iniciar l'aplicació ens hauria d'aparèixer la finestra de la figura 6.1 mostrada en l'anterior punt. En aquesta finestra hi trobem dos menús, el de *configuració* i el d'*ajuda*, així com altres camps i botons.

El menú de *configuració* conté els elements mostrats en la figura 6.2, és a dir, una opció per al canvi de nom d'usuari, una opció per al canvi de port de recepció de peticions de trucades i una opció per tancar l'aplicació.

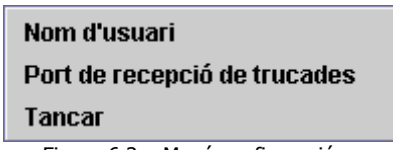

Figura 6.2 – Menú configuració –

Per altra banda, el menú *Ajuda* conté els elements mostrats en la figura 6.3, els quals permeten obtenir informació d'ajuda i informació sobre l'autor i la versió del programa.

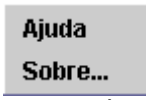

Figura 6.3 – Menú d'ajuda –

Al prémer sobre l'opció *Nom d'usuari* del menú de *configuració* ens apareix una finestra, figura 6.4, en la qual se'ns mostra l'actual nom d'usuari i se'ns permet de canviar-lo per un de nou.

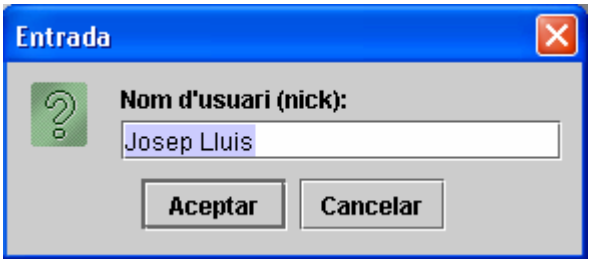

Figura 6.4 – Pantalla canvi de nom d'usuari –

Si l'opció que s'escull és la de *Port de recepció de trucades* del menú *configuració*, aleshores es mostra una finestra, figura 6.5, similar a l'anterior en la qual es pot apreciar l'actual port de recepció de trucades, tenint la possibilitat de canviar-lo. Cal tenir en compte però que els canvis no tindran efecte fins al pròxim inici de l'aplicació.

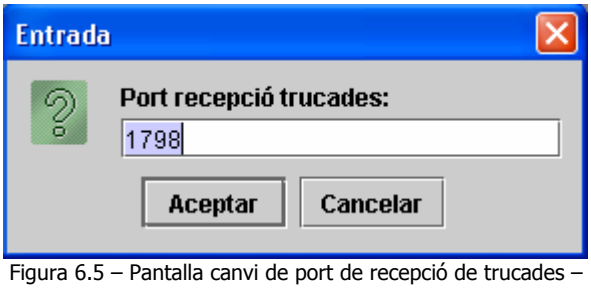

Figura 6.5 – Pantalla canvi de port de recepció de trucades –

Amb l'opció *Ajuda* del menú *Ajuda* apareix un pannell, figura 6.6, en el qual se'ns informa d'on obtenir més informació o ajuda.

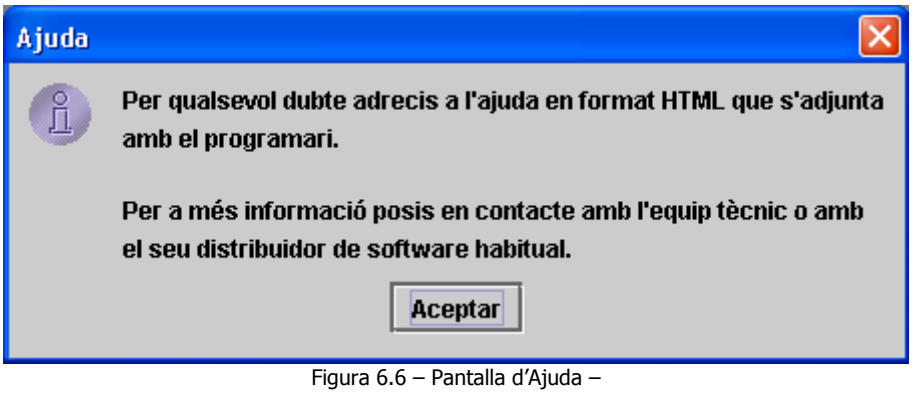

Figura 6.6 – Pantalla d'Ajuda –

Per altra banda, si es prem sobre l'opció *Sobre...* del menú *Ajuda*, ens apareix la típica informació sobre la versió de l'aplicació, l'autor i la data de l'última revisió (figura 6.7).

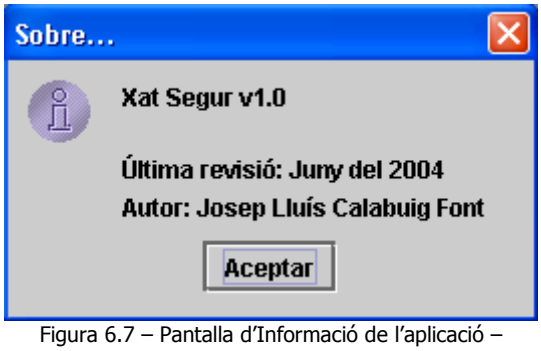

Figura 6.7 – Pantalla d'Informació de l'aplicació –

Pel que fa al camp IP, aquest és utilitzat per indicar l'adreça IP del destinatari d'una trucada que vulguem iniciar, mentre que el camp Port n'és el port de connexió, és a dir, és el port de recepció de peticions de trucades de l'aplicació de l'usuari amb el que volem contactar.

Un cop entrada la IP i port del destinatari, només cal prémer sobre el botó Acceptar per a iniciar el procés de petició de trucada a l'usuari amb qui es vol establir una conversa.

Tan quan s'inicia una trucada com quan se'n rep una, apareix una finestra igual a la que es mostra a la figura 6.8 (en la que es mostra a continuació s'ha aconseguit establir connexió amb èxit i s'està fent la negociació de claus, però els missatges que hi apareixen poden variar segons si s'està fent de part servidor o part client, així com si s'estableix la comunicació amb èxit o hi ha problemes en la connexió).

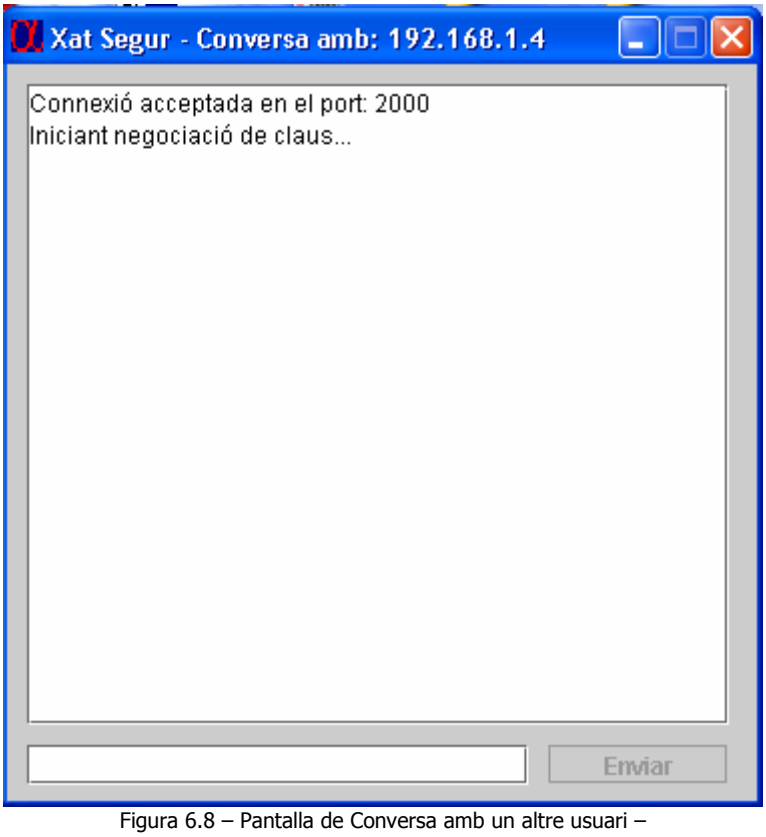

Figura 6.8 – Pantalla de Conversa amb un altre usuari –

En aquest punt ambdues parts implicades poden trigar un cert temps a completar la negociació de claus (uns pocs segons), però un cop completada hauria aquesta negociació, hauria d'aparèixer un missatge com el que es mostra en la figura 6.9.

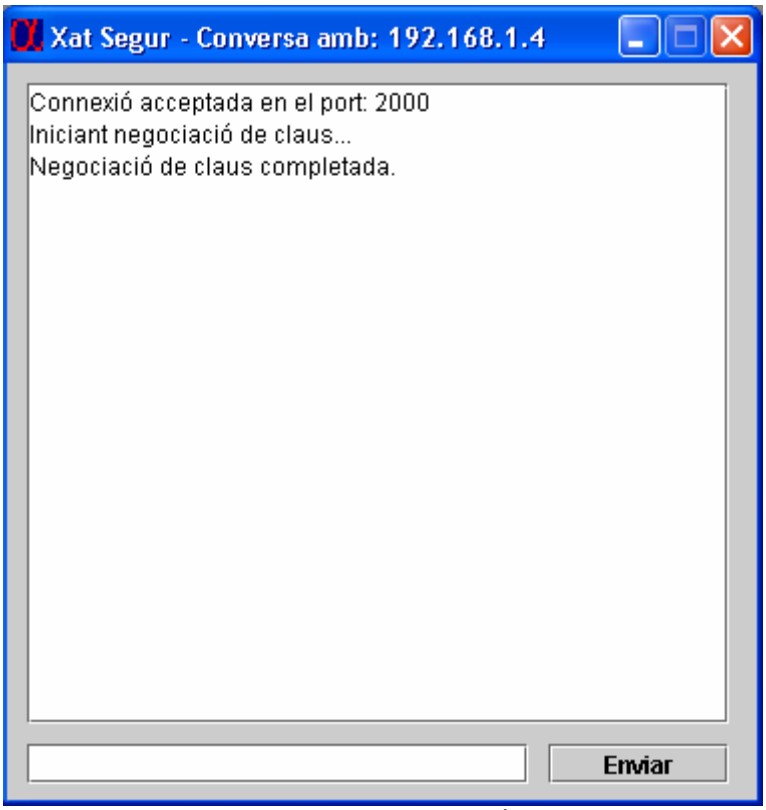

Figura 6.9 – Pantalla de Conversa amb negociació de claus completada –

A partir d'aquest punt es pot ja mantenir una conversa amb l'altre usuari (es podrà observar això pel fet que el botó *Enviar* ja està habilitat, mentre que en el procés de negociació de claus està deshabilitat). Un exemple de conversa es pot veure en la figura 6.10.

Cal mencionar que els missatges que s'enviïn els dos interlocutors a partir d'aquest moment ja seran xifrats mitjançant l'algorisme DES, i per tan, en un principi haurien de ser confidencials.

Quan un usuari vulgui donar per finalitzada la trucada té dos possibilitats: O bé tanca la finestra de la conversa mitjançant el botó en forma de creu que apareix a la cantonada superior dreta de la finestra de conversa; o bé enviar el text */quit* (amb la barra inclosa). Qualsevol de les dues opcions provocarà que es tanqui la connexió i que a l'altra usuari li aparegui un missatge com el que mostra la figura 6.11 (si s'utilitza el text */quit* per tanca la comunicació el que provoca la desconnexió també veu el missatge en la seva finestra).

Evidentment, si la connexió es talles per alguna altra raó també apareixeria un missatge com el que s'esmentava fa un moment, o bé segons el cas, algun altre tipus de missatge d'error més específic.

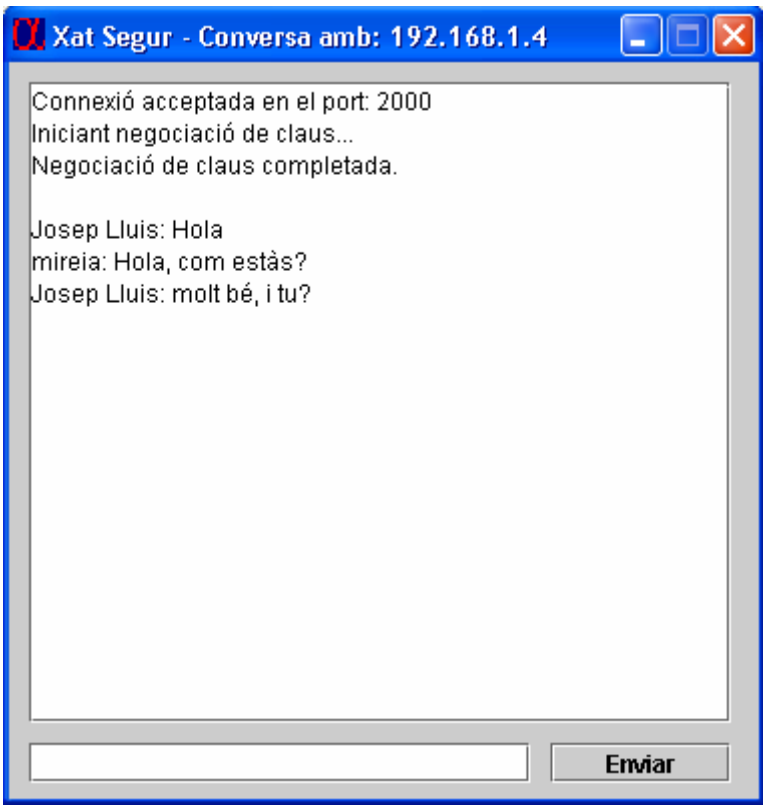

Figura 6.10 – Pantalla de Conversa amb conversa en curs –

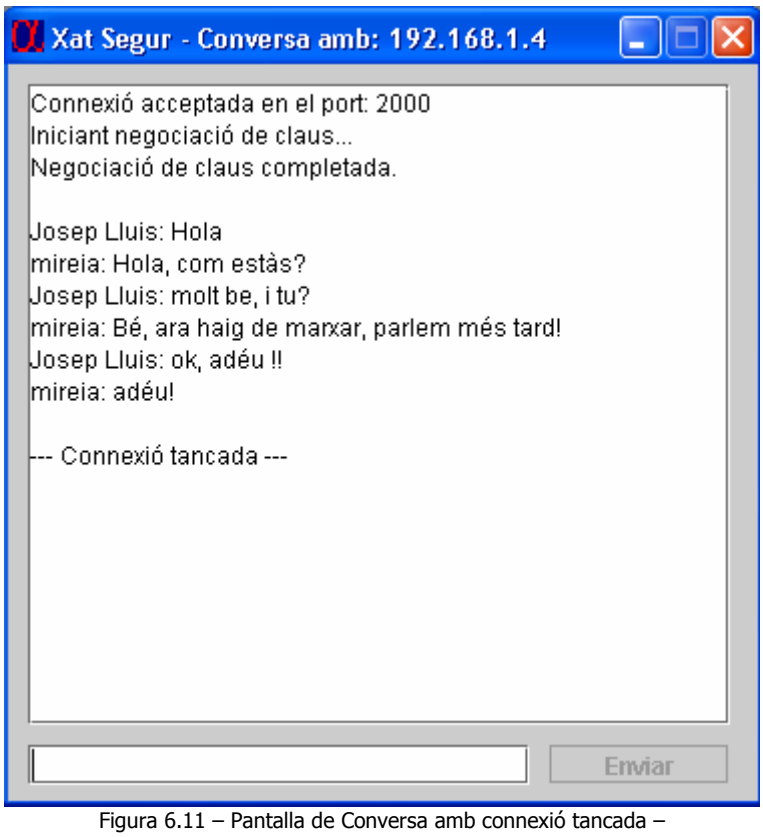

Figura 6.11 – Pantalla de Conversa amb connexió tancada –

Quan s'ha dit que un usuari iniciava una trucada s'ha entès com a que l'usuari receptor de dita trucada l'acceptava sense tenir la possibilitat de refutar-la. La veritat és que l'aplicació només farà això si es té marcada la casella de la pantalla principal que s'indica amb un cercle vermell en la figura 6.12.

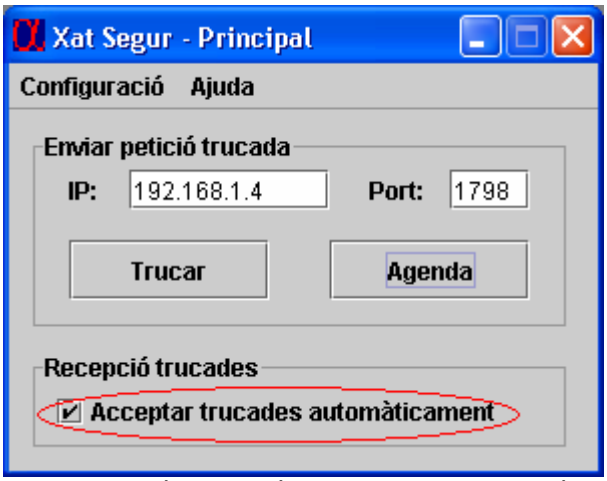

Figura 6.12 – Opció de recepció de trucades de forma automàtica –

Si aquesta casella no està marcada, aleshores a l'usuari que fa de receptor de la trucada li apareixerà un missatge tal com el de la figura 6.13 per a què pugui acceptar o refutar la trucada entrant que rep.

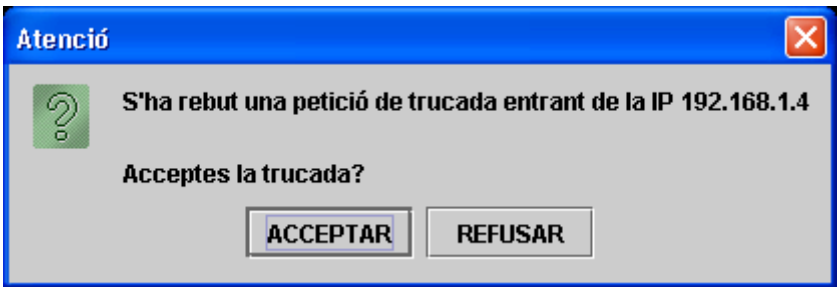

Figura 6.13 – Pantalla on es pregunta si s'accepta la trucada o no –

De totes maneres, hi ha un límit en la quantitat total de trucades o converses simultànies que el programa permet mantenir. De fet aquest nombres està fixat en 10 converses, de manera que en el moment en que s'intenta obrir una onzena conversa apareix un missatge d'error com el que mostra la figura 6.14 (Caldrà tenir en compte que les finestres de conversa encara obertes però que tinguin la part de comunicacions tancada seguiran contant també per a aquest límit).

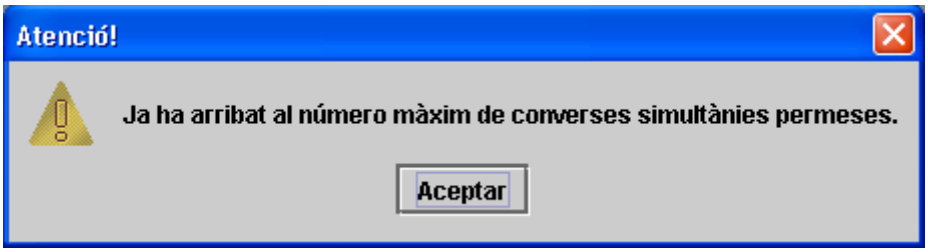

Figura 6.14 – Missatge indicant que ja hi ha masses finestres de converses obertes –

Retornant a la pantalla principal (mirar figura 6.1) podem recordar que ens havíem deixat per veure encara el botó *Agenda*, el qual al ser premut ens obrirà una nova finestra com la que mostra la figura 6.15.

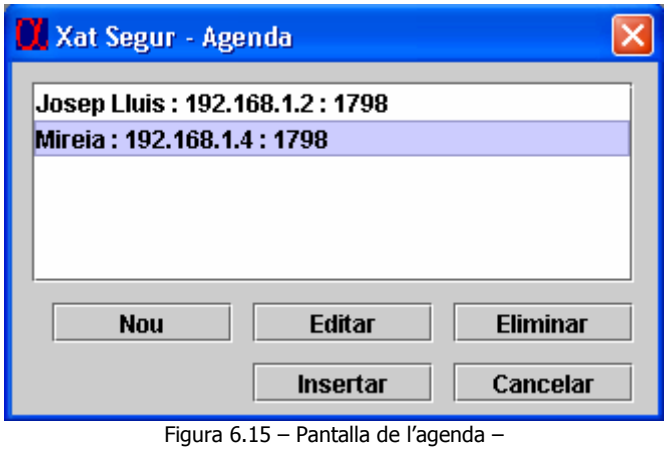

Figura 6.15 – Pantalla de l'agenda –

Aquesta Agenda ens mostra un llistat de tots els contactes que hi haguem introduït, així com ens permet crear nous contactes, editar-ne de ja creats o eliminar els que ja no vulguem.

Les dades que es mostren de cada contacte són el seu nom, la seva adreça IP i el port que utilitza per a la recepció de trucades entrants (és a dir, el port amb el que em de contactar). Aquestes mateixes dades seran les que se'ns demanaran al crear un nou contacte i les que podrem editar d'un contacte ja existent.

Per afegir un nou contacte a l'agenda només caldrà prémer sobre el botó *Nou*, de manera que ens apareixerà una finestra com la de la figura 6.16 i on podrem entrar totes les dades del nou contacte.

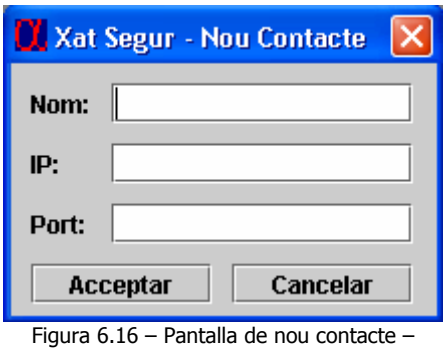

Figura 6.16 – Pantalla de nou contacte –

En canvi si el que volem és editar un contacte ja existent, primer el seleccionem de la llista de l'agenda i posteriorment premem sobre el botó Editar. Ens apareixerà una finestra igual a la de creació d'un nou contacte, però ara amb les dades del contacte a editar ja introduïdes i apunt per ser modificades. En la figura 6.17 es pot veure un exemple d'aquesta finestra.

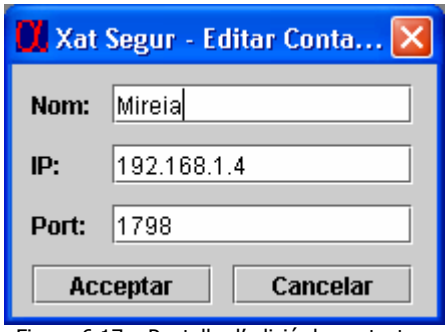

Figura 6.17 – Pantalla d'edició de contacte –

No cal dir que per eliminar un contacte de l'agenda simplement cal seleccionar-lo i prémer sobre el botó *Eliminar*.

Per altra banda, si es vol utilitzar l'agenda per configurar una trucada, és a dir, per afegir les dades d'un contacte en els camps IP i Port de la pantalla principal, només caldrà seleccionar el contacte a trucar i prémer el botó *Insertar* (la finestra es tancarà i en la finestra principal ja hi constaran l'adreça IP i el port del contacte que s'havia seleccionat).

7. Conclusions i línies futures

A l'aplicació se li ha fet un exhaustiu repàs en busca de bugs, així com se li han passat diferents proves simulant diferents situacions i diferents possibles problemes amb què es podria trobar. El resultat ha estat un codi força robust i estable, que en casos excepcionals pot deixar anar algun missatge d'error per la consola en forma de text (els quals s'han minimitzat), però que en cap cas implicarà un mal funcionament de l'aplicació o que aquesta es penji o es tanqui inesperadament.

També s'ha intentat crear una interfície gràfica el més simple i amigable possible, de manera que l'usuari no es pugui desorientar durant l'ús de l'aplicació. En aquest aspecte també s'han tingut molt en compte aspectes de comoditat per a l'usuari (per exemple: en la pantalla de diàleg no cal prémer el botó Enviar cada cop que es vol enviar un missatge, sinó que s'ha generat un receptor d'events en la casella per a escriure el text a enviar, de manera que captura quan l'usuari prem l'enter i així envia el missatge que s'hagi escrit sense necessitat de prémer el botó Enviar).

Altres consideracions que s'han de tenir en compte, així com les possibles aplicacions i ampliacions que es podran fer d'aquest software es presenten en els següents punts.

7.1 Consideracions d'execució

L'aplicació desenvolupada s'ha testat, amb èxit, sota diferents plataformes Linux, així com sota Windows 98 i Windows XP, utilitzant per a les proves diferents versions del SDK, concretament amb les versions 1.4.1 i 1.4.2 (també s'ha fet alguna prova puntual amb la versió 1.3, a la qual se li han hagut d'incorporar les classes de seguretat i criptografia en la carpeta d'extensió de classes del SDK).

Un aspecte que s'ha detectat en totes les execucions és que la primera negociació de claus que es realitza és molt més lenta que la resta de negociacions de claus que es puguin produir en els següents establiments de trucada.

El fet que les primeres accions siguin més lentes és un fenomen típic de la majoria d'aplicacions que corren sota la plataforma Java i es deu bàsicament a que Java utilitza la JVM (Java Virtual Machine). La JVM necessita carregar-se prèviament a l'execució i molt sovint necessita resoldre els enllaços dinàmics que han quedat pendents de concretar en la pre-compilació, això fa que molt freqüentment la primera execució d'una part del codi sigui més lenta, mentre que les posteriors execucions d'aquesta mateixa part del codi, al no tenir que fer tants passos previs, siguin més ràpides.

7.2 Protocol de comunicacions utilitzat

La manca d'estàndards en el mercat del software de missatgeria instantània ha fet impossible que es pogués seguir un estàndard pel que fa al procés d'establiment de comunicació, així com en el d'enviament i recepció dels textos dels usuaris.

Així, tot el protocol de comunicacions que s'ha utilitzat és totalment particular d'aquesta aplicació i totalment incompatible amb qualsevol altre sistema de missatgeria instantània que existeixi. A més a més, el sistema de comunicacions és extremadament simple, doncs només s'han implementat els aspectes justos que requeria l'aplicació que s'implementava . Això no vol dir que les comunicacions no segueixin un estàndard, doncs es realitzen per mitjà del protocol TCP/IP i via Sockets de Java (que ja és tota una garantia), doncs on hi ha diferencies és a nivell d'aplicació.

De totes maneres, encara que hagués existit un estàndard aquest no ens hagués servit, doncs la naturalesa de l'aplicació que s'ha creat és certament diferent de la que s'acostuma a utilitzar en altres programes de missatgeria instantània (aquí no hi ha servidor, és punt a punt, i s'utilitza el xifrat DES i l'algorisme Diffie-Hellman per a la negociació de claus, aspectes que no trobarem en la resta de programes de missatgeria instantània).

7.3 Reciclatge del codi font i ampliació

La idea principal del codi font que s'ha creat és que sigui el màxim de reutilitzable possible, permetent desenvolupar així noves aplicacions millorades d'aquest tipus de software de missatgeria instantània punt a punt.

Evidentment, l'opció més desitjable seria ampliar l'aplicació desenvolupada amb noves funcionalitats i corregint tots els possibles errors i bugs que hi poguessin existir actualment.

Seria interessant que en futures versions es permetés l'ús de diferents colors en la pantalla de diàleg, de manera que fos més senzill identificar qui parla en cada moment. Així mateix, l'ús d'emoticons també ajudaria a popularitzar aquesta eina.

També seria adequat que es poguessin mantenir converses en grup, doncs l'aplicació actual només permet realitzar converses entre dos usuaris dins d'una mateixa conversa. Evidentment això implicaria canvis significatius, ja que el mateix algorisme de Diffie-Hellman s'hauria de modificar per tal de que fos possible la negociació de claus entre 3 o més membres (també es podrien estudiar altres alternatives a la negociació de claus entre 3 membres que provablement serien més senzilles).

Tampoc quedaria de menys, tot i que elevaria la complexitat de l'aplicació, el permetre la transferència de fitxers via ftp entre dos usuaris connectats, tal i com ja fan la gran majoria d'aplicacions de missatgeria instantània.

Altres aspectes com la possibilitat de mantenir converses per veu o vídeo es desmarcarien dels objectius a curt termini, doncs això ja requeririen un nombre d'hores de feina i una complexitat fora del que pretenia ser aquesta projecte.

Pel que fa al tema de la seguretat, no estaria malament substituir el xifrat DES per un xifrat més robust, com per exemple el triple DES, de manera que el desxifrat per força bruta fos realment quasi impossible. Així com estaria bé també que l'aplicació pogués fer ús de certificacions electròniques, per tal d'evitar el problema de la suplantació d'identitats.

8. Glossari

Terme	Descripció
AOL	American OnLine
CLASSPATH	Variable d'entorn per definir on cal trobar les classes de Java
DES	Data Encryption Standard
FTP	File Transfer Protocol
GNOME	GNu Object Model Environment
GNU / GPL	GNU General Public License
HTTP	HiperText Terminal Protocol
IP	Internet Protocol
JCE	Java Cryptography Extension
JDK	Java Development Kit
JVM	Java Virtual Machine
KDE	K Desktop Environtment
LINUX	Sistema operatiu de codi obert i gratuit creat inicialment per Linus Torwals
OO	Orientat a Objectes
PATH	Variable d'entorn per definir on hi ha el compilador i la JVM de Java
Peer-to-Peer	Punt a Punt
SDK	Software Development Kit
TCP	Transmission Control Protocol
TFC	Treball Final de Carrera
UML	Unified Modeling Language
UNIX	Sistema operatiu de codi tancat i propietari de l'empresa ATT
URL	Unified Resource Locator

9. Bibliografia

Informació de seguretat i criptografia

- http://www.semper.org/sirene/outsideworld/security.html
  URL amb informació de seguretat i criptografia.

- http://www.iec.csic.es/criptonomicon/seguridad/
  URL amb informació de seguretat i criptografia.

- Document electrònic: seguretat.pdf
  Títol: Seguretat a les xarxes
  Autor: Manuel Pons Martorell

- http://daniellerch.com/html/diffie_hellman.html
  URL amb informació sobre l'algorisme Diffie-Hellman.

- http://www.eurologic.es/cifrado/clavepub.htm
  URL amb informació sobre l'algorisme Diffie-Hellman.

Informació de Java i Programació Orientada a Objectes

- http://java.sun.com
  URL oficial sobre Java de Sun Microsystems.

- Material Docent, en format CD, de l'assignatura "Fonaments de Programació II" de l'Enginyeria Tècnica en Informàtica de Gestió de la UOC (Universitat Oberta de Catalunya).

- "Java 2 Network Security"
  IBM redbook printed by Prentice Hall PTR.
  Marco Pistola et al.

- "La bíblia de Java"
  Anaya multimedia
  Laurence Vanhelsuwé et al.