Treball fi de Carrera

XAT SEGUR

Alumne
Gerard Prats Abril

Consultor
Antoni Martinez Balleste

08/01/2006

Memòria TFC

Index de continguts

1. Introducció i especificació del problema 3
2. Fonaments o estat de l'art 7
3. Disseny del programa 13
4. Aspectes concrets de la implementació 23
5. Manual d'instal·lació 28
6. Manual d'usuari 31
7. Proves 36
8. Valoració econòmica 39
9. Conclusions 42
10. Glossari 44
11. Bibliografia / Recursos 45
12. Codi font 46

1. INTRODUCCIÓ I ESPECIFICACIÓ DEL PROBLEMA

El creixement desmesurat de les xarxes d'ordinadors ha provocat en els últims anys un important augment en els problemes relacionats amb la seguretat, afectant a tota mena d'àmbits del món informàtic. La seguretat s'ha convertit d'aquesta manera en una de les mancances més rellevants de la nova xarxa d'internet.

Perspectiva legal i tècnica de la seguretat informàtica

La signatura digital, regulada a l'Estat espanyol mitjançant el Reial decret llei 14/1999, de 17 de setembre, va esdevenir un primer pas cap a la regulació de la seguretat a aquesta xarxa. Posteriorment, l'any 2002, el govern va aprovar una nova llei de serveis de la societat de la informació (LSSI), a fi de regular de manera més global el conjunt de les activitats que es duen a terme a internet.

Malgrat existir un marc legal i una normativa vigent al respecte, la seguretat segueix sent una de les principals mancances de moltes de la aplicacions que utilitzen les comunicacions de la xarxa i que intercanvien de manera oberta informació de caràcter personal o privat. La naturalesa de les xarxes actuals de computadors, sobretot en àmbits locals o LAN, estableixen un context idoni per als interessos de persones malintencionades que poden, mitjançant una senzilla aplicació *sniffer* o *escoltador de xarxa*, tenir a l'abast qualsevol dada que circuli per la xarxa a la que estan connectats. Aquestes aplicacions intercepten la informació que travessa una interfície de xarxa i, una vegada l'han aconseguit capturar, l'emmagatzemen per al seu posterior anàlisis. Tot això es pot fer de forma passiva, sense modificar les dades que envia el programa, i sense que sigui possible saber si algú les ha llegit o no.

Quan dos usuaris es decideixen a intercanviar informació a través d'internet poden ser víctimes d'un atac de diferents maneres:

- **Interrupció**: Un tercer usuari fa que les dades enviades NO arribin al seu destí
- **Intercepció**: Les dades poden arribar, però també són capturades per una tercera persona.
- **Modificació**: Les dades que arriben al destinatari han estat modificades per un tercer
- **Impostura**: El primer usuari no ha enviat dades per un tercer fa que el segon en rebi com si fossin realment enviades per l'altre.

En tot cas, és evident que la normativa no és suficient per protegir l'usuari d'aquests atacs, sinó que les pròpies aplicacions han de ser capaces de discriminar aquella informació que l'usuari pot considerar privada i posar-la en un entorn segur desde un punt de vista tecnològic. De la mateixa manera caldria autenticar els missatges per evitar atacs del tipus impostura.

Hi ha diferents serveis de seguretat coneguts que les aplicacions poden oferir als usuaris per protegir-los d'atacs per part de tercers:

- **Confidencialitat**: Es tracta de codificar les dades de manera que només el destinatari pugui llegir-les. Si un tercer les intercepta no les podrà interpretar.
- **Integritat**: Consisteix a proporcionar un mecanisme per garantir que les dades no han estat modificades per cap tercer.
- **Autenticació**: Es tracta de verificar la identitat de l'usuari emissor de cada missatge de manera que no sigui possible la suplantació.
- **No rebuig**: Consisteix a oferir la possibilitat de demostrar que un usuari ha enviat o rebut un determinat missatge.

Existeixen en la actualitat recursos suficients per poder oferir aquests diferents serveis. Entre ells, l'esmentada signatura digital, que permet la identificació inequívoca de missatges, emissors i destinataris en qualsevol comunicació, ja sigui mitjançant el protocol de correu electrònic, una transacció web, etc. Aquest sistema és molt adequat per a entorns de comunicació en els que, com a mínim, un dels dos comunicants ha d'estar correctament identificat mitjançant un certificat oficial, com seria el cas d'una connexió amb una entitat bancària. Aquest és, però, un dels principals handicaps de la signatura digital, ja que els tràmits per obtenir un certificat personal legal no són immediats i requereixen d'una visita presencial a la delegació de l'agència tributaria.

Malgrat això, a banda de la signatura digital, existeixen tota una sèrie de tècniques criptogràfiques no dependents d'aquest sistema que permeten implantar l'intercanvi segur d'informació en plataformes on els usuaris no hagin d'estar identificats de manera tant segura i legal. Aquest és el cas de les tècniques de xifratge simètric que s'aborden més a fons en un apartat posterior d'aquest manual i que permeten aplicar diferents algorismes d'encriptació per protegir les dades que intercanvien les aplicacions. Cal tenir en compte, però, que cal escollir l'algorisme més adequat per l'aplicació que es desenvolupa, ja que normalment a més seguretat menys velocitat de xifratge, i a l'inrevés.

Aplicacions de missatgeria

Si ens centrem en el marc del programari, la seguretat informàtica afecta a aplicacions de tot tipus. Aquest treball tractarà d'aprofundir en els sistemes de missatgeria, que s'han fet molt populars a internet sobretot desde que Microsoft va incloure la versió 4 del seu *Messenger* al sistema operatiu Windows XP. A banda d'aquesta coneguda solució, n'hi ha d'altres com poden ser *ICQ*, Instant Messenger de AOL o *Yahoo Messenger*, que bàsicament tenen la mateixa funcionalitat que la de Microsoft, encara que treballen en xarxes diferents. Cal dir també que cada un d'aquests sistemes de missatgeria té un servidor i un programa client propis per connectar-s'hi, i aquest últim es sol oferir de manera gratuïta per descarregar-lo desde la web comercial.

La missatgeria instantània és un punt intermig entre els sistemes coneguts de xat, com podria ser IRC (Internet Relay Chat), i els missatges de correu electrònic que resideixen de manera permanent a l'escriptori de l'usuari mentre hi hagi una connexió a internet disponible. Aquestes aplicacions ofereixen una finestra on es pot escriure un missatge en text pla o bé afegint-hi *emoticons* que representen els estats d'ànim de l'usuari i enviar-lo a un o més destinataris. Aquests últims reben el missatge a l'acte i el poden contestar de manera immediata, és a dir, en temps real.

A les diferents versions de programes de missatgeria que han anat sorgint s'hi ha començat a afegir tota una sèrie d'aplicacions i utilitats extra com poden ser les trucades telefòniques a través d'internet, els missatges d'avís al mòbil, així com jocs i compartiment de fitxers entre usuaris.

Existeixen, a més, algunes aplicacions clients híbrides, no desenvolupades per aquestes empreses, i que permeten a l'usuari configurar comptes de diferents sistemes en un sol programa client. D'aquestes n'existeixen moltes de lliure distribució que poden descarregar-se desde internet, essent algunes de les més conegudes *Trillian*, *Miranda*, *Instant Messenger* i *Gaim*.

Aquest sistema de comunicació ha guanyat adeptes de manera molt ràpida en els últims anys. Es calcula que a través de America OnLine, un dels principals proveïdors d'aquests tipus de serveis, corren aproximadament uns 300 milions de missatges cada dia, el que ve a promitjar uns 3 missatges diaris per usuari.

Aspectes de la seguretat en missatgeria instantània

Els sistemes de missatgeria generen tot un corrent d'informació que travessa la xarxa amb continguts ben diversos. És evident, però, que ningú espera ni desitja que algú altre pugui estar escoltant, o en aquest cas llegint, la informació de la conversa que està mantenint amb algú altre a través de Messenger o qualsevol altre programa.

Malgrat això, la gran majoria dels sistemes de missatgeria actuals no inclouen característiques de seguretat per protegir aquesta informació. En realitat, com ja s'ha comentat, és suficient un escoltador qualsevol de xarxa per poder capturar la informació d'una conversa i poder-ne llegir directament el contingut. A més a més, per si això fos poc, existeixen aplicacions com *MSN Sniffer*, que capturen, organitzen i mostren la informació que troben en una xarxa classificant-la per converses i mostrant-la exactament com si l'intrús estigués veient la conversa a la pantalla d'un dels usuaris.

En tot cas es pot dir que sorprèn que això sigui així, donada l'amplia difusió que tenen actualment les diferents tècniques de xifratge en seguretat informàtica. Si cerquem, per exemple, algun tipus de referència que faci l'empresa propietària del conegut Messenger pel que fa a la seguretat, ens trobarem amb una curiosa plana on es donen una sèrie de consells per millorar la seguretat en l'ús de l'eina. Sorprenentment, alguns d'aquests consells es podrien obviar si el servei de missatgeria portés inclòs un sistema de xifratge segur. A la plana es fa referència a evitar proporcionar dades de caràcter personal en les converses, així com no enviar missatges de caràcter privat en xarxes privades, com seria el cas de la xarxa de la feina. Aquest article és consultable a la següent adreça: http://www.microsoft.com/spain/seguridad/usuarios/chat/imsafety.mspx .

Davant d'aquesta situació, és evident que no existeixen prou garanties en aquests sistemes ja que, a més de informació personal i privada, en converses de missatgeria s'hi pot capturar qualsevol cosa: informació legal, comercial, dades personals, bancàries, etc. En tot cas, ometre aquest tipus d'informació en les converses limita la utilitat de l'eina i no és una solució real i definitiva als problemes de seguretat plantejats.

Pel que fa als serveis de seguretat que s'esmentaven anteriorment en aquest apartat, la confidencialitat, la integritat i l'autenticació de les dades serien fites adequades a assolir en el desenvolupament d'una eina d'aquestes característiques. El no rebuig, per la seva part, és un requeriment de seguretat més adequat per transaccions electròniques i similars.

2. FONAMENTS O ESTAT DE L'ART

El present treball desenvolupa una solució en forma de programari a la problemàtica de seguretat comentada a l'apartat anterior. L'aplicació reprodueix fidelment les característiques de tots els sistemes comercials de missatgeria que s'han comentat. És a dir, consisteix en un servidor de missatgeria i al qual s'hi poden connectar múltiples usuaris mitjançant una aplicació client per mantenir diferents converses entre sí. D'ara en endavant es farà referència aquestes dues aplicacions com a *Client Missatger* i *Servidor Missatger*.

La diferència essencial que ofereix el sistema d'aquest treball respecte les solucions comercials conegudes és que que tota la informació que intercanvien els diferents clients viatja xifrada mitjançant un mètode de xifratge reconegut i provat com és DES, i en base a un sistema de comunicació xifrada de clau compartida. Les característiques del mètode de xifratge i altres aspectes tècnics específics de les aplicacions es comenten de manera més detallada a l'apartat *Aspectes concrets de la implementació*. D'altra banda, es pot dir que el xifratge de la informació mitjançant claus aporta a la aplicació dos serveis de seguretat: Confidencialitat i autenticació.

En aquest sistema cada usuari es dóna d'alta mitjançant una adreça e-mail pròpia única, que serà la que l'identifiqui com a usuari. El servidor ha de emetre, a més, una clau pública i una privada per a cada un dels usuaris per poder donar funcionalitat al xifratge. Aquestes claus tenen un format B64, bàsicament una seqüència de caràcters i nombres, i hauran de ser utilitzades pels usuaris per accedir al sistema i comunicar-se amb els seus contactes. Per a establir comunicació amb un altre usuari és necessari obtenir la seva adreça i la seva clau pública (la privada és totalment intransferible), de manera que en base a la clau privada pròpia i a la pública remota el software Missatger Client pot establir una clau de xifratge diferent i única per a cada conversa.

És important tenir en compte que les claus són generades mitjançant l'algorisme Diffie-Hellman, també comentat àmpliament a l'apartat *Aspectes concrets de la implementació,* i que una clau qualsevol aleatòria no pot funcionar en el sistema. Això permet autenticar de manera més confiable els usuaris i assegurar que totes les converses que tenen lloc utilitzen una clau de xifratge diferent. En aquest sentit, quan l'aplicació rep un missatge xifrat amb una clau desconeguda el descarta i actua com si no hagués rebut res.

Així doncs, el sistema de claus permet garantir que, de cara a possibles intents d'escoltar les converses, hi haurà tantes claus de xifratge com parelles d'usuaris possibles. Això limita moltíssim qualsevol possible intent de desxifrar aquesta clau per part de possibles intrusos, i fa que només

qui hagi obtingut la clau pública d'un usuari i estigui en possessió d'una clau privada pròpia pugui establir una conversa en clar amb ell.

Aquesta petita però efectiva modificació respecte la resta de sistemes de missatgeria existents esdevé un canvi molt rellevant pel que fa a la seguretat. El principal avantatge que comporta és que el fet que la informació de la conversa travessi diferents xarxes deixa de ser un problema, doncs ho fa en un format que no pot ser llegit per cap usuari en temps real. Aquest fet augmenta notablement la confiança de l'usuari sobretot en xarxes no particulars com serien la de la feina, la universitat o qualsevol altre entorn on es pogués connectar, i li permeten utilitzar un canal de comunicació no segur de manera totalment segura i fiable.

S'eviten així tota menta de restriccions en el contingut dels missatges, podent utilitzar, a més, la missatgeria com un sistema segur a través del que es poden enviar dades personals, bancàries o privades, de manera més confiable que un compte de correu electrònic qualsevol. Així doncs, si hem de fer arribar a algun conegut dades de tipus personal o confidencial, resultaria més segur fer-ho a través del Client Missatger que no pas mitjançant un compte de correu Hotmail, per posar un exemple.

Altrament, a diferència de la resta de sistemes, en el Missatger és imprescindible conèixer no només el compte d'usuari d'aquelles persones a qui tenim a la llista de contactes sinó també la seva clau pública. En realitat, un usuari que només disposa de l'adreça e-mail d'un dels seus contactes pot tenir-lo a la seva llista i visualitzar el seu estat, però no podrà establir mai una conversa en clar.

Una consideració final a fer respecte aquesta implementació és que, amb el disseny que es proposava per el TFC, resulta relativament senzilla de desenvolupar. És a dir, la seguretat és un petit afegit al sistema de missatgeria i no comporta grans despeses de recursos per l'aplicació, la qual cosa justifica encara més que sigui una opció viable i interessant de cara al futur.

Pel que fa a altres possibles implementacions de criptografia en missatgeria instantània, ja s'ha comentat que la majoria de les aplicacions no tenen en compte aquest aspecte de la seguretat, però malgrat això hi ha alguna excepció. El sistema més semblant el trobem al client Gaim, que no cal oblidar que es tracta únicament d'un programa client híbrid, però que permet, per exemple, comunicar-se a través del Microsoft Messenger escollint el mètode de xifratge desitjat. Això sí, el xifratge només serà possible quan a l'altre extrem de la conversa hi hagi un usuari amb el mateix client i sistema de xifratge activat. D'altra banda, per AIM existeix també la possibilitat d'incorporar PGP com a eina de xifratge.

Recursos obtinguts durant el període de recerca

Algorisme Diffie-Hellman

La utilitat d'aquest algorisme és **permetre un entorn segur per la negociació i l'intercanvi de claus** malgrat s'estigui utilitzant un canal de comunicació no segur, és a dir, sense encriptació. Es tracta d'un algorisme de clau pública que s'acostuma a utilitzar conjuntament amb algorismes de xifratge simètric.

La seqüència de negociació de les claus és la següent:

- L'emissor escull un nombre primer gran **p** i un enter **g** més petit que **p** que fa arribar al destinatari. A continuació escull un nombre gran menor de **p** que utilitzarà com a clau privada $\mathbf{d_A}$ i fa el càlcul de la clau pública corresponent de la següent manera: $\mathbf{e_A = g^{d_A} \bmod p}$
- De manera semblant, el destinatari escull una clau privada $\mathbf{d_B}$ i una clau pública emprant: $\mathbf{e_B = g^{d_B} \bmod p}$
- Els dos participants intercanvien en aquest moment les claus públiques respectives i calculen un secret compartit: $\mathbf{s_A = e_B^{d_A} = (g^{d_B})^{d_A} = g^{d_B d_A}}$ per l' emissor i $\mathbf{s_B = e_A^{d_B} = (g^{d_A})^{d_B} = g^{d_A d_B} = g^{d_B d_A} \bmod p}$ pel destinatari.

Aquest sistema permet que, encara que un tercer interceptés les dades p, g i les claus públiques, no li seria possible calcular el secret compartit sense tenir una de les claus privades. Per a obtenir-les caldria calcular el logaritme discret d'una de les claus públiques que és un problema intractable computacionalment.

Com a contrapartida, aquest algorisme és sensible a atacs del tipus "*home al mig*". En un d'aquests atacs un tercer podria fer-se passar per l'emissor o el receptor, donat que Diffie-Hellman no disposa d'un mecanisme per validar la identitat dels participants en la comunicació.

Algorismes de xifratge simètric

Entenem com a mètodes de xifratge simètric o criptosistemes de clau compartida tots aquells que empren una mateixa clau per a xifrar i desxifrar la informació que s'envia entre els dos terminals de la comunicació. Entren en aquesta categoria els algorismes de xifratge per blocs i els de xifratge de flux, ambdós detallats àmpliament al mòdul didàctic 3 del dossier Criptografia de la UOC.

El principal inconvenient que pateixen aquest tipus d'algorismes és que prèviament a l'enviament de la informació entre les dues parts cal negociar d'alguna manera la clau compartida que s'utilitzarà per el xifrat/desxifrat de les dades. Com ja s'ha vist a l'apartat anterior, l'algorisme Diffie-Hellman ofereix una solució per tal de que aquesta negociació es pugui realitzar de manera segura i sense que tercers puguin endevinar la clau compartida en base a la informació obtinguda de la comunicació entre les dues parts.

És important també que aquesta clau sigui llarga i difícil d'endevinar, ja que avui en dia els ordinadors poden desxifrar claus molt ràpidament. Per posar un exemple, l'algorisme de xifratge DES utilitza una clau de 56 bits, la qual cosa significa que hi ha un total de 72 mil milions de claus possibles.

Xifratge de flux

En un sistema de xifratge de flux, tant l'emissor com el receptor disposen d'una mateixa clau anomenada **llavor del generador**, i d'un mateix algorisme determinista, anomenat **generador pseudo-aleatori**. Es proporciona la clau com a entrada de l'algorisme i aquest genera a la seva sortida una seqüència anomenada **seqüència de xifratge**.

Concretament, l'emissor va sumant a cada bit del missatge en clar un bit de la seqüència de xifratge, mentre que el receptor fa servir el mateix algorisme determinista per obtenir de nou la seqüència original. Es altament recomanable, en aquest sistema, que la clau o seqüència de xifratge sigui totalment aleatòria, és a dir, que no hi hagi un període a partir del que la seqüència es vagi repetint.

Xifratge per blocs

Els algorismes de xifratge per blocs prenen blocs de mida fixa del text en clar per produir un bloc de mida fixa de text xifrat. És important que el mida del bloc sigui suficientment gran com per evitar atacs de text xifrat. Així mateix, l'assignació de blocs d'entrada a blocs de sortida ha de ser d'un a un, per fer el procés reversible i aparentar aleatorietat.

Existeixen diferents modes de xifratge per blocs:

1. **ECB (Electronic Code Book)**. El missatge es divideix en blocs de *k* bits, i s'encripta cada bloc independentment. En el procés de desencriptar el text xifrat es trosseja en blocs de *k* bits i es procedeix al desencriptat de cada bloc.
2. **CBC (Cipher Block Chaining)**. Aquest mètode fa una o-exclusiva de cada bloc en clar amb l'anterior bloc xifrat, solventant els problemes que presentava el mètode ECB. Pel primer bloc s'utilitza un *vector d'inicialització*. És un dels modes més utilitzats a la pràctica.
3. **OFB (Output Feedback Mode)**. En aquest mode s'utilitza la clau de la sessió per crear un bloc pseudoaleatori de mida gran (pad) que s'aplica en o-exclusiva sobre el text en clar, per tal de generar el text xifrat. L'avantatge que això suposa és que el pad pot ser generat independentment del text en clar i això incrementa la velocitat d'encriptació i desencriptació.
4. **CFB (Cipher Feedback Mode)**. Es tracta d'una variant del mètode OFB per a missatges molt llargs.

Comunicació de processos a nivell de transport: Sockets

Un socket o connector és un punt d'accés als serveis de comunicació en l'àmbit de transport. Tot socket té una adreça que l'identifica i, mitjançant la qual, es pot establir una comunicació amb un altre socket perquè actuï com a extrem d'un canal de comunicació bidireccional.

Els sockets proporcionen una interfície general de comunicació entre processos. D'aquesta manera es poden comunicar diferents processos que s'executen en un mateix ordinador, de manera semblant a com es faria amb les pipes, però la opció més habitual és utilitzar-los com a canal de comunicació entre aplicacions que corren en ordinadors diferents i estan connectats a través de la xarxa.

Així doncs tenim 2 tipus diferents de sockets però en l'aplicació que es desenvolupa en aquest treball només s'utilitzaran aquells que fan referència a la comunicació a través de la xarxa.

Per aquesta comunicació s'utilitza *l'espai de noms d'Internet*, prou conegut pels usuaris de la xarxa. En aquest àmbit els sockets es poden comunicar tant amb protocol *TCP* com *UDP*. Les adreces internet de les màquines es representen mitjançant números de 32 bits (4 bytes) utilitzats en el protocol *IP*. Altrament, existeix un *número de port* assignat al socket, que serveix per a distingir els sockets d'una màquina determinada.

En una connexió entre 2 sockets existeix una certa jerarquia, segons s'indica a continuació.

Socket servidor

És el socket que es crea inicialment i que ha d'actuar com a destinació d'una comunicació. La IP del servidor estarà ben determinada, així que només cal assignar un número de port al socket servidor, per tal que els sockets clients s'hi puguin connectar. En aquesta assignació de port és important tenir en compte que existeixen uns *ports reservats*, entre l'1 i el 1023, i que la resta de ports disponibles van desde el 1024 fins el 65535.

Socket client

En aquest cas el port d'origen sol ser irrellevant i el més habitual és deixar que el sistema li assigni un de manera automàtica, ja que aquest port només el necessitarà el socket servidor per tal de saber a on li ha d'enviar les dades intercanviades.

Estils de comunicació

Existeixen també 2 modes diferents de comunicació entre sockets.

1. **Seqüència de bytes**: Les dades es transmeten d'extrem a extrem com un corrent o flux ordenat de bytes. Estil *orientat a connexió*, per tant utilitza protocol TCP.
2. **Estil datagrama**: Cada vegada que l'emissor escriu dades es transmet un paquet individual o datagrama, i cada vegada que el destinatari en vol llegir rep com a màxim un datagrama. És un estil *no orientat a connexió* i els datagrames poden perdre el seu ordre o arribar duplicats. S'utilitza, per tant, el protocol UDP.

Donades les característiques de seguretat en l'enviament i la recepció de la informació que requereix el Xat Segur, l'aplicació es desenvolupa utilitzant comunicació en seqüència de bytes, és a dir, mitjançant protocol TCP.

3. DISSENY DEL PROGRAMA

Generalitats

El present treball du a terme un software de missatgeria amb un sistema de treball prou semblant al d'altres coneguts programes, com és el cas del Messenger de Microsoft o ICQ.

Aquest tipus de comunicació implica que hi ha d'haver un software intermediari ubicat teòricament a un servidor separat i habilitat perquè les aplicacions client s'hi puguin connectar remotament i rebre informació sobre els altres clients a qui tenen a la seva llista de contactes. En aquest sentit, el funcionament del Missatger serà exactament igual que el dels programes ja citats.

Pel que fa a la implementació, tot el sistema s'ha desenvolupat amb el llenguatge Java, emprant el JSDK 1.4.2 amb l'afegit de les llibreries JDOM i IAIK per a resoldre el tractament de fitxers XML i alguns aspectes de la seguretat, respectivament.

Processos, comunicació i interacció client/servidor

Servidor Missatger

Aquesta eina ha d'estar contínua execució a la màquina servidora. Es tracta d'un programa *dimoni* que manté una connexió socket oberta al port 6000 de la seva IP o adreça a internet. Es tracta d'una aplicació sense interfície gràfica.

El Servidor Missatger du a terme la tasca d'identificar els diferents clients que es connecten al port d'entrada, tenint com a referència un fitxer XML (configura.xml) que detalla els usuaris i contrasenyes vàlides. Quan s'engega el servidor aquest inicialitza a la memòria tots els usuaris que troba dins el fitxer i, per tant, en cas d'establir-hi modificacions és necessari reiniciar-lo. En aquest aspecte és important tenir en compte que els usuaris venen identificats per adreces e-mail, així com succeeix amb la coneguda aplicació Microsoft Messenger. Així doncs, no hi ha d'haver 2 e-mails iguals al sistema doncs aquesta és la clau principal que identifica els usuaris.

Una vegada acceptada una nova connexió entrant, el servidor recull del client un llistat dels usuaris amb qui es comunica, i a partir d'aquest moment es responsabilitza d'informar-lo de qualsevol canvi d'estat en aquesta llista d'usuaris.

A continuació es detallen els diferents estats en que pot trobar-se un usuari al sistema:

- Desconnectat ( ): L'usuari existeix però no està connectat en aquests moments
- Connectat ( ): L'usuari està connectat i disponible per a establir conversa
- Ocupat ( ): L'usuari està connectat però té posat un avís de que està ocupat
- Away ( ): L'usuari està connectat però té posat un avís de que no hi és

A més d'aquest 4 estats, els usuaris tenen la opció d'establir admissió o no admissió ( ) als usuaris de la seva llista. Això queda indicat a la llista d'usuaris que s'envia al servidor quan s'estableix la connexió perquè aquest no enviï cap avís als que no tenen admissió.

A continuació es detalla l'estructura del document de configuració del servidor:

- *configura.xml* [configuració local del servidor]

```
<ConfiguraServidor>
      <Conexio>
            <Port>6000</Port>
      </Conexio>
      <Usuaris>
            <Usuari>
                  <Email>joan_vives@hotmail.com</Email>
                  <Contrassenya>jv234</Contrassenya>
            </Usuari>
            <Usuari>
                  <Email>miquel_soler@hotmail.com</Email>
                  <Contrassenya>ms345</Contrassenya>
            </Usuari>
      </Usuaris>
</ConfiguraServidor>
```

Tota la comunicació entre clients i servidor es realitza mitjançant l'intercanvi directe d'objectes Java. En concret, s'ha dissenyat una classe anomenada *SocketMissatge* que s'empra en els enviaments de dades en totes dues direccions (client<->servidor). El sistema ubica dins d'aquesta classe informació diversa.

A continuació s'expliquen breument alguns dels camps d'aquesta classe donada la seva importància per entendre de quina manera es processa la informació:

Header (capçalera): És un indicador del tipus de missatge que s'està enviant. Aquests són els valors constants possibles:

CLIENT_LOGIN : Missatge de petició d'entrada al sistema
CLIENT_LOGOUT : Missatge de finalització de connexió
CHAT : Informació de conversa
SERVER_DOWN : Finalització del servidor
USUARIS_LIST : Enviament de llista d'usuaris
CANVI_ESTAT : Canvi de l'estat d'un usuari
USER_OK : Resposta a CLIENT_LOGIN: usuari acceptat
USER_KO : Resposta a CLIENT_LOGIN: usuari denegat
SYNCRO : Sincronització utilitzada per detectar caigudes dels usuaris

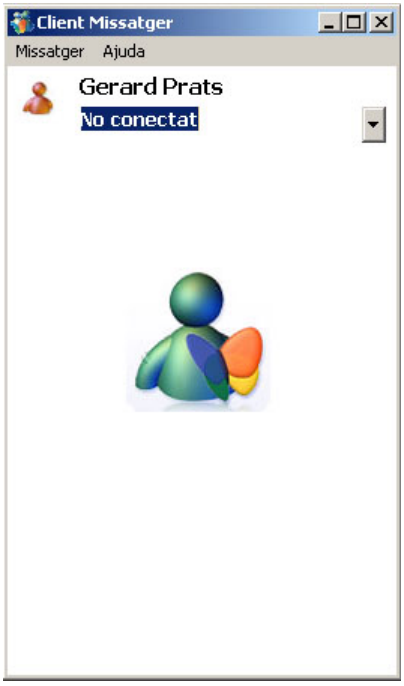
Email: Indicador de l'usuari que envia el missatge (només per missatges de client)
Desti: Indicador de l'usuari que ha de rebre el missatge (missatges tipus CHAT)
Usuari: Objecte Usuari amb tota la informació (missatges tipus CLIENT_LOGIN)
Missatge: Contingut del missatge (missatges tipus CHAT)
LlistaUsuaris: Llista dels usuaris del client (missatges tipus USUARIS_LIST)

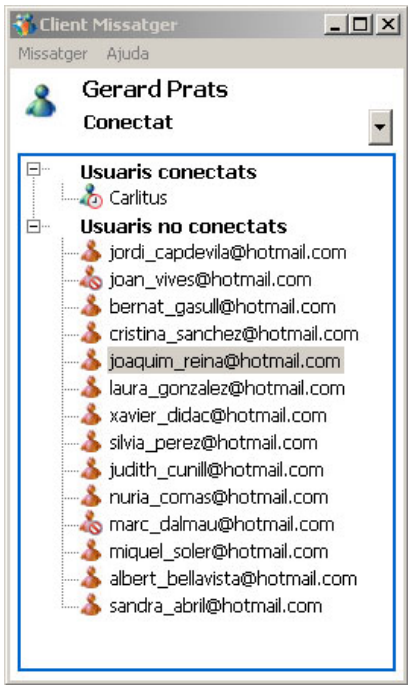
Pel que fa a l'establiment de converses, el servidor missatger actua com a pont entre cada parella d'usuaris. Això vol dir que s'encarrega de processar tots els missatges que arriben amb una capçalera diferent de CHAT. Per aquests últims, únicament s'encarrega de cercar la connexió que hi ha oberta amb el destí i enviar-li el missatge. Cal tenir en compte que aquests missatges, quan passen pel servidor, tenen el contingut del camp Missatge xifrat. Això vol dir que és inintel·ligible pel servidor i que es manté la privacitat dels usuaris que estan parlant.

La connexió es perd quan el client envia un missatge amb la capçalera CLIENT_LOGOUT al servidor i, aquest, ho fa constar a tots els usuaris de la seva llista mitjançant un altre missatge CLIENT_LOGOUT.

Client Missatger

L'eina Client Missatger és força semblant, almenys en aparença, a les esmentades a l'inici d'aquestes especificacions. La interfície gràfica, verticalment allargada, permet mostrar un llistat dels contactes i ofereix un menú per configurar l'aplicació i executar diferents opcions:



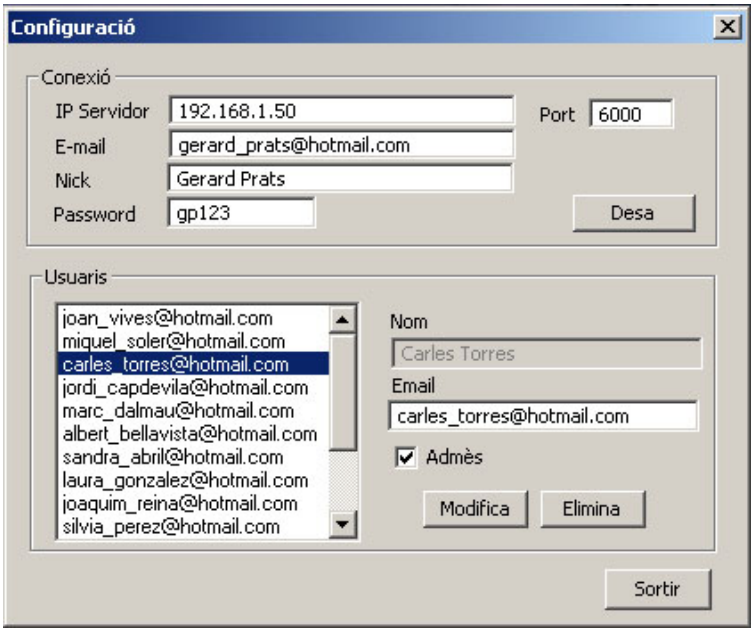


Pel que fa a la interfície gràfica, és important tenir en compte que per familiaritat i facilitat d'accés s'han utilitzat com a mostra les icones de l'aplicació de missatgeria de Microsoft, però en un hipotètic cas de comercialització del programa caldria dissenyar-ne unes de pròpies. La finestra principal del missatger pot redimensionar-se o maximitzar-se a gust de l'usuari, donat que el seu contingut s'hi adapta perfectament.

A la imatge de la dreta es pot observar també de quina manera el Client Missatger mostra a l'usuari el seu estat i el dels diferents usuaris de la seva llista.

Concertar-se: Estableix la connexió amb el servidor missatger
Desconnectar: Envia la petició de desconnexió al servidor missatger
Configuració: Obre la finestra de configuració
Sortir de Missatger: Surt de l'aplicació

La opció *Configuració* dóna accés a la pantalla on es poden indicar les dades de connexió amb el servidor, així com canviar el nick utilitzat i els usuaris de la llista.



Aquesta pantalla s'alimenta del següent fitxer:

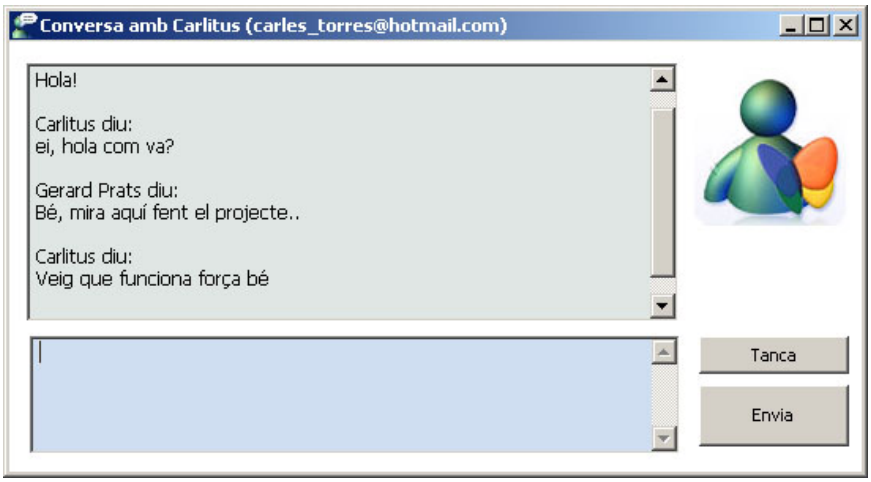
- *usuaris.xml* [document de configuració local]


```
<ConfiguraClient>
      <Conexio>
            <IPServidor>192.168.1.3</IPServidor>
            <Port>6000</Port>
            <Nick>Joan</Nick>
            <Email>joan_vives@hotmail.com </Email>
            <Contrassenya>jv234</Contrassenya>
            <PublicKey>MIHgMIGXBgkqhkiG9w0BAwEwgYkCQQDUVF4LadvtmEun
      BRbPLVgQPbcuOKKr1bqTT3uYhhl08Gb+TmnJ2foQ1RWYPrLzfMGJtEgycj2bO
      rrJOv8eKPZZAkApL3OpJWsgxjimPh3v8nUoQKFvM/Bz6m4qS8kzG+IrK9kHDp
      3vFUNUitIxhLL/jo0CDIRLLt+xPzyV+HCuvN2iAgIB/wNEAAJBAIAc7Wd30Cb
      NYVVIrgkreHeskN/I1LPCWrOHOfD5OwaFWj7KJAwOFET+y1WoY2/kAt/HrY7r
      p0WxurZhXM3YJBE=</PublicKey>
            <PrivateKey>MIHhAgEAMIGXBgkqhkiG9w0BAwEwgYkCQQDUVF4Ladv
      tmEunBRbPLVgQPbcuOKKr1bqTT3uYhhl08Gb+TmnJ2foQ1RWYPrLzfMGJtEgy
      cj2bOrrJOv8eKPZZAkApL3OpJWsgxjimPh3v8nUoQKFvM/Bz6m4qS8kzG+IrK
      9kHDp3vFUNUitIxhLL/jo0CDIRLLt+xPzyV+HCuvN2iAgIB/wRCAkBn4eNwe9
      GEIUBnbqI9YD1M8QtSeks20QZYnqHtkrSTybFm0Wp1pn3Ct8WoC7NDOjcqCr6
      6wRThmFlILfmgP97O</PrivateKey>
      </Conexio>
      <Usuaris>
            <Usuari>
                  <Nick>Joan Vives</Nick>
                  <Email>joan_vives@hotmail.com</Email>
                  <Admes>0</Admes>
                  <PublicKey>MIHgMIGXBgkqhkiG9w0BAwEwgYkCQQDUVF4Ladv
            tmEunBRbPLVgQPbcuOKKr1bqTT3uYhhl08Gb+TmnJ2foQ1RWYPrLzfM
            GJtEgycj2bOrrJOv8eKPZZAkApL3OpJWsgxjimPh3v8nUoQKFvM/Bz6
            m4qS8kzG+IrK9kHDp3vFUNUitIxhLL/jo0CDIRLLt+xPzyV+HCuvN2i
```

```
                AgIB/wNEAAJBAIXb0BZ0P4JA3+3JGFIbd3Y8AKv4I2DN5T2Bdy4VLVv
                1D6cuP7i64uKmSUm0Er76RetaAFd2RC+DyA3m010LgK8=</PublicKe
                y>
                </Usuari>
                ....   [més usuaris que aquí no es mostren]
        </Usuaris>
    </ConfiguraClient>
```

Encara que no es poden gestionar desde la pantalla de configuració, el fitxer usuaris.xml conté les claus públiques dels contactes i la clau pública i privada de l'usuari de l'aplicació. Es pot observar a més que aquestes claus estan codificades en base 64, donat que és una codificació que ofereix moltes més garanties a l'hora d'emmagatzemar les dades en fitxers i enviar-les a través de sockets. La resta d'informació és gestionable desde la pantalla que s'ha mostrat a la plana anterior.

Pel que fa a la finestra de conversa, s'hi accedeix amb un simple *doble-click* sobre qualsevol dels contactes connectats que hi hagi a l'aplicació.



L'usuari pot obrir tantes finestres de conversa com vulgui, sempre i quan sigui amb usuaris diferents. Quan un usuari escrigui a un altre i aquest no tingui la finestra de xat oberta, ja sigui perquè no havien parlat abans o perquè l'ha tancada, l'aplicació crearà una nova finestra de conversa per mostrar el missatge.

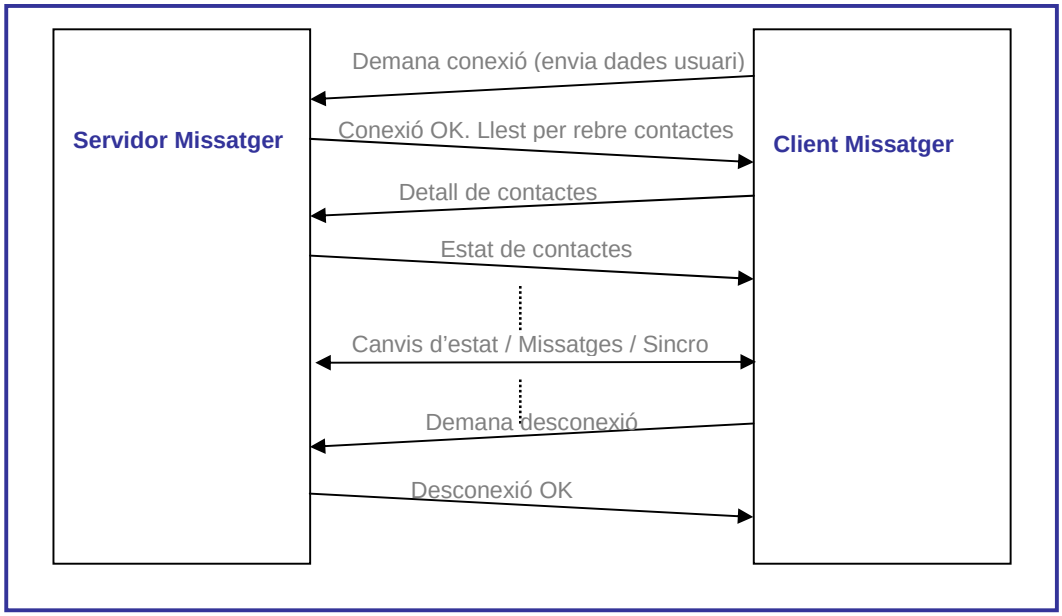
Interacció Client-Servidor

Pel que fa a la comunicació del client amb el servidor de missatgeria es segueix l'esquema que es mostra a la plana següent. Bàsicament es tracta d'un procés que es pot dividir en 7 fases. Les 4

primeres fan referència a l'establiment de la connexió, la 5a als processos que es duen a terme amb la connexió establerta i les 2 últimes

1. **Establiment de la connexió**: El Client demana la connexió al servidor i li envia un missatge que conté les seves dades d'usuari, inclosos l'e-mail i la contrasenya per identificar-se.
2. **Acceptació/denegació**: El servidor respon OK o KO a la petició de l'usuari.
3. **Llista d'usuaris**: En cas d'haver-se acceptat l'usuari, aquest envia la seva llista de contactes al servidor, informant dels seus e-mails i el seu estat (admès/no admès).
4. **Estat d'usuaris**: El servidor retorna la llista completa d'usuaris rebuda amb els seus estats corresponents per tal d'inicialitzar la finestra de contactes.
5. **Sincronització/Enviaments**: El servidor envia cada 10s un missatge SYNCRO que ha de ser respost pel client per verificar que es manté connectat. Entre sincronitzacions client i servidor s'envien actualitzacions dels estats propis i dels altres usuaris, així com missatges tipus CHAT entrants i sortints.
6. **Tancament de connexió**: El client envia un missatge CLIENT_LOGOUT indicant que tanca el socket. El servidor respon donant de baixa l'usuari i informant a tots els seus contactes.
7. **Confirmació del tancament**: El servidor retorna un missatge de tancament per confirmar al client que ha tancat la connexió.

Diagrama d'interacció client/servidor



Interacció Client-Client

Com ja s'ha introduït en aquest capítol, la interacció entre dos clients es fa utilitzant el servidor com a pont. Tots aquells missatges que porten la capçalera CHAT travessen el servidor sense processament, ja que aquest l'únic que fa és enviar-los al destinatari que porten indicat.

En realitat el programa no realitza un procés d'establiment o tancament de conversa ja que, donats els requeriments, no era estrictament necessari. Les aplicacions clients s'envien missatges de conversa sense preocupar-se de si l'usuari remot té o no la finestra oberta. Al arribar el missatge al destinatari, l'aplicació verifica si hi ha una finestra de xat amb aquell usuari i la crea en cas de no ser així.

D'altra banda, la interacció client-client toca de ple els aspectes de la seguretat que implementa el sistema. Com s'ha vist a la configuració del client, aquest incorpora una agenda de claus públiques dels contactes. És aquesta agenda la que permet obviar el procés d'establiment de connexió ja que es tracta de claus generades amb l'algorisme Diffie-Hellman, en base a uns valors secrets que només el servidor coneix. Així doncs, el que interessa ara en aquest apartat és conèixer de quina manera es posa la informació de la conversa en un medi segur.

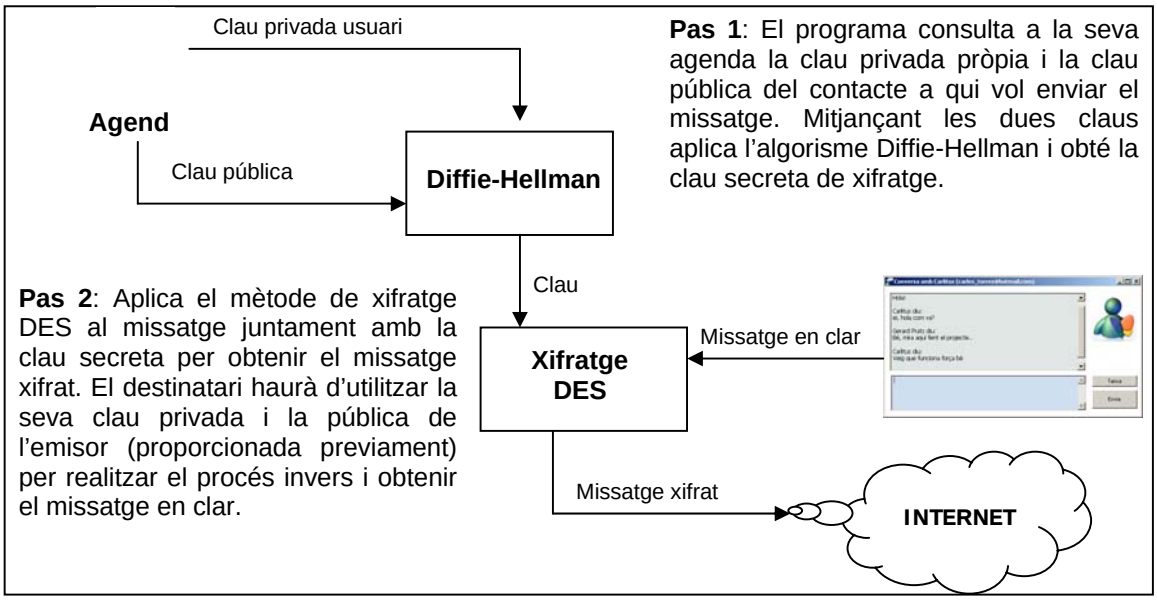
El programa utilitza un mètode de xifratge prou comú i conegut com és el DES. Aquest mètode requereix d'una clau secreta i un missatge d'entrada per generar una sortida xifrada. En el sistema del Missatger, tots dos extrems utilitzen la mateixa clau secreta ja que es tracta d'un sistema de xifratge simètric. Aquesta clau secreta ha de ser única per a cada parella d'usuaris i això s'aconsegueix mitjançant l'ús de les claus públiques i privades i l'algorisme Diffie-Hellman.

El procés és el següent:

1. **Generació del secret**: El client que vol enviar el missatge agafa la seva clau privada i la clau pública del destinatari que té emmagatzemada a l'agenda. Aplicant l'algorisme Diffie-Hellman obté la clau secreta compartida.
2. **Xifratge**: El client xifra el missatge sortint amb l'algorisme DES i la clau secreta generada.
3. **Enviament**: La informació s'envia al destinatari

De manera Inversa, a l'aplicació del destinatari succeeix el següent:

1. **Recepció**: El destinatari rep el missatge xifrat de l'emissor
2. **Generació del secret**: L'aplicació agafa la seva clau privada i la clau pública de l'usuari que li ha enviat el missatge, cercant-la a la seva agenda. Mitjançant l'algorisme Diffie-Hellman obté la clau secreta compartida, que ha de ser la mateixa que la que ha generat l'emissor.
3. **Desxifratge**: Mitjançant la clau generada i el missatge xifrat entrant, aplica la funció de desxifratge de l'algorisme DES, obtenint el text en clar del missatge.



Aspectes a desenvolupar en un futur

En aquest apartat de disseny s'ha detallat la manera en que l'autor ha dut a terme el desenvolupament del sistema en base a uns requeriments inicials. Aquests han estat donats parcialment pel document de requeriments del TFC, encara que l'alumne ha considerat oportú l'aportació d'una sèrie d'afegits. Malgrat això, hi ha tota una sèrie de millores o possibilitats que no s'han inclòs al programa senzillament per manca de temps. En alguns casos s'ha arribat a considerar la possibilitat de fer-les i fins i tot s'ha invertit temps en investigar de quina manera es podrien implementar.

Per aquest motiu es detallen a continuació algun d'aquests aspectes que coincideixen amb el l'usuari considera que són les principals mancances de l'aplicació.

- Gestió completa del fitxer Usuaris.xml desde la finestra de Configuració. Encara que es tracta d'una pantalla molt funcional i útil per l'usuari, no és possible gestionar al 100% les dades del fitxer. Així doncs mancaria oferir la possibilitat d'afegir nous contactes, així com editar les dades que fan referència a les claus públiques i privades tant de l'usuari com dels contactes.
- Possibilitat de canviar el Nick de l'usuari local i l'estat admissió/no admissió dels contactes quan ja hi ha una connexió establerta. Actualment el programa bloqueja qualsevol canvi en la configuració una vegada s'ha connectat amb el servidor.
- La finestra de xat, que és potser la que presenta més mancances en el Client Missatger, hauria d'incorporar la gestió de text amb format, és a dir, podent establir fonts, mides i colors personalitzats per a cada usuari. Això s'havia estudiat durant el desenvolupament de l'eina però es va descartar al veure que requeria utilitzar certs components especials per tractar el text com a HTML i no es requeria de prou temps. Mitjançant aquest tractament també cabia la possibilitat d'utilitzar *emoticonos* semblants als del Messenger.
- Un altre aspecte que es va considerar i que no hagués estat massa costós d'afegir és la gravació de logs de les converses en fitxer pla de text o bé en format XML.
- També hi hauria la possibilitat d'implementar un petit servei d'avís de missatges que no han arribat al destinatari. Així, quan un dels dos usuaris d'una conversa perdés la conexió l'altre sabria exactament quins han estat els missatges que no han arribat a l'altra persona.
- Finalment, també es podria sense massa dificultat implementar la possibilitat de que un usuari estigués conectat em mode invisible, és a dir, podent veure a la resta d'usuaris però sense que aquests el puguin veure a ell.

4. ASPECTES CONCRETS DE LA IMPLEMENTACIÓ

Planificació

La planificació s'ha realitzat tenint en compte la descomposició de tasques que s'indica al dossier del Treball Fi de Carrera. En total es disposava de 14 setmanes per dur a terme les 14 tasques indicades, començant la planificació el dilluns 3 d'octubre i finalitzant aquesta el dia 9 de gener, dia previ a l'entrega final del TFC.

A la següent taula es descriuen les diferents tasques i les dates de realització estipulades:

N	Descripció de la tasca	Realització
1	Cerca d'informació sobre l'algorisme Diffie-Hellman	03/10/05 a 09/10/05
2	Cerca d'informació sobre algorismes de xifratge simètric	03/10/05 a 09/10/05
3	Cerca d'informació sobre comunicació de processos a nivell de transport (sockets)	10/10/05 a 16/10/05
4	Cerca d'informació sobre llibreries de programació que permetin treballar amb Diffie-Hellman, DES, etc.	10/10/05 a 16/10/05
5	Disseny del programa a nivell de processos, comunicació i interacció client/servidor	17/10/05 a 23/10/05
6	Dissenyar la interfície gràfica d'usuari	24/10/05 a 30/10/05
7	Desenvolupar el programari, a alt nivell, amb comentaris i de forma descriptiva.	31/10/05 a 06/11/05
8	Implementació de la interfície gràfica	07/11/05 a 13/11/05
9	Afegiment del cos del programa, amb les rutines de comunicació	14/11/05 a 27/11/05
10	Afegiment de l'intercanvi de claus i del xifratge	28/11/05 a 11/12/05
11	Elaboració de la memòria: Objectiu del projecte, estat de l'art i recursos	12/12/05 a 18/12/05
12	Elaboració de la memòria: disseny del programa (amb diagrames senzills i descripció a alt nivell)	19/12/05 a 25/12/05
13	Elaboració de la memòria: Manual d'usuari, jocs de proves, incidències i conclusions	25/12/05 a 01/12/05
14	Realitzar la presentació del projecte amb PowerPoint	02/01/05 a 09/01/05

Segons aquesta planificació, les 14 setmanes de què s'ha disposat per dur a terme el treball es desglossen de la següent manera:

- 2 setmanes de ***recerca***
- 2 setmanes de ***disseny***
- 6 setmanes de ***desenvolupament***
- 3 setmanes per redactar la ***memòria***
- 1 setmana per la ***presentació*** PowerPoint

Cal tenir en compte també que tot software ha de ser provat i testejat de manera adequada. En aquest cas, les proves s'han realitzat paral·lelament al desenvolupament de la memòria, donat el curt temps de que es disposava per a la realització del projecte.

Llibreries de programació

Per implementar els diferents processos que gestionen la seguretat del programa, ha estat necessària la utilització d'unes llibreries especialitzades que suporten el sistema de comunicació i els algorismes que s'han descrit en apartats anteriors.

En aquest punt s'ha tingut en compte l'experiència prèvia de l'autor en el desenvolupament de software amb les *llibreries IAIK*, donat que aquestes van ser utilitzades anteriorment en un altre treball. Per aquest motiu van esdevenir les primeres a ser revisades i validades. Els resultats van ser positius, donat que es tracta d'unes llibreries força elaborades i amb tota mena d'opcionalitats. Se'n destaquen a continuació les més importants per al present treball:

- Reimplementen totalment la base del JCE 1.2 de SUN
- Permeten generar nombres aleatoris amb diferents algorismes, com és el cas del triple DES.
- Principals algorismes criptogràfics:
    - MD2, MD5, SHA-1, RIPEMD-160, ...
    - RSA, DSA, ECC, ESDH (Generació de claus Diffie-Hellman)...
    - DES, Triple DES, IDEA, Blowfish, RC2, RC5, RC6, AES, ...
- Generació i gestió de certificats X.509
- Generació i gestió de CRLs
- Treball amb estandars PKCS (#1, #5, #7, #8, #10, #12)

Per al disseny de la interfície gràfica també es va considerar necessària la utilització d'unes llibreries especialitzades o Plug-in per l'entorn de desenvolupament Eclipse, donat que aquest entorn, per defecte, no implementa cap suport per aquesta funcionalitat. En tot cas, desenvolupar una eina com aquesta de manera gràfica i sense un entorn adequat per realitzar el disseny no hagués estat gens recomanable.

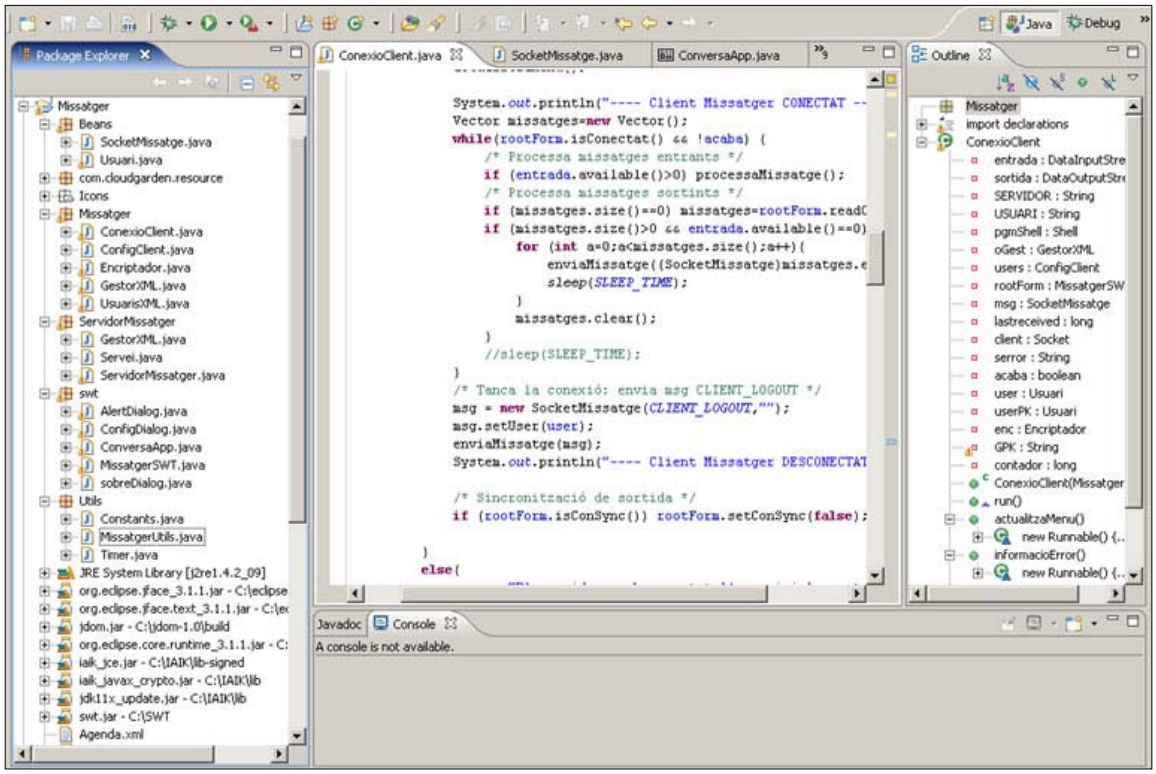
Es va optar per utilitzar el plug-in Jigloo GUI Builder, donat que l'autor hi té una mínima familiaritat i les seves característiques són adequades per la feina que s'ha de dur a terme:

- Plug-in de lliure utilització per fins no comercials
- Suporta tant SWT com SWING, incloent una conversió directa entre ambdues tecnologies i la possibilitat de utilitzar en *embedded* components d'una tecnologia dins de l'altra
- Llibreria totalment integrada amb Eclipse, un cop instal·lada només cal obrir una classe amb interfície gràfica per visualitzar-la en mode disseny.

Altrament, donat que els fitxers de configuració de les aplicacions es guarden en format XML, també és totalment necessari la utilització de la llibreria JDOM per Java. Aquesta llibreria implementa una jerarquia d'objectes que permeten llegir, processar i crear estructures XML que posteriorment es poden guardar a disc.

Enfocament i mètode seguit

El desenvolupament de la implementació s'ha fet, com s'indica a l'apartat de planificació, durant un període de 6 setmanes. Es tracta d'un període curt però suficient per dur a terme els requeriments mínims del sistema. A continuació es mostra una imatge de l'entorn de desenvolupament amb el projecte obert, a fi de comentar alguns aspectes generals de la implementació.



A l'àrea esquerra de la pantalla es troba l'explorador de paquets. És la part que permet veure de manera més global quina és l'estructura del projecte. El primer que cal dir és que les dues aplicacions, *Client Missatger* i *Servidor Missatger*, estan ubicades dins el mateix projecte. Quin sentit té això? Doncs bé, en realitat és totalment imprescindible donat que per poder enviar objectes a través d'una connexió *socket* totes dues aplicacions han d'estar utilitzant la mateixa classe (a dins del mateix projecte!). A més a més, això permet tenir una única versió d'aquestes classes compartides, evitant estar actualitzant contínuament d'un projecte a l'altre i tenint, malgrat això, sempre el dubte de si s'ha actualitzat l'última vegada o no. Aquestes classes compartides estan ubicades 2 dels paquets principals de l'aplicació: Beans i Utils.

A continuació es realitza un comentari breu de cada una de les classes de què consta el projecte, a fi d'entendre com està estructurat i de quina manera es relacionen cada una d'aquestes classes. Les classes es llisten agrupades en base als diferents paquets que les contenen:

- **Beans** : *Les classes contingudes en aquest paquet implementen estructures adequades per a emmagatzemar dades de les aplicacions.*
  SocketMissatge: Estructura de dades utilitzada per enviar informació a través dels sockets.
  Usuari: Estructura de dades que conté informació d'un usuari (email, contrasenya, estat).
- **com.cloudgarden.resource** : *Aquest paquet dóna suport a les llibreries SWT i no es comenta a nivell de classes perquè no ha estat desenvolupat amb el projecte.*
- **Missatger :** Aquí venen incloses les classes específiques de l'aplicació client que no són Beans ni Utils.
  ConexioClient: Classe tipus thread que s'encarrega de gestionar tot el referent a la connexió amb el servidor.
  ConfigClient: Estructura de dades que permet emmagatzemar tota la informació de la pantalla de configuració del client.
  Encriptador: Gestiona exclusivament el procés d'encriptació i desencriptació de missatges. Treballa amb la classe SocketMissatge.
  GestorXML: Processa el parsejat XML, és a dir, consulta i altera el document de configuració en format XML. Gestiona tota la informació d'entrada i sortida mitjançant les classes Usuari i ConfigClient.
- **ServidorMissatger :** En aquest paquet s'inclouen les classes específiques de l'aplicació Servidor que no són Beans ni Utils.
  GestorXML: Homòleg del GestorXML del paquet Missatger, però per l'aplicació servidor.
  Servei: Classe tipus thread que es genera per gestionar una connexió al servidor. És l'encarregada d'atendre a la classe ConexioClient.

ServidorMissatger: Classe principal (main) del servidor. Inicialitza el programa i manté el socket obert generant tants threads *Servei* com connexions entrants li arriben.

- **swt :** Com el seu nom indica, aquest paquet conté totes les classes que generen la interfície gràfica de l'aplicació client.

  AlertDialog: Finestra diàleg configurable que mostra els missatges d'error i informació

  ConfigDialog: Pantalla de configuració de l'aplicació client

  ConversaApp: Finestra de conversa que en realitat és una aplicació independent.

  MissatgerSWT: Classe principal del programa client que correspon a la finestra de contactes.

  sobreDialog: Finestra d'informació a la que s'accedeix pel menú d'ajuda.

- **Utils :** En aquest paquet s'hi troben les classes que tenen una utilitat molt específica i que donen suport a la resta de classes del sistema.

  Constants: Definició de constants utilitzades en el programa.

  MissatgerUtils: Conversió de Object a Bytes i Bytes a Object per a l'enviament de missatges a través de socket. L'utilitzen tant el servidor com el client.

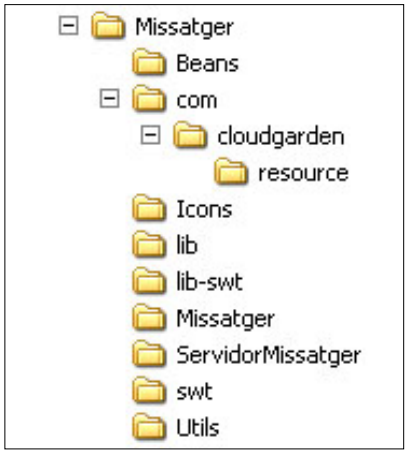
  Timer: Típica classe timer personalitzada per a l'ús del missatger.

Com es pot concloure amb la lectura d'aquesta estructura, s'ha intentat aprofitar al màxim les capacitats del llenguatge Java per organitzar i reutilitzar de la millor manera possible els diferents components del projecte, assolint en aquest sentit un nivell força satisfactori.

5. MANUAL D'INSTAL·LACIÓ

El sistema de missatgeria que es presenta en aquest treball ve empaquetat dins d'un fitxer comprimit ZIP anomenat "Missatger.zip". Dins d'aquest arxiu es troben els dos programes de què consta el sistema, és a dir, l'aplicació client i l'aplicació servidor.

A la imatge de la figura es pot observar el contingut d'aquest arxiu, una vegada s'ha descomprimit, organitzat en carpetes. Les carpetes *Beans*, *com*, *Missatger*, *ServidorMissatger*, *swt* i *Utils* contenen arxius d'extensió .class. És a dir, són els arxius llenguatge Java del programa en versió compilada. En cas de requerir les fonts en versió .java d'aquests arxius haurem de consultar el fitxer "Compilacio.zip", que té una estructura semblant però incorpora tots els fitxers de codi font i els arxius de configuració per editar-lo mitjançant Eclipse.



A la carpeta Icons s'hi troben diferents imatges JPG i GIF. Es tracta d'icones que el programa utilitza i que cal que es mantinguin en aquest directori. Finalment, dins de *lib* i *lib-swt* trobarem les diferents versions de les llibreries que incorporen els programes.

Name	Size	Type	Date Modified
iaik_javax_crypto.jar	28 KB	Executable Jar File	08/02/2005 17:49
iaik_jce.jar	666 KB	Executable Jar File	08/02/2005 17:49
jdk11x_update.jar	25 KB	Executable Jar File	07/02/2003 18:36
jdom.jar	150 KB	Executable Jar File	09/09/2004 16:00
org.eclipse.core.runtime_3.1.1.jar	444 KB	Executable Jar File	29/09/2005 9:03
org.eclipse.jface.text_3.1.1.jar	656 KB	Executable Jar File	29/09/2005 9:03
org.eclipse.jface_3.1.1.jar	653 KB	Executable Jar File	29/09/2005 9:03

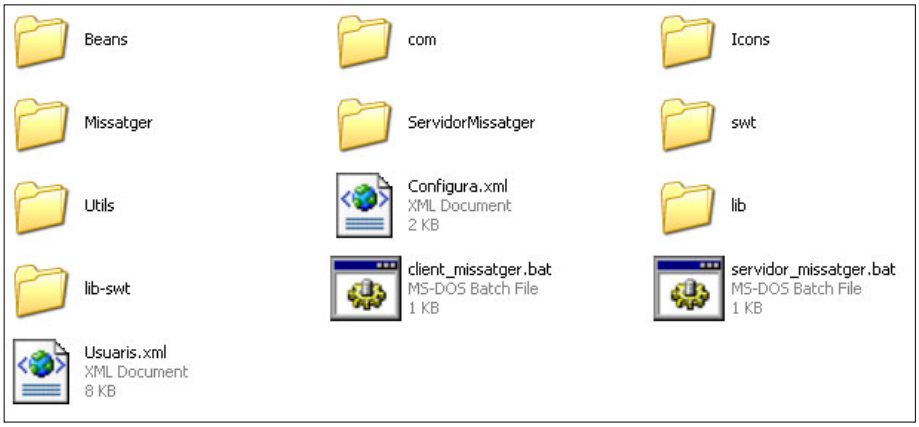
A la figura superior s'indiquen els fitxers que es troben dins el directori *lib*. Es tracta, com es pot veure, de llibreries IAIK, JDOM i Eclipse necessàries per a l'execució correcta el programa. D'altra banda, les llibreries referents a SWT s'han ubicat al directori lib-swt, donat que se s'ha utilitzat una versió actualitzada no inclosa amb Eclipse i requereix un tractament separat.

A continuació es mostra el contingut de la carpeta lib-swt.

Name	Size	Type	Date Modified
.classpath	1 KB	CLASSPATH File	29/09/2005 10:05
.project	1 KB	PROJECT File	29/09/2005 10:05
about.html	2 KB	HTML Document	29/09/2005 10:05
javaw.exe.manifest	1 KB	MANIFEST File	29/09/2005 10:05
src.zip	1.319 KB	WinZip File	29/09/2005 10:05
swt.jar	1.253 KB	Executable Jar File	29/09/2005 10:05
swt-awt-win32-3139.dll	32 KB	Application Extension	29/09/2005 10:05
swt-gdip-win32-3139.dll	68 KB	Application Extension	29/09/2005 10:05
swt-win32-3139.dll	308 KB	Application Extension	29/09/2005 10:05

Per a la correcta instal·lació del sistema és necessari descomprimir l'arxiu subministrat a la carpeta arrel del disc dur (*C:*\). D'aquesta manera el programa queda ubicat automàticament a dins de la carpeta *C:\Missatger*. Aquesta ubicació no és estrictament necessària per poder utilitzar el programa, però en cas de modificar-la caldrà modificar també els arxius .bat que s'encarreguen d'executar les aplicacions. Per aquest motiu és altament recomanable no realitzar cap canvi.

Aquest és ara el contingut dels arxius executables a la carpeta *C:\Missatger* :

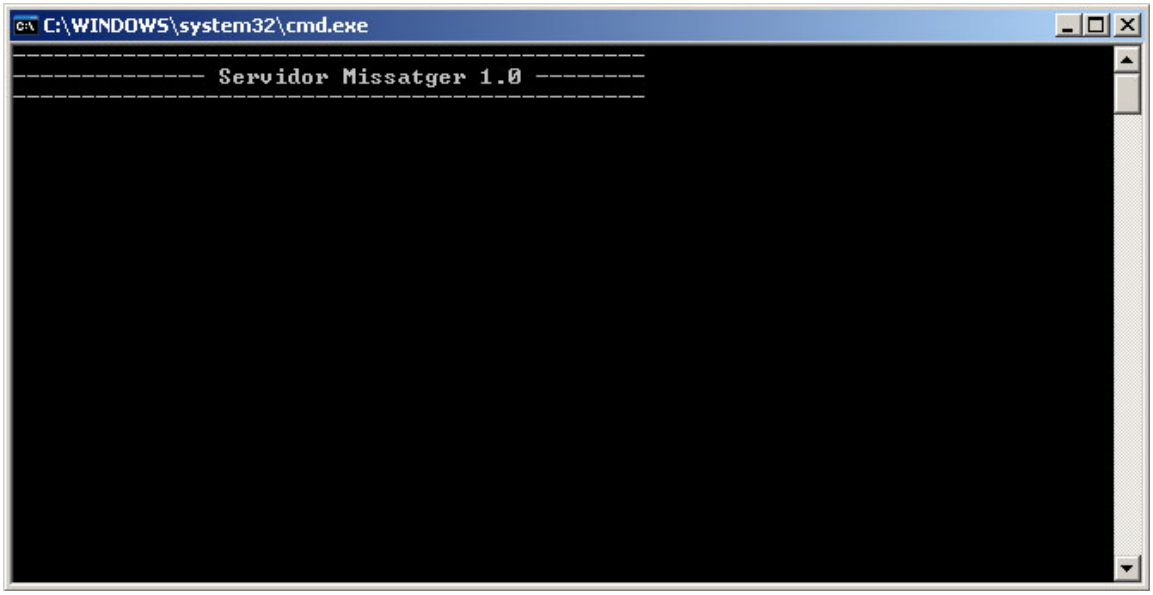


Com ja s'ha fet constar anteriorment a l'apartat de disseny de l'aplicació, els fitxers Usuaris.xml i Configura.xml contenen la configuració del software client i servidor respectivament. S'ha de tenir en compte que aquest fitxers inclouen una configuració inicial que no està en blanc, és a dir, hi ha un total de 16 comptes d'usuari donats d'alta al servidor. A la configuració del client, per la seva part, hi trobarem un d'aquests usuaris configurat com a usuari principal del programa, fent-hi constar la seva clau pública i privada. Pel que fa a la resta d'usuaris, venen donats d'alta com a contactes d'aquest primer, incloent les seves claus públiques per poder-s'hi comunicar.

Per facilitar les proves amb el sistema existeix un fitxer Agenda.xml que ve inclòs a la versió en codi font “Compilacio.zip”. En aquest fitxer hi figura una llista de tots els usuaris que venen configurats i les seves respectives claus públiques i privades, a fi de poder configurar un Client Missatger per cada un d’ells si fos necessari.

D’altra banda, els fitxers *client_missatger.bat* i *servidor_missatger.bat* són arxius d’execució per lots que criden la sentència java adequada per posar en funcionament cada una de les versions del software. Així doncs, si tots els arxius estan ubicats al seu lloc en tenim prou amb executar aquests fitxers per posar en marxa els programes.

Pel que fa al servidor, aquesta és la finestra que s’obté al iniciar-lo.



En aquesta pantalla s’anirà incloent informació de les diverses connexions que el servidor pugui rebre de les aplicacions clients. Bàsicament ens informarà dels usuaris que es connecten amb èxit, dels que es connecten però se’ls denega l’entrada i dels que surten del servidor. Pel que fa al client s’obre una finestra de MS-DOS en blanc i la pantalla principal de l’aplicatiu segons s’ha pogut veure al capítol de disseny.

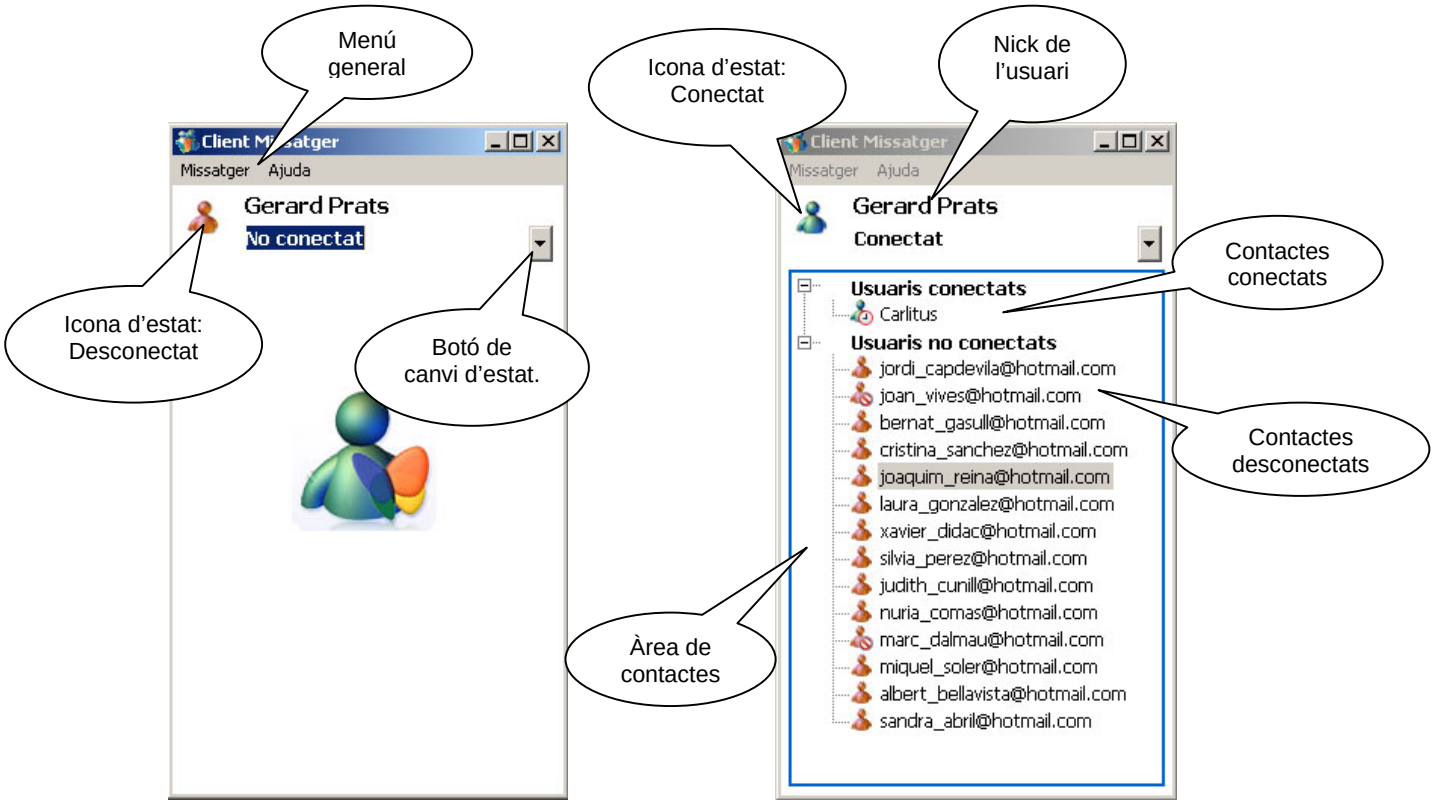
En qualsevol dels dos casos cal tenir en compte que si es tanca la finestra MS-DOS s’aturarà l’execució del software.

6. MANUAL D'USUARI

En aquest apartat es desenvolupa una petita guia orientada a l'usuari final, és a dir, a aquella persona que es pot imaginar descarregant una versió comercial del Client Missatger desde una web i instal·lant-lo en el seu PC amb l'intenció d'iniciar converses amb altres usuaris. En aquest sentit, cal fer notar que l'aplicació no incorpora encara totes les funcionalitats que serien necessàries perquè aquest procés es portés a terme de la manera mes senzilla possible, però tot això és justificat per una òbvia limitació de temps en el desenvolupament del programa. Malgrat tot, amb una petita inversió extra de temps resultaria senzill adaptar tots aquests processos per guanyar un nivell de "amigabilitat" de l'entorn considerable.

Finestra principal del Missatger

A continuació es mostra una figura amb la pantalla principal de l'aplicació i els diferents elements a destacar-hi.



La pantalla principal ofereix informació sobre l'estat de l'usuari (inicialment desconnectat), així com el seu nick segons consta a la configuració del programa. Ofereix, al mateix temps, la possibilitat d'interactuar amb el menú general (superior) o amb el botó de canvi d'estat.

Pel que fa a aquest últim, hi ha 2 possibilitats:

- Si l'usuari està desconnectat oferirà, al prémer sobre el botó, la possibilitat de seleccionar *Desconnectat* ( ) o *Connectat* ( ). Seleccionant la segona opció el *Client Missatger* es connectarà amb el servidor. Les icones mostrades al costat dels estats són les que el programa mostrarà a la icona d'estat.
- Si ja s'ha establert la connexió, a més de les opcions *Desconnectat* i *Connectat* n'oferirà 2 més: *Ausent* ( ) i *Ocupat* ( ). Aquestes dues noves alternatives responen a la possibilitat de mostrar als contactes 3 estats diferents de connexió, segons la disponibilitat de l'usuari.

Menú de l'aplicació

Si ens centrem en el menú general, desplegant la opció *Missatger* tenim les següents alternatives:

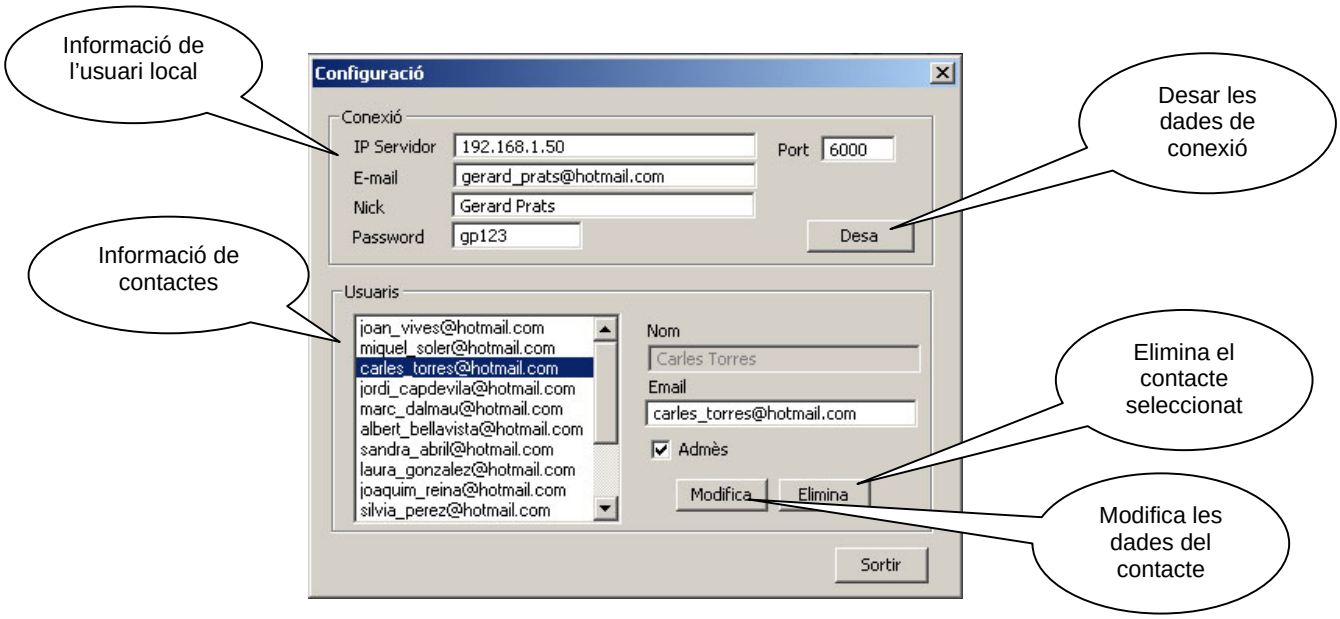
Concertar-se: Mitjançant aquesta opció el programa farà exactament el mateix que quan es canvi l'estat mitjançant el botó de canvi d'estat. **Desconnectar**: Només està habilitat quan hi ha una connexió establerta amb el servidor. Permet, lògicament, tancar aquesta connexió i tornar a l'estat inicial del programa. **Configuració**: Obre la finestra de configuració del programa. Les opcions d'aquesta pantalla s'aprofundeixen en el proper subapartat d'aquest manual. **Sortir**: Desconnecta del servidor, en cas d'haver-hi una connexió establerta, i finalitza el programa. Es pot obtenir el mateix resultat prement el botó "x" a la cantonada superior dreta de la finestra.

Pel que fa al menú *Ajuda*, aquest desplega una única opció **Informació**, mitjançant la qual s'obre una finestra amb informació sobre l'aplicació i la seva versió.

Altrament, si l'usuari ho desitja es pot redimensionar la finestra principal del programa de la manera que es vulgui, maximitzant-la si així ho desitja l'usuari per ocupar la totalitat de la pantalla, ja que els elements interiors s'adaptaran al mida total.

Configuració del Missatger

A través de la opció Configuració del menú Missatger accedim a la finestra que ens permet configurar els diferents paràmetres del programa:



Aquesta pantalla es divideix en dues parts. A la primera, delimitada pel frame **Connexió**, hi trobem les dades pròpies de l'usuari local. Aquestes dades permeten la connexió al servidor de missatgeria i aporten informació necessària com el password per identificar-se o bé el nick a mostrar. Si canviem el contingut del camp *Nick*, per exemple, i premem el botó **Desa** observarem, al tancar la finestra, que s'ha actualitzat el nick a la finestra principal del Missatger.

Per una altra banda hi ha el frame **Usuaris**. Aquesta àrea de la finestra permet accedir a la llista d'usuaris (e-mails) que figuren a la pantalla de contactes i modificar-ne les adreces o bé l'estat d'admissió. Cal tenir en compte que si desmarquem la casella **Admès** per algun contacte aquest ens veurà sempre en estat de no connectat a la seva llista. Per efectuar qualsevol modificació només cal seleccionar un usuari (e-mail) a la llista de l'esquerra perquè s'actualitzin les dades a la part dreta del frame. En aquest moment se'ns ofereix la possibilitat de modificar l'e-mail i la marca

d'admissió prement el botó Modifica. Altrament també podem donar de baixa el contacte prement **Elimina**.

En finalitzar totes les operacions, només cal prémer el botó **Sortir** per tancar la finestra.

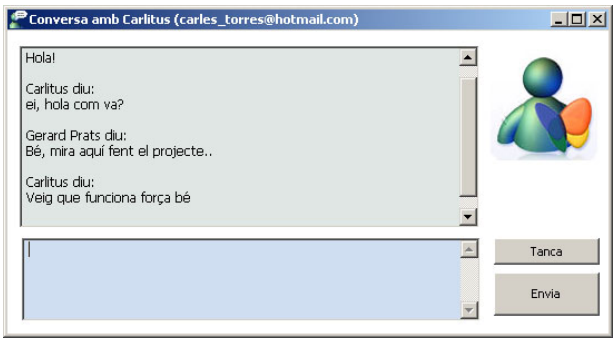
Llista de contactes

Ens centrem ara a la pantalla principal de l'aplicació, un cop l'usuari està connectat:

Tal i com es senyalava a la primera plana d'aquest manual, els contactes apareixen organitzats en dos grups: connectats i no connectats. A més d'això, el Missatger informa, mitjançant les diferents icones a les que s'ha fet referència anteriorment, de l'estat dels usuaris connectats (connectat, ausent o bé ocupat) mitjançant una icona a davant del nom. També mitjançant aquesta icona podem veure quins contactes estan configurats en estat de no admesos (/). Aquests contactes no poden obrir conversa amb l'usuari, donat que no el visualitzen com a connectat. No obstant, l'usuari sí que té possibilitat de fer-ho.

La finestra de contactes s'actualitza cada vegada que hi ha un canvi en l'estat d'algun dels usuaris, així que es pot veure en temps real com canvien aquests estats i quins són els usuaris que entren i surten del servidor.

Finestra de xat



Per a establir conversa amb algú de la llista de contactes cal fer un doble-click sobre un dels usuaris del grup Usuaris connectats. En aquest moment, encara que l'usuari seleccionat no veu res encara, s'obre una finestra com la que es mostra a la dreta. Existeixen en aquesta finestra 2 àrees de text, una superior i de mida més gran, sobre la que es va imprimint en temps real la conversa entre els dos usuaris, i una d'inferior una mica més petita, que és l'espai on l'usuari local ha d'escriure els missatges que vol enviar. Per a executar

l'enviament d'un missatge només cal prémer la tecla **ENTER** o bé fer click al botó **Envia**. Una vegada s'ha enviat un missatge, aquest apareix al requadre superior de la conversa.

Les finestres de conversa dels dos usuaris són totalment independents, és a dir, un dels dos pot tancar-la sense que l'altre ho adverteixi de cap manera. En aquest cas, si el que manté la conversa oberta envia un nou missatge l'altre usuari veu de nou com s'obre una finestra de xat per mostrar aquest missatge, com si fos una conversa nova.

Configuració d'usuaris i claus

El programa no incorpora en aquesta versió la opció per donar d'alta usuaris desde la pantalla de configuració. Per a fer-ho, cal tenir l'e-mail i la clau pública del contacte que es vol donar d'alta i editar directament el fitxer Usuaris.xml, ubicat al directori de l'aplicació.


```
<Usuari>
      <Nick>Joan Vives</Nick>
      <Email>joan_vives@hotmail.com</Email>
      <Admes>0</Admes>
      <PublicKey>MIHgMIGXBgkqhkiG9w0BAwEwgYkCQQDUVF4Ladv
tmEunBRbPLVgQPbcuOKKr1bqTT3uYhhl08Gb+TmnJ2foQ1RWYPrLzfM
GJtEgycj2bOrrJOv8eKPZZAkApL3OpJWsgxjimPh3v8nUoQKFvM/Bz6
m4qS8kzG+IrK9kHDp3vFUNUitIxhLL/jo0CDIRLLt+xPzyV+HCuvN2i
AgIB/wNEAAJBAIXb0BZ0P4JA3+3JGFIbd3Y8AKv4I2DN5T2Bdy4VLVv
1D6cuP7i64uKmSUm0Er76RetaAFd2RC+DyA3m010LgK8=</PublicKe
y>
</Usuari>
```


Aquest és un exemple de *tag* d'usuari en format típic *XML*. L'únic que cal fer per donar d'alta un usuari nou és modificar el contingut dels camps Nick, Email, Admes i PublicKey en aquest exemple i afegir-lo entre els tags <Usuaris>...</Usuaris> del fitxer *XML* mencionat.

7. PROVES

S'ha testejat el Client Missatger per assegurar-ne un bon funcionament mitjançant una sèrie de proves que s'han dividit en dues fases independents:

Proves individuals

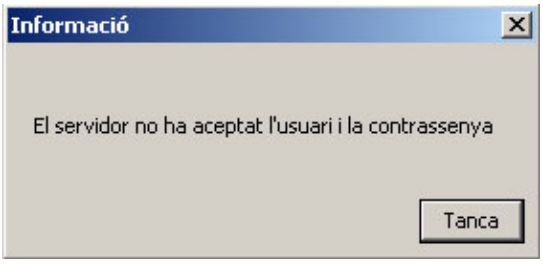
Es tracta de realitzar proves sobre una única aplicació de Client Missatger utilitzant en alguns casos la connexió amb el Servidor Missatger. Les àrees més afectades per aquesta bateria de proves són la pantalla de configuració, l'establiment de la connexió i els canvis d'estat de l'usuari.

Finestra de configuració

- S'ha comprovat que totes les caixes de text (textbox) estan limitades a un límit coherent de dades, per evitar qualsevol error del programa, bé sigui en el tractament d'aquestes dades o a l'hora de guardar-les sobre l'arxiu XML.
- S'ha verificat que tots els camps del frame Configuració queden actualitzats al prémer el botó desar.
- Al actualitzar el camp Nick i tornar a la finestra de contactes el nick queda correctament actualitzat.
- Es poden modificar els e-mails i estats d'admissió de diferents usuaris, alternant amb la eliminació d'alguns sense generar cap tipus de problema. Després d'això el Client missatger es connecta correctament mostrant els contactes correctament.

Establiment de connexió amb el servidor

- Si es canvia la contrasenya d'usuari per una altra a la pantalla de configuració i s'intenta connectar amb el servidor l'aplicació respon amb el següent missatge:

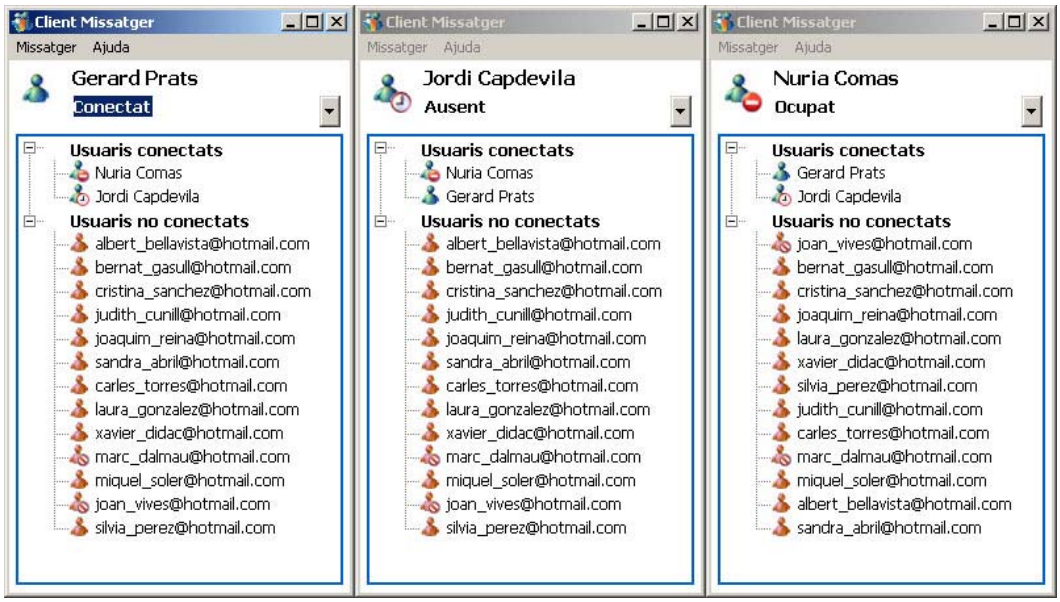


- A continuació, es pot tornar a la pantalla de configuració per restablir la contrasenya correcta. Al demanar que es connecti de nou ho fa sense problemes. Es pot verificar també que el servidor ha mostrat a la seva finestra la informació de l'intent de connexió sense èxit i continuat la seva execució.
- Si es modifica el port o la IP del servidor per una que no és correcta el programa queda en espera uns segons inhabilitant les diferents opcions del programa que hi puguin interferir, i acaba mostrant un missatge que informa de que no s'ha pogut trobar el servidor.
- Si es connecta el Client Missatger al servidor i es mata l'aplicació de manera sobtada el servidor ho detecta al cap d'uns segons i tanca la connexió perquè pugui tornar a entrar.

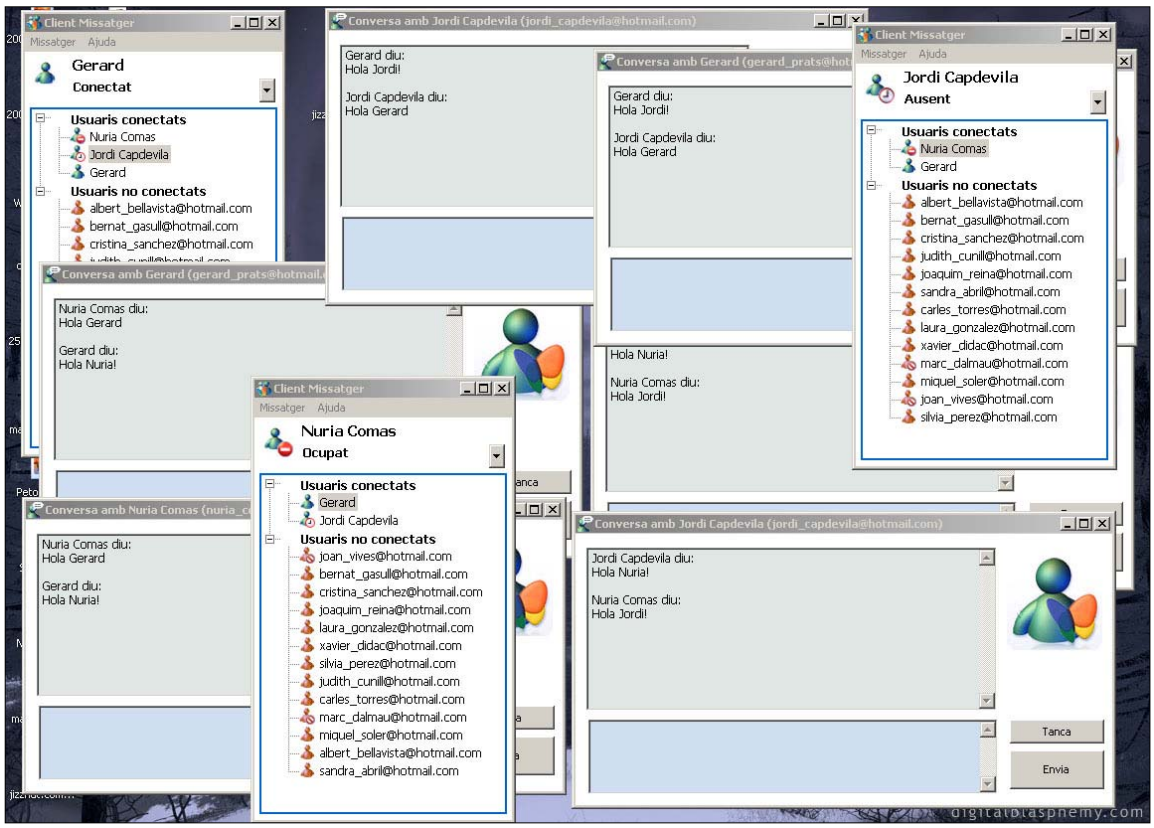
Jocs de proves en local amb varis usuaris

En aquest apartat de proves, a diferència de l'anterior, s'executen localment (en el mateix PC) 3 aplicacions Client Missatger amb les dades dels següents usuaris:
Gerard Prats (gerard_prats@hotmail.com)
Jordi Capdevila (jordi_capdevila@hotmail.com)
Nuria Comas (nuria_comas@hotmail.com)

La prova consisteix en connectar els tres usuaris de manera consecutiva al servidor i establir per a cada un d'ells un estat diferent (connectat, ausent i ocupat, respectivament). Un cop fet això s'obren converses de manera circular: Gerard Prats a Jordi Capdevila, Jordi Capdevila a Nuria Comas i Nuria Comas a Gerard Prats. Un cop fet això es tracta d'alternar l'enviament de missatges amb els canvis d'estat dels diferents usuaris, algun dels quals es desconnectarà i tornarà a entrar per reprendre les converses vigents.



A la mostra de pantalla anterior es pot veure com els tres missatgers han establert simultàniament la connexió i s'informen de manera correcta dels estats dels seus usuaris. A continuació es mostra l'escriptori obtingut al iniciar converses segons s'ha especificat a la plana anterior:



Un cop fet això es procedeix a continuar introduïnt missatges a les diferents converses de manera aleatòria per a garantitzar que en cap moment una comunicació es vegi afectada per una altra. D'altra banda, durant la realització d'aquesta operació es realitzen també canvis d'estat en els diferents usuaris.

Jocs de proves en remot

L'última etapa de testeig, una vegada s'ha comprovat que el software respon correctament a les proves anteriors, tracta de realitzar una connexió i una conversa a 3 o més de manera semblant a com s'ha fet al subapartat anterior, però aquesta vegada tenint les aplicacions clients a ordinadors diferents, cada un d'ells amb la seva adreça IP. D'aquesta manera es pot assegurar que no sorgeix cap problema quan la informació viatja realment a través de la xarxa internet.

8. VALORACIÓ ECONÒMICA

Com gairebé qualsevol software existent, el sistema de missatgeria presentat podria esdevenir un producte comercial amb certes possibilitats de mercat. Per aquest motiu resulta interessant analitzar-lo no només desde una perspectiva tècnica sinó també tenint en compte els costos que ha comportat el seu desenvolupament de manera específica, tant pel que fa als recursos humans necessaris com pel cost de les diverses eines que hagi calgut utilitzar.

A continuació s'ofereix un detall dels diferents costos que es podrien atribuïr al desenvolupament d'aquesta sistema:

Recursos humans

S'ha promitjat un total de 20 hores invertides per setmana. Com a recurs humà principal s'ha considerat el perfil d'un programador sènior, aproximant el seu cost de treball a jornada completa sobre uns 18.000 €/bruts anuals. Això fa que el seu cost de producció per l'empresa sigui d'uns 75 € el dia, és a dir, uns 9,37 € la hora. A continuació es detallen les hores invertides agrupades per tipus de tasca i el seu cost:

Tasca	Setmanes	Hores	Preu
Recerca d'informació	2	40	374,8 €
Disseny de l'aplicació	2	40	374,8 €
Proves i redactat documental	3	60	562,2 €
Presentació	1	20	187,4 €
TOTAL	8	160	**1499,2 €**

Així doncs, obtenim que podem desenvolupar la base de les aplicacions i la documentació en un total de 160 hores, que són exactament un mes de treball a jornada completa (40 hores setmanals) o bé dos mesos a mitja jornada. D'altra banda, el cost total de les gestions per part del treballador arriben gairebé als 1500 €.

Software

En aquest apartat es compta amb l'avantatge d'haver treballat amb algunes eines de lliure distribució. Així doncs, l'entorn de desenvolupament Eclipse i la llibreria JDOM poden ser utilitzades gratuïtament i no comporten cap cost pel projecte.

Hi ha, però altres elements que sí requereixen d'una compra per poder ser utilitzats amb fins comercials:

Software	Llicència	Preu	Llicència addicional
Jigloo SWT/Swing builder	Single-user	63,56 €	63,56 €
IAIK JCE SE	Basic License	500 €	100 €
TOTAL		**563,56 €**	

Així doncs, el software utilitzat suposa una inversió de 563,56 €, tenint en compte que es tracta de llicències per un sol usuari. Si calgués ampliar el nombre de persones que desenvolupen l'eina s'hauria d'abonar un total de 163,56 € extres per cada llicència nova.

Resum

La següent taula resumeix el total de despeses utilitzant la informació donada fins ara:

Tipus d'inversió	Total despeses
Recursos humans	1.499,2 €
Compra de software	563,56 €
TOTAL	**2.062,76 €**

La suma total aproximada de despeses específiques per desenvolupar el Client Missatger puja a un total que supera lleugerament els 2.000€ (unes 343.214 de les pessetes antigues). Evidentment es tracta d'un pressupost aproximat i totalment orientatiu, ja que no s'han tingut en compte molts altres aspectes, com podrien ser, per exemple, el disposar d'un ordinador de desenvolupament. Per aquest motiu les dades que s'ofereixen en aquest capítol són només una aproximació al cost o també al valor que pot tenir el treball realitzat, i al que caldria per engegar-lo de manera seriosa perquè esdevingués un producte comercial real.

Consideracions

No obstant això, és oportú fer dues consideracions finals. Per una part, que el cost total assolit és bastant petit, donat el nombre d'usuaris que poden arribar a utilitzar un sistema d'aquestes característiques. Realment hi ha aplicacions molt més complexes i costoses que un sistema de missatgeria, i aquest és un dels avantatges principals que tindria dur a la realitat aquest projecte. D'altra banda, però, existeix un impediment força evident. Existeixen al mercat diferents sistemes de missatgeria coneguts i tots ells són, a hores d'ara, completament gratuïts. Això vol dir que els ingressos que les empreses que els han desenvolupat obtenen provenen d'altres vies, com seria lògicament el cas de la publicitat. Això compensa en gran manera el petit cost que el desenvolupament de les eines ocasiona, ja que els ingressos indirectes que les aplicacions puguin provocar són petits comparats amb els que s'obtenen en una venta directa de software.

9. CONCLUSIONS

El present Treball Fi de Carrera ha esdevingut per l'estudiant una oportunitat idònia per afrontar un projecte informàtic de la manera més semblant possible a com caldria fer-ho desde dins d'una empresa comercial. S'han executat les tasques comunes més essencials en la feina de qualsevol programador professional, desde la recerca d'informació i recursos tècnics per als requeriments de l'aplicatiu, passant pel seu disseny i desenvolupament, fins el redactat de manuals del software i documentació diversa relacionada. Altres aspectes no tant tècnics com la realitat actual del mercat pel que fa a l'aplicació i les consideracions econòmiques del seu desenvolupament també han resultat importants de cara a la perspectiva que l'alumne ha assolit mitjançant aquest TFC.

Un dels aspectes que més diferencien aquest projecte de la resta de pràctiques i programes que s'han desenvolupat al llarg de les diverses assignatures dels estudis d'informàtica és el fet de no existir una guia que defineixi exhaustivament de quina manera s'ha d'implementar el sistema. Això vol dir que l'alumne es veu obligat per primera vegada a prendre decisions, és a dir, a escollir entre diferents possibilitats pel que fa als recursos utilitzats pel desenvolupament (tenint en compte, a més, el seu cost i els requeriments de llicència), així com pel disseny, el o els llenguatges de programació emprats, l'aspecte dels programes i les funcionalitats que acabarà tenint. Referent a tot això només es troben 3 petites consideracions al document de requeriments del TFC, que resulten bastant globals i tenen només la intenció de posar uns mínims per els projectes que es desenvolupen. Aquesta perspectiva del Projecte Fi de Carrera és molt important, ja que una assignatura que sovint fa de pont entre la finalització dels estudis i la incorporació al món laboral ha de proporcionar a l'estudiant una perspectiva el més pròxima possible al que aquest es pot trobar una vegada sigui contractat per alguna empresa com a programador.

Entrant més en els aspectes de desenvolupament del projecte, la planificació, que esdevé també un recurs essencial i indispensable de cara al futur professional de l'alumne, ha estat prou útil per mesurar en quin estat es trobava en cada moment el projecte i la realització de les diferents tasques. En realitat s'ha vist que és bastant difícil preveure amb molta precisió el temps requerit per a cada una d'elles, especialment les de programació que poden allargar-se de manera exagerada quan es presenta un problema que no es sap com resoldre, però es pot considerar que la planificació que es va fer inicialment era prou encertada. Tanmateix, si calgués fer-hi alguna modificació segurament s'hagués reduït una de les 3 setmanes destinades a la elaboració de la memòria per incorporar-les a la fase de desenvolupament, ja que aquesta última ha estat amb diferència la que més hores ha requerit.

Un altre aspecte rellevant ha estat el que fa referència a les prestacions finals del programa. El Client Missatger incorpora tots els requeriments que es demanaven al document de TFC, amb alguns afegits que l'alumne, per cost de temps i utilitat, ha considerat més viables.

En primer lloc s'ha optat per crear una finestra de configuració que permet gestionar els paràmetres del programa sense haver d'anar a obrir directament el fitxer de configuració. A més d'això, el programa incorpora la possibilitat de configurar l'admissió i no admissió dels diferents contactes, així com la possibilitat de canviar l'estat de l'usuari i ser detectat pels contactes com a usuari ocupat o ausent. Aquests són, principalment, els afegits que se li han donat al Client Missatger per dotar-lo d'una certa funcionalitat extra respecte els requeriments.

Altrament, però, el programa també incorpora altres característiques més secundaries, però no per això menys interessants, com són el tractament de fitxers de configuració en format XML, la possibilitat de canviar el Nick de l'usuari, una finestra diàleg estàndard per avisar dels diferents errors del programa o les característiques visuals de la pantalla principal, que es pot redimensionar a gust de l'usuari per aprofitar millor l'espai. Per tots aquests aspectes comentats, en termes generals es considera que les funcionalitats donades al sistema són considerables, tenint en compte el temps de què es disposava per desenvolupar-lo.

Finalment, cal fer mencció també a la realització de la presentació en Powerpoint del projecte, ja que requereix d'una síntesis molt precisa de tot el que s'ha dut a terme en el projecte i esdevé un pas més pel que fa a l'acostament a l'àmbit professional de la informàtica. És important saber què és important i què no quan es presenta un projecte a un client o potser a un responsable de la pròpia empresa, seleccionar els diagrames o les imatges més rellevants i properes a l'usuari, saber fugir dels excessius tecnicismes i donar un aspecte seriós i coherent a tota la presentació.

Concloent amb una consideració més personal i global de la feina feta, l'autor es considera força satisfet amb el projecte desenvolupat, amb els coneixements diversos i l'enriquiment professional que aquest li ha aportat i amb l'enfocament que se li ha donat al TFC desde la UOC.

10. GLOSSARI

- *sniffer (escoltador de xarxa):* Utilitat de xarxa que permet analitzar a nivell TCP/IP tot el tràfic circulant per un punt determinat d'una xarxa, com pot ser un ordinador qualsevol que hi està conectat.
- *Embedded*: (embebido) Terme utilitzat en informàtica generalment per indicar la possibilitat d'utilitzar una tecnologia dins d'una altra.
- *emoticonos*: Imatges utilitzades en els sistemes de missatgeries per expressar emocions o estats d'ànim.
- atac de l'home al mig: Atac de a la seguretat d'un sistema de comunicació entre parells que consisteix en suplantar la identitat dels dos usuaris situant-se al mig de la conexió.
- *Socket*: Mètode de comunicació entre diferents aplicacions, semblant a les pipes però que permet l'enviament de la informació a través de la xarxa, és a dir, entre ordinadors diferents.
- *Tag*: Unitat base de l'estructura dels fitxers XML que s'indica delimitada amb els símbols "<" i ">".
- Xml (eXtensive Markup Language): Llenguatge que permet organitzar la informació en un fitxer de text pla de manera jeràrquica, facilitant-ne la cerca i recuperació

11. BIBLIOGRAFIA/RECURSOS

- Mundomicro: intercambio de claves Diffie-Hellman. http://webs.ono.com/usr005/jsuarez/diffiehellman.html
- Zona Gratuíta.Com: http://www.zonagratuita.com/servicios/seguridad/criptografia.html
- Seguridad en Java http://www.uv.es/~sto/cursos/seguridad.java/html/sjava.html
- Informació llibreries criptogràfiques (PPT) http://www.rediris.es/cert/iris-pca/gti-pca/tras-JT01/Librerias_Criptografica.ppt
- Plana principal del software de desenvolupament Eclipse. http://www.eclipse.org/
- IAIK – Llibreries criptogràfiques http://jce.iaik.tugraz.at/
- JDOM : http://www.jdom.org/
- Jigloo GUI: http://www.cloudgarden.com/jigloo/
- Apartat de seguretat de la plana W3C.org: http://www.w3.org/Signature/
- Plana web del Java de SUN: http://java.sun.com/
- X-extraInternet: http://www.x-extrainternet.com/messengers.asp
- UOC. *Criptografia (Apunts de l'assignatura)*. 2003
- UOC. Administració de sistemes operatius en xarxa (Apunts de l'assignatura).2004
- UOC. *Enginyeria del programari (Apunts de l'assignatura)*. 2004
- UOC. Xarxes. Aplicacions i protocols d'Internet (*Apunts de l'assignatura*). 2004

12. CODI FONT

En aquest apartat s'inclou el codi de les diferents classes del projecte. Algunes d'elles són específiques d'una de les dues aplicacions mentre que altres són compartides. Per tant, no hi ha un codi font separat per el client i el servidor missatger. D'altra banda, cal tenir en compte que en els procediments més complexes i hi ha un gran nombre de tabulacions, cosa que fa difícil de seguir el contingut del codi en un document amb una amplada bastant limitada. Malgrat tot, s'ha disposat el codi en una font petita i d'amplada fixe per millorar-ne la lectura.

Paquet Beans

```
// SocketMissatge

import java.net.Socket;
import java.util.Hashtable;

public class SocketMissatge implements java.io.Serializable
{
	/**
	 *
	 */
	private static final long serialVersionUID = -7549222510508429929L;
	private long		serieID;
	private int		Header;
	private String		Email;
	private String		Desti;
	private String		Missatge;
	private String		Host;
	private Usuari		User;
	private Hashtable LlistaUsuaris;
	private String		Dades;
	private Socket		CSocket;
	private byte[]		EncodedParams;
	private int		Estat;

	public SocketMissatge()
	{
		//Classe buida
	}

	public String getDades() {
		return Dades;
	}

	public void setDades(String dades) {
		Dades = dades;
	}

	public String getDesti() {
		return Desti;
	}

	public void setDesti(String desti) {
```

```
        Desti = desti;
}

public int getHeader() {
        return Header;
}

public void setHeader(int header) {
        Header = header;
}

public String getHost() {
        return Host;
}

public void setHost(String host) {
        Host = host;
}

public String getMissatge() {
        return Missatge;
}

public void setMissatge(String missatge) {
        Missatge = missatge;
}

public Usuari getUser() {
        return User;
}

public void setUser(Usuari user) {
        User = user;
}

public String getEmail() {
        return Email;
}

public void setEmail(String mail) {
        Email = mail;
}

public SocketMissatge(int header)
{
        Header = header;
}

public SocketMissatge(int header,String missatge)
{
        Header = header;
        Missatge=missatge;
}

public String getSocketMissatge()
{
        return Missatge;
}

public Socket getCSocket() {
        return CSocket;
}
```

```
	public void setCSocket(Socket socket) {
		CSocket = socket;
	}

	public Hashtable getLlistaUsuaris() {
		return LlistaUsuaris;
	}

	public void setLlistaUsuaris(Hashtable llistaUsuaris) {
		LlistaUsuaris = llistaUsuaris;
	}

	public byte[] getEncodedParams() {
		return EncodedParams;
	}

	public void setEncodedParams(byte[] encodedParams) {
		EncodedParams = encodedParams;
	}

	public int getEstat() {
		return Estat;
	}

	public void setEstat(int estat) {
		Estat = estat;
	}

	public long getSerieID() {
		return serieID;
	}

	public void setSerieID(long serieID) {
		this.serieID = serieID;
	}
}
```

// Usuari

```
import java.util.Hashtable;

public class Usuari implements java.io.Serializable {

	private String Nick;
	private String Email;
	private String Password;
	private boolean Admes;
	private boolean Conectat;
	private String IP;
	private Hashtable UClients;
	private String PrivateKey;
	private String PublicKey;
	private int Estat;

	public int getEstat() {
		return Estat;
	}

	public void setEstat(int estat) {
		Estat = estat;
	}
```

```
public Usuari(){
}

public boolean isAdmes() {
	return Admes;
}

public void setAdmes(boolean admes) {
	Admes = admes;
}

public String getEmail() {
	return Email;
}

public void setEmail(String email) {
	Email = email;
}

public String getNick() {
	return Nick;
}

public void setNick(String nom) {
	Nick = nom;
}

public String getIP() {
	return IP;
}

public void setIP(String ip) {
	IP = ip;
}

public boolean isConectat() {
	return Conectat;
}

public void setConectat(boolean conectat) {
	Conectat = conectat;
}

public String getPassword() {
	return Password;
}

public void setPassword(String password) {
	Password = password;
}

public Hashtable getUClients() {
	return UClients;
}

public void setUClients(Hashtable clients) {
	UClients = clients;
}

public String getPrivateKey() {
	return PrivateKey;
}
```

```
	public void setPrivateKey(String privateKey) {
		PrivateKey = privateKey;
	}

	public String getPublicKey() {
		return PublicKey;
	}

	public void setPublicKey(String publicKey) {
		PublicKey = publicKey;
	}
}
```

Paquet Missatger

```
// ConexioClient

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.InetAddress;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.Vector;

import org.eclipse.swt.SWT;
import org.eclipse.swt.widgets.Shell;

import swt.AlertDialog;
import swt.MissatgerSWT;
import Beans.SocketMissatge;
import Beans.Usuari;
import Utils.Constants;
import Utils.MissatgerUtils;


public class ConexioClient extends Thread implements Constants {

	private DataInputStream entrada = null;
	private DataOutputStream sortida = null;
	private String SERVIDOR;
	private String USUARI;
	private Shell pgmShell;
	private GestorXML oGest;
	private ConfigClient users;
	private MissatgerSWT rootForm;
	private SocketMissatge msg;
	private SocketMissatge msg2;
	private long lastreceived;
	private String lastemail;
	private Socket client = null;
	private String serror;
	private boolean acaba;
	private Usuari user;
	private Usuari userPK;
	private Encriptador enc;
	private String GPK;
	private long contador;

	public ConexioClient(MissatgerSWT msgSWT){
		/* Preparar la classe */
		this.pgmShell = msgSWT.getShell();
```

```
            this.oGest = new GestorXML();
            this.users = oGest.getConfigClient();
            this.userPK = oGest.getUsuari();
            this.rootForm=msgSWT;
            enc = new Encriptador(userPK.getPrivateKey(), pgmShell);
            SERVIDOR = oGest.getConfigClient().getIPServidor();
            lastreceived=0;
            lastemail="";
            contador=0;
      }

      public void run()
      {
            try    {
                byte[] data;

                  /* Obrir socket i conectar amb el servidor */
                  client                          =                          new
Socket(InetAddress.getByName(SERVIDOR),Integer.parseInt(users.getPort()));
                  sortida = new DataOutputStream(client.getOutputStream());
                  entrada = new DataInputStream(client.getInputStream());
                  if (client != null && sortida != null && entrada != null) {

                        /* Generar i enviar identificació */
                        user = oGest.getUsuari();
                        SocketMissatge            msg            =            new
SocketMissatge(CLIENT_LOGIN,"");
                        user.setPrivateKey("");
                        user.setPublicKey("");
                        msg.setUser(user);
                        USUARI=user.getEmail().trim();
                        enviaMissatge(msg);

                        /* Esperar resposta USEROK o USERKO */
                        data = new byte[MAX_MISSATGE];
                        entrada.read(data);
                        msg                                                    =
((SocketMissatge)MissatgerUtils.bytesToObject(data));
                        if (msg.getHeader()==USER_OK){

                              /* Generar i enviar llista d'usuaris */
                              msg = new SocketMissatge(USUARIS_LIST,"");
                              msg.setUser(user);
                              msg.setLlistaUsuaris(oGest.getUsuarisClient());
                              enviaMissatge(msg);

                              /* Rebre llista de contactes */
                              data = new byte[MAX_MISSATGE];
                              entrada.read(data);
                              msg2                                             =
((SocketMissatge)MissatgerUtils.bytesToObject(data));

                              if       (msg2.getHeader()==USUARIS_LIST       &&
msg2.getLlistaUsuaris()!=null){
                                    /* Processar llista */
                                    pgmShell.getDisplay().asyncExec(new
Runnable() {
                                          public void run () {

      rootForm.processaMissatge(msg2);
                                          }
                                    });
```

```
                                        /* Enviar LIST_OK */
                                        msg = new SocketMissatge(LIST_OK,"");
                                        msg.setDesti(user.getEmail().trim());
                                        enviaMissatge(msg);

                                        rootForm.setConectat(true);
                                        actualitzaMenu();

                                        System.out.println("----  Client  Missatger
CONECTAT -------");
                                        Vector missatges=new Vector();
                                        while(rootForm.isConectat() && !acaba) {
                                        /* Processa missatges entrants */
                                              if (entrada.available()>0) {
                                                    data          =          new
byte[MAX_MISSATGE];
                                                    entrada.read(data);
                                                    msg                        =
((SocketMissatge)MissatgerUtils.bytesToObject(data));
                                                    processaMissatge(msg);

      //System.out.println("Missatge rebut");
                                                    msg           =           new
SocketMissatge(MSG_OK,"");
                                                    enviaMissatge(msg);

      //System.out.println("Missatge rebut confirmat");
                                              }
                                              sleep(SLEEP_TIME);
                                              /* Processa missatges sortints */
                                              if             (missatges.size()==0)
missatges=rootForm.readCuaMissatges();
                                              if      (missatges.size()>0      &&
entrada.available()==0){
                                                    for                      (int
a=0;a<missatges.size();a++){

      enviaMissatge((SocketMissatge)missatges.elementAt(a));

      //System.out.println("Missatge enviat");
                                                          msg        =        new
SocketMissatge();
                                                          while
(!(msg.getHeader()==MSG_OK)){
                                                                data    =    new
byte[MAX_MISSATGE];

      entrada.read(data);
                                                                msg            =
((SocketMissatge)MissatgerUtils.bytesToObject(data));
                                                                if
(!(msg.getHeader()==MSG_OK)) {

      processaMissatge(msg);

      //System.out.println("Missatge rebut dins l'enviat");
                                                                      msg = new
SocketMissatge(MSG_OK,"");

      enviaMissatge(msg);

      //System.out.println("Missatge rebut dins l'enviat confirmat");
                                                                }
```

```
	//System.out.println("Missatge enviat confirmat");
                                                                }

                                                                //sleep(SLEEP_TIME);
                                                        }
                                                        missatges.clear();
                                                }
                                                sleep(SLEEP_TIME);
                                        }
                                                /*    Tanca    la    conexió:    envia    msg
CLIENT_LOGOUT */
                                                msg                  =                  new
SocketMissatge(CLIENT_LOGOUT,"");
                                                msg.setUser(user);
                                                enviaMissatge(msg);
                                                System.out.println("----  Client  Missatger
DESCONECTAT ----");

                                                /* Sincronització de sortida */
                                                if                   (rootForm.isConSync())
rootForm.setConSync(false);
                                        }
                                        else{
                                                serror="Error de protocol en la interacció
amb el servidor";
                                                informacioError();
                                        }

                                }
                                else{
                                        serror="El servidor no ha aceptat l'usuari i la
contrassenya";
                                        informacioError();
                                }
                        }

                        entrada.close();
                        sortida.close();
                        client.close();

                } catch (UnknownHostException e) {
                        serror="No  s'ha  pogut  trobar  el  servidor.\n  Intenta-ho  una
altra vegada.";
                        informacioError();
                } catch (IOException e) {
                        serror="Error en la conexió.\n Intenta-ho una altra vegada.";
                        informacioError();
                } catch(Exception exception) {
                        serror="S'ha   produït   un   error.\n   Intenta-ho   una   altra
vegada.";
                        informacioError();
                        exception.printStackTrace();
                }
                rootForm.setConectat(false);
                actualitzaMenu();

        }

        public void actualitzaMenu(){
                if (!rootForm.isDisposed()){
                        pgmShell.getDisplay().syncExec(new Runnable() {
                                public void run () {
```

```
                                rootForm.actualitzaMenu(rootForm.isConectat());
                                rootForm.actualitzaCombo(rootForm.isConectat());
                        }
                });
        }
}
public void informacioError(){
        if (!rootForm.isDisposed()){
                pgmShell.getDisplay().syncExec(new Runnable() {
                        public void run () {
                                new AlertDialog(ERROR, serror);

                        }
                });
        }
}

public void enviaMissatge(SocketMissatge mesg)
{
        byte[] data;
        /* Afegin el servidor i l'email de l'usuari a tots els paquets */
        mesg.setEmail(USUARI);
        mesg.setHost(SERVIDOR);

        /* Si és un missatge tipus CHAT aplica el xifratge */
        if (mesg.getHeader()==CHAT){
                contador++;
                mesg.setSerieID(contador);
                synchronized(enc){
                        GPK = oGest.getUsuariClientPK(mesg.getDesti());
                        mesg                                                    =
enc.Encripta(oGest.getUsuariClientPK(mesg.getDesti()),mesg);
                }
        }

data = MissatgerUtils.objectToBytes(mesg);
try {
        sortida.write(data,0,data.length);
        sortida.flush();
} catch (java.net.SocketException se){
        /* La conexió ha petat, sortir */
                try {
                        client.close();
                } catch (IOException e) {
                        /* No fer res */
                }
                serror="S'ha perdut la conexió amb el servidor.";
                informacioError();
                rootForm.setConectat(false);
                actualitzaMenu();
                this.stop();
} catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
        }
}

public synchronized void processaMissatge(SocketMissatge mesg){

        msg = mesg;
                if (msg.getDesti()==null) msg.setDesti("");
                // Comprovem que el missatge sigui per nosaltres
```

```
                    if
(!((msg.getDesti().trim().compareTo(user.getEmail().trim())!=0)&&(msg.getHeader()
==CHAT))){
                          // Comprovem que no sigui una repetició
                          if              (!(msg.getHeader()==CHAT              &&
msg.getSerieID()<=lastreceived                                                  &&
lastemail.compareTo(msg.getEmail().trim())==0)){

                                 if (msg.getHeader()!=SYNCRO){
                                        if (msg.getHeader()==CHAT){
                                               /* Desencriptar */
                                               synchronized(enc){
                                                      String          PK          =
oGest.getUsuariClientPK(msg.getEmail());
                                                      msg                         =
enc.Desencripta(PK,msg);
                                               }
                                        }
                                        if (msg!=null){
                                               if (msg.getHeader()==CHAT){

       lastreceived=msg.getSerieID();
                                                      lastemail=msg.getEmail();

       //System.out.println("Missatge      CHAT:      "+msg.getSerieID()+"      de
"+msg.getEmail()+ " ("+lastreceived+")");
                                               }
                                               /* Processar el missatge */
                                               pgmShell.getDisplay().asyncExec(new
Runnable() {
                                                      public void run () {

       rootForm.processaMissatge(msg);
                                                      }
                                               });
                                        }
                                        else {
                                               System.out.println("Errors        al
desencriptar el missatge: "+enc.getSerror());
                                        }
                                 }
                                 else{
                                        /* Enviar sincronisme */
                                        msg.setUser(user);
                                        enviaMissatge(msg);
                                 }
                          }
                    }
       }

       public Usuari getUser() {
              return user;
       }
}
```

// ConfigClient

```
import java.util.Vector;

import Beans.Usuari;


public class ConfigClient {
```

```
private String IPServidor;
private String Port;
private String Nick;
private String EMail;
private String Password;
private String IP;
private Vector Usuaris;
private String PrivateKey;
private String PublicKey;

public String getPrivateKey() {
	return PrivateKey;
}

public void setPrivateKey(String privateKey) {
	PrivateKey = privateKey;
}

public String getPublicKey() {
	return PublicKey;
}

public void setPublicKey(String publicKey) {
	PublicKey = publicKey;
}

public String getIP() {
	return IP;
}

public void setIP(String ip) {
	IP = ip;
}

public String getEMail() {
	return EMail;

}

public void setEMail(String mail) {
	EMail = mail;
}

public String getIPServidor() {
	return IPServidor;
}

public void setIPServidor(String servidor) {
	IPServidor = servidor;
}

public String getNick() {
	return Nick;
}

public void setNick(String nick) {
	Nick = nick;
}

public String getPort() {
	return Port;
}
```

```
        public void setPort(String port) {
                Port = port;
        }

        public ConfigClient(){

                IPServidor="";
                Port="";
                Nick="";
                EMail="";
                Password="";
                IP="";
                Usuaris=new Vector();

        }

        public String getPassword() {
                return Password;
        }

        public void setPassword(String password) {
                Password = password;
        }

        public Vector getUsuaris() {
                return Usuaris;
        }

        public void setUsuaris(Vector usuaris) {
                Usuaris = usuaris;
        }

        public void addUsuari(Usuari oUsuari){
                Usuaris.addElement(oUsuari);
        }
        public String getUsuari(int num){
                return (String) (Usuaris.elementAt(num));
        }
        public int getNumUsuaris(){
                return(Usuaris.size());
        }
}
```

// Encriptador

```
import iaik.security.dh.DHPrivateKey;
import iaik.security.dh.DHPublicKey;
import iaik.utils.Base64Exception;
import iaik.utils.Util;

import java.io.IOException;
import java.security.AlgorithmParameters;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.security.PrivateKey;
import java.security.PublicKey;

import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.KeyAgreement;
import javax.crypto.NoSuchPaddingException;
```

```
import javax.crypto.SecretKey;

import org.eclipse.swt.widgets.Shell;

import swt.AlertDialog;
import Beans.SocketMissatge;
import Utils.Constants;

public class Encriptador implements Constants {
	private PrivateKey clau_privada;
	private Shell _shell;
	private String serror;
	private  byte[] tempByte;

	public String getSerror() {
		return serror;
	}

	public Encriptador(String PVK, Shell sh){
		this._shell = sh;
		try {
			clau_privada	=	new
DHPrivateKey(Util.Base64Decode(PVK.getBytes()));
		} catch (InvalidKeyException e) {
			serror="La clau privada proporcionada no és vàlida";
		} catch (Base64Exception e) {
			serror="La clau privada no és un base 64 vàlid";
		}
	}

	public SocketMissatge Encripta(String publica, SocketMissatge msg){

		KeyAgreement dh_key_agreement;
		String msg_clar = msg.getMissatge();
		try {

			/* Llegeix la clau pública */
			PublicKey	clau_publica	=	new
DHPublicKey(Util.Base64Decode(publica.getBytes()));

			/* Crea la negociació de conversa */
			dh_key_agreement = KeyAgreement.getInstance("DH");
			dh_key_agreement.init(clau_privada);
			dh_key_agreement.doPhase(clau_publica,true);

			/* Genera el secret */
			SecretKey DesKey1 = dh_key_agreement.generateSecret("DES");

		/* Crea el xifrador */
			Cipher Cipher1 = Cipher.getInstance("DES/CBC/PKCS5Padding");
		Cipher1.init(Cipher.ENCRYPT_MODE, DesKey1);

		byte[] msg_xifrat = Cipher1.doFinal(msg_clar.getBytes());
		byte[] encodedParams = Cipher1.getParameters().getEncoded();
		tempByte=msg_xifrat;
		//System.out.println("Missatge	enviat:	["+new
String(Util.Base64Encode(msg_xifrat))+"] a "+msg.getDesti());
		msg.setMissatge(new String(Util.Base64Encode(msg_xifrat)));
		msg.setEncodedParams(encodedParams);

		return msg;

		} catch (NoSuchAlgorithmException e) {
```

```
                serror="No s'ha trobat l'algorisme de la clau privada";
            } catch (InvalidKeyException e) {
                serror="La clau privada proporcionada no concorda";
            } catch (Base64Exception e) {
                serror="La clau pública no és un base 64 vàlid";
            } catch (NoSuchPaddingException e) {
                serror="No es troba el mètode seleccionat per xifrar";
            } catch (IllegalStateException e) {
                serror="Error en el procés de xifratge";
            } catch (IllegalBlockSizeException e) {
                serror="ERROR IllegalBlockSize";
            } catch (BadPaddingException e) {
                serror="ERROR BadPaddingException";
            } catch (IOException e) {
                serror="Error al llegir els paràmetres del desxifrador";
            }

            //informacioError();
        return null;
      }
      public SocketMissatge Desencripta(String publica, SocketMissatge msg){

            KeyAgreement dh_key_agreement;

            /* Crea la negociació de conversa */
            try {

                dh_key_agreement = KeyAgreement.getInstance("DH");
                dh_key_agreement.init(clau_privada);

                /* Llegeix la clau pública */
                PublicKey            clau_publica            =            new
DHPublicKey(Util.Base64Decode(publica.getBytes()));

                dh_key_agreement.doPhase(clau_publica,true);

                //System.out.println("Missatge       rebut:             ["+new
String(msg.getMissatge().getBytes())+"] de "+msg.getEmail());

                /* Obre els paràmetres utilitzats */
              AlgorithmParameters                    params                    =
AlgorithmParameters.getInstance("DES");
              params.init(msg.getEncodedParams());

                SecretKey DesKey = dh_key_agreement.generateSecret("DES");
                Cipher Cipher1 = Cipher.getInstance("DES/CBC/PKCS5Padding");
                Cipher1.init(Cipher.DECRYPT_MODE, DesKey, params);
                msg.setMissatge(new
String(Cipher1.doFinal(Util.Base64Decode(msg.getMissatge().getBytes()))));

                return msg;

            } catch (NoSuchAlgorithmException e) {
                serror="No s'ha trobat l'algorisme de la clau privada";
            } catch (InvalidKeyException e) {
                serror="La clau privada proporcionada no concorda";
            } catch (Base64Exception e) {
                serror="La clau pública no és un base 64 vàlid";
            } catch (NoSuchPaddingException e) {
                serror="No es troba el mètode seleccionat per desxifrar";
            } catch (IllegalStateException e) {
                serror="Error en el procés de desxifratge";
            } catch (IllegalBlockSizeException e) {
```

```
                    serror="ERROR IllegalBlockSize";
             } catch (BadPaddingException e) {
                    serror="ERROR BadPaddingException";
             } catch (IOException e) {
                    serror="Error al llegir els paràmetres del desxifrador";

             } catch (InvalidAlgorithmParameterException e) {
                    serror="No   s'ha   trobat   l'algorisme   utilitzat   en   els
paràmetres";
             }

             //informacioError();
             return null;

       }
       public PrivateKey getClau_privada() {
             return clau_privada;
       }

       public void setClau_privada(PrivateKey clau_privada) {
             this.clau_privada = clau_privada;
       }

       public void informacioError(){
             _shell.getDisplay().syncExec(new Runnable() {
                    public void run () {
                           new AlertDialog(ERROR, serror);

                    }
             });
       }
}
```

// GestorXML

```
package Missatger;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.Hashtable;
import java.util.List;
import java.util.Vector;

import org.jdom.Document;
import org.jdom.Element;
import org.jdom.JDOMException;
import org.jdom.input.SAXBuilder;
import org.jdom.output.XMLOutputter;

import swt.AlertDialog;
import Beans.Usuari;
import Utils.Constants;

public class GestorXML implements Constants {
       public GestorXML(){
       }

       public ConfigClient getConfigClient(){

             // Crea un nou objecte UsuarisXML
```

```
            ConfigClient users = new ConfigClient();

            // Obre i parseja el fitxer usuaris.xml mitjançant JDOM
            FileInputStream in = null;
            try {
                  in = new FileInputStream("usuaris.xml");
                  SAXBuilder parser = new SAXBuilder();
                  Document xmldoc = null;

                  xmldoc = parser.build(in);

                  Element root = xmldoc.getRootElement();

                  // Obre el node principal
                  Element oConexio = root.getChild("Conexio");

                  //  Extreu  les  dades  de  conexió  i  les  guarda  a  l'objecte
UsuarisXML

      users.setIPServidor((oConexio.getChild("IPServidor")).getText());
                  users.setPort((oConexio.getChild("Port")).getText());
                  users.setNick((oConexio.getChild("Nick")).getText());
                  users.setEMail((oConexio.getChild("Email")).getText());

      users.setPassword((oConexio.getChild("Contrassenya")).getText());

      users.setPrivateKey((oConexio.getChild("PrivateKey")).getText());

      users.setPublicKey((oConexio.getChild("PublicKey")).getText());

                  // Lectura d'usuaris
                  Element oUsuaris = root.getChild("Usuaris");
                  List lst = oUsuaris.getChildren();
                  Vector usrs = new Vector();
                  for (int a=0;a<lst.size();a++){
                        Element eUsuari = (Element) lst.get(a);
                        Usuari oUsuari = new Usuari();
                        oUsuari.setNick(eUsuari.getChild("Nick").getText());
                        oUsuari.setEmail(eUsuari.getChild("Email").getText());
                        oUsuari.setAdmes((new
Boolean(eUsuari.getChild("Admes").getText())).booleanValue());
                        if
(eUsuari.getChild("Admes").getText().compareTo("1")==0) {
                              oUsuari.setAdmes(true);
                        }
                        else {
                              oUsuari.setAdmes(false);
                        }

      oUsuari.setPublicKey((eUsuari.getChild("PublicKey")).getText());
                        usrs.addElement(oUsuari);
                  }
                  users.setUsuaris(usrs);

            } catch (FileNotFoundException e) {
                  new AlertDialog(ERROR,"No es troba el fitxer usuaris.xml");
            } catch (JDOMException e) {
                  new    AlertDialog(ERROR,"Error    al    parsejar    document
usuaris.xml");
            } catch (IOException e) {
                  new AlertDialog(ERROR,"Error al llegir document usuaris.xml");
            }
            return users;
```

```
	}

	public Usuari getUsuari(){

		// Crea un nou objete UsuarisXML
		Usuari user = new Usuari();

		// Obre i parseja el fitxer usuaris.xml mitjançant JDOM
		FileInputStream in = null;
		try {
			in = new FileInputStream("usuaris.xml");
			SAXBuilder parser = new SAXBuilder();
			Document xmldoc = null;
			xmldoc = parser.build(in);

			Element root = xmldoc.getRootElement();

			// Obre el node principal
			Element oConexio = root.getChild("Conexio");

			// Extreu les dades de l'usuari local //
			user.setNick((oConexio.getChild("Nick")).getText());
			user.setEmail((oConexio.getChild("Email")).getText());

	user.setPassword((oConexio.getChild("Contrassenya")).getText());

	user.setPrivateKey((oConexio.getChild("PrivateKey")).getText());
			user.setPublicKey((oConexio.getChild("PublicKey")).getText());


		} catch (FileNotFoundException e) {
			new AlertDialog(ERROR,"No es troba el fitxer usuaris.xml");
		} catch (JDOMException e) {
			new	AlertDialog(ERROR,"Error	al	parsejar	document
usuaris.xml");
		} catch (IOException e) {
			new AlertDialog(ERROR,"Error al llegir document usuaris.xml");
		}

		return user;
	}

	public void setConfigClient(ConfigClient users) {

		/*Crear els nodes */
		Document xmldoc = new Document();
		Element oConfiguraClient = new Element("ConfiguraClient");
		Element oConexio = new Element ("Conexio");
		Element oUsuaris = new Element ("Usuaris");

		/* Dades de conexió */
		Element oIPServidor = new Element ("IPServidor");
		oIPServidor.setText(users.getIPServidor());
		Element oPort = new Element ("Port");
		oPort.setText(users.getPort());
		Element oNick = new Element ("Nick");
		oNick.setText(users.getNick());
		Element oEmail = new Element ("Email");
		oEmail.setText(users.getEMail());
		Element oPassword = new Element ("Contrassenya");
		oPassword.setText(users.getPassword());
		Element oPublicKey = new Element ("PublicKey");
		oPublicKey.setText(users.getPublicKey());
```

```
		Element oPrivateKey = new Element ("PrivateKey");
		oPrivateKey.setText(users.getPrivateKey());

		oConexio.addContent(oIPServidor);
		oConexio.addContent(oPort);
		oConexio.addContent(oNick);
		oConexio.addContent(oEmail);
		oConexio.addContent(oPassword);
		oConexio.addContent(oPublicKey);
		oConexio.addContent(oPrivateKey);
		oConfiguraClient.addContent(oConexio);

		Vector lUsers = users.getUsuaris();

		for (int a=0;a<lUsers.size();a++){

			/* Element usuari */
			Element oUsuari = new Element ("Usuari");

			/* Dades */
			Element ouNick = new Element ("Nick");
			ouNick.setText(((Usuari) (lUsers.get(a))).getNick());
			Element ouEmail = new Element ("Email");
			ouEmail.setText(((Usuari) (lUsers.get(a))).getEmail());
			Element oAdmes = new Element ("Admes");
			boolean bVal = ((Usuari) (lUsers.get(a))).isAdmes();
			if (bVal == true) {
				oAdmes.setText("1");
			}
			else {
				oAdmes.setText("0");
			}
			Element oPK = new Element("PublicKey");
			oPK.setText(((Usuari) (lUsers.get(a))).getPublicKey());

			oUsuari.addContent(ouNick);
			oUsuari.addContent(ouEmail);
			oUsuari.addContent(oAdmes);
			oUsuari.addContent(oPK);
			oUsuaris.addContent(oUsuari);

		}
		oConfiguraClient.addContent(oUsuaris);
		xmldoc.setRootElement(oConfiguraClient);

		FileOutputStream out;
		try {
			out = new FileOutputStream("usuaris.xml");
			XMLOutputter xmlout = new XMLOutputter();
			xmlout.output(xmldoc, out);
			out.close();
		} catch (FileNotFoundException e) {
			new AlertDialog(ERROR,"No es troba el fitxer usuaris.xml");
		} catch (IOException e) {
			new	AlertDialog(ERROR,"Error	al	escriure	document
usuaris.xml");
		}
	}

	public Hashtable getUsuarisClient(){

		// Obre i parseja el fitxer usuaris.xml mitjançant JDOM
		FileInputStream in = null;
```

```
		Hashtable usrs=null;
		try {
			in = new FileInputStream("usuaris.xml");
			SAXBuilder parser = new SAXBuilder();
			Document xmldoc = null;
			xmldoc = parser.build(in);

			Element root = xmldoc.getRootElement();

			// Lectura d'usuaris
			Element oUsuaris = root.getChild("Usuaris");
			List lst = oUsuaris.getChildren();
			usrs = new Hashtable();
			for (int a=0;a<lst.size();a++){
				Element eUsuari = (Element) lst.get(a);
				Usuari oUsuari = new Usuari();
				oUsuari.setNick(eUsuari.getChild("Nick").getText());
				oUsuari.setEmail(eUsuari.getChild("Email").getText());
				oUsuari.setAdmes((new
Boolean(eUsuari.getChild("Admes").getText())).booleanValue());
				if
(eUsuari.getChild("Admes").getText().compareTo("1")==0) {
					oUsuari.setAdmes(true);
				}
				else {
					oUsuari.setAdmes(false);
				}

	//oUsuari.setPublicKey(eUsuari.getChild("PublicKey").getText());
				usrs.put(oUsuari.getEmail().trim(),oUsuari);
			}

		} catch (FileNotFoundException e) {
			new AlertDialog(ERROR,"No es troba el fitxer usuaris.xml");
		} catch (JDOMException e) {
			new	AlertDialog(ERROR,"Error	al	parsejar	document
usuaris.xml");
		} catch (IOException e) {
			new AlertDialog(ERROR,"Error al llegir document usuaris.xml");
		}

		return usrs;
	}

	public String getUsuariClientPK(String email){

		// Obre i parseja el fitxer usuaris.xml mitjançant JDOM
		FileInputStream in = null;
		Hashtable usrs=null;
		try {
			in = new FileInputStream("usuaris.xml");
			SAXBuilder parser = new SAXBuilder();
			Document xmldoc = null;
			xmldoc = parser.build(in);
			Usuari oUsuari = null;

			Element root = xmldoc.getRootElement();

			// Lectura d'usuaris
			Element oUsuaris = root.getChild("Usuaris");
			List lst = oUsuaris.getChildren();
			usrs = new Hashtable();
			for (int a=0;a<lst.size();a++){
```

```
                        Element eUsuari = (Element) lst.get(a);

                        if
(eUsuari.getChild("Email").getText().trim().compareTo(email)==0){
                                return(eUsuari.getChild("PublicKey").getText());
                        }
                }

            } catch (FileNotFoundException e) {
                new AlertDialog(ERROR,"No es troba el fitxer usuaris.xml");
            } catch (JDOMException e) {
                new    AlertDialog(ERROR,"Error    al    parsejar    document
usuaris.xml");
            } catch (IOException e) {
                new AlertDialog(ERROR,"Error al llegir document usuaris.xml");
            }

            return null;
        }

}
```

Paquet Servidor Missatger

```
// GestorXML

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.StringReader;
import java.util.Hashtable;
import java.util.List;
import java.util.Vector;

import org.jdom.Document;
import org.jdom.Element;
import org.jdom.JDOMException;
import org.jdom.input.SAXBuilder;

import Beans.Usuari;

public class GestorXML {
        public GestorXML(){
        }

        public int getPort(){
            // Obre i parseja el fitxer Configura.xml mitjançant JDOM
            FileInputStream in = null;
            try {
                in = new FileInputStream("Configura.xml");
                SAXBuilder parser = new SAXBuilder();
                Document xmldoc = null;
                xmldoc = parser.build(in);

                // Obre el node principal
                Element root = xmldoc.getRootElement();
                Element oConexio = root.getChild("Conexio");

                return Integer.parseInt(oConexio.getChildText("Port"));

            } catch (FileNotFoundException e) {
```

```
		e.printStackTrace();
	} catch (JDOMException e) {
		e.printStackTrace();
	} catch (IOException e) {
		e.printStackTrace();
	}
	return 0;
}

public Hashtable getUsuarisXML(){

	// Crea una taula per ubicar els usuaris
	Hashtable hUsuaris  = new Hashtable();

	// Obre i parseja el fitxer Configura.xml mitjançant JDOM
	FileInputStream in = null;
	try {
		in = new FileInputStream("Configura.xml");
	} catch (FileNotFoundException e) {
		e.printStackTrace();
	}
	SAXBuilder parser = new SAXBuilder();
	Document xmldoc = null;
	try {
		xmldoc = parser.build(in);
	} catch (JDOMException e) {
		e.printStackTrace();
	} catch (IOException e) {
		e.printStackTrace();
	}

	// Obre el node principal
	Element root = xmldoc.getRootElement();

	// Lectura d'usuaris
	Element oUsuaris = root.getChild("Usuaris");
	List lst = oUsuaris.getChildren();

	for (int a=0;a<lst.size();a++){
		Element eUsuari = (Element) lst.get(a);
		Usuari oUsuari = new Usuari();
		oUsuari.setEmail(eUsuari.getChild("Email").getText());

oUsuari.setPassword(eUsuari.getChild("Contrassenya").getText());
		oUsuari.setNick("");
		oUsuari.setConectat(false);
		oUsuari.setIP("");
		oUsuari.setAdmes(false);
		hUsuaris.put(oUsuari.getEmail(),oUsuari);
	}
	return hUsuaris;
}

/* Genera la classe Usuari en base al string rebut */
public Usuari getIdentificaClient(String xmlstr){

	Usuari user = new Usuari();
	SAXBuilder parser = new SAXBuilder();
	Document xmldoc = null;

	try {
		xmldoc = parser.build(new StringReader(xmlstr));
	} catch (JDOMException e) {
```

```
                System.out.println("Error al parsejar IdentificaClient");
                e.printStackTrace();
            } catch (IOException e) {
                System.out.println("Error al parsejar IdentificaClient");
                e.printStackTrace();
            }

            /* Obtenir les dades del fitxer XML */
            Element root = xmldoc.getRootElement();
            user.setNick(root.getChildText("Nick"));
            user.setEmail(new String(root.getChildText("Email")).trim());
            user.setPassword(new
String(root.getChildText("Contrassenya")).trim());

            return user;
        }

        public Vector getUsuarisClient(String xmlstr){
            Vector vUsuaris = new Vector();
            SAXBuilder parser = new SAXBuilder();
            Document xmldoc = null;

            /* Parsejar el document XML */
            try {
                xmldoc = parser.build(new StringReader(xmlstr));
            } catch (JDOMException e) {
                System.out.println("Error al parsejar UsuarisClient");
                return null;
            } catch (IOException e) {
                System.out.println("Error al parsejar UsuarisClient");
                return null;
            }

            /* Lectura d'usuaris */
            Element root = xmldoc.getRootElement();
            Element oUsuaris = root.getChild("Usuaris");
            List lst = oUsuaris.getChildren();

            for (int a=0;a<lst.size();a++){
                Element eUsuari = (Element) lst.get(a);
                Usuari oUsuari = new Usuari();
                oUsuari.setEmail(eUsuari.getChild("Email").getText());
                oUsuari.setAdmes((new
Boolean(eUsuari.getChild("Admes").getText())).booleanValue());
                if (eUsuari.getChild("Admes").getText().compareTo("1")==0) {
                    oUsuari.setAdmes(true);
                }
                else {
                    oUsuari.setAdmes(false);
                }
                vUsuaris.addElement(oUsuari);
            }
            return vUsuaris;
        }

}
```

// Servei

```
import java.io.DataInputStream;
import java.net.Socket;
import java.util.Hashtable;
import java.util.Vector;
```

```
import Beans.SocketMissatge;
import Beans.Usuari;
import Utils.Constants;
import Utils.MissatgerUtils;

class Servei implements Runnable, Constants
{
	private DataInputStream in;
	private Socket socket;
	private boolean acaba=false;
	private Thread fil;
	private Usuari user;
	private Vector missatges;
	private Utils.Timer timer;
	private boolean waiting;

	public Servei(Socket _socket, Usuari _user)
	{
		try	{
			this.socket = _socket;
			this.user = _user;
			in=new DataInputStream(socket.getInputStream());
			missatges=new Vector();
			fil=new Thread(this,"SERVEI");
			fil.start();
		}
		catch(Exception e){
			System.out.println("Constructor del Servei"+e);
		}
	}

	private void tancaConexio(){
		SocketMissatge msg = new SocketMissatge(CLIENT_LOGOUT);
		user.setConectat(false);
		msg.setUser(user);
		Hashtable				cli				=
ServidorMissatger.getClients(msg.getUser().getEmail());
		if (cli!=null){
			ServidorMissatger.enviaClients(msg, cli);
		}
		ServidorMissatger.posaOffline(user);
	}

	public void run()
	{
		byte[] data;
		waiting=false;
		timer=new Utils.Timer(10000);
		while(!acaba)
		{
			try	{
				/* Recepció de missatges del client */
				if (in.available()>0){
					data = new byte[MAX_MISSATGE];
					in.read(data);
					SocketMissatge		msg		=
((SocketMissatge)MissatgerUtils.bytesToObject(data));
					if (msg.getHeader()!=MSG_OK) {
						processa(msg);
						msg = new SocketMissatge(MSG_OK,"");
						ServidorMissatger.enviaMissatgeSocket(msg,
socket);
```

```
                                }
                        }
                        /* Enviament de missatges d'altres clients */
                        if (missatges.size()>0){
                                synchronized(missatges){
                                        for (int a=0;a<missatges.size();a++){
                                                SocketMissatge         msg         =
(SocketMissatge)missatges.elementAt(a);

        ServidorMissatger.enviaMissatge((SocketMissatge)missatges.elementAt(a));
                                                System.out.println("MSG_INFO:
"+msg.getEmail()+" >> "+msg.getDesti());


                                                while (!(msg.getHeader()==MSG_OK)){
                                                        data           =           new
byte[MAX_MISSATGE];
                                                        in.read(data);
                                                        msg                          =
((SocketMissatge)MissatgerUtils.bytesToObject(data));
                                                        if
(!(msg.getHeader()==MSG_OK)) processa(msg);
                                                }
                                        }
                                missatges.clear();
                                }
                        }
                        /* Sincronisme */
                        if (timer.isEstat()){
                                if (!waiting){
                                        /* Enviar sincronisme cada 10s */
                                        SocketMissatge       msg       =       new
SocketMissatge(SYNCRO,"");
                                        msg.setDesti(user.getEmail());
                                        ServidorMissatger.enviaMissatge(msg);
                                        waiting=true;
                                }
                                else {
                                        /* Timeout superat, això no xuta */
                                        new Exception();
                                }
                        }
                        Thread.sleep(SLEEP_TIME);
                }
                catch(Exception e) {
                        acaba = true;
                        timer=null;
                        System.out.println("SCK_ERROR:       "+user.getEmail()+"
desconectat.");
                        e.printStackTrace();
                }
        }//END WHILE
        tancaConexio();
        System.out.println("------- SURT: "+user.getEmail()+" ------");
    }

    public void addMissatge(SocketMissatge msg){
        synchronized(missatges){
                missatges.add(msg);
                //System.out.println("Afegit missatge de "+msg.getEmail()+" a
la cua de missatges. Destí: "+msg.getDesti());
        }
    }
```

```
	public void setAcaba(boolean acaba) {
		this.acaba = acaba;
	}

	private void processa(SocketMissatge msg){
		switch (msg.getHeader()){
		case CLIENT_LOGOUT: acaba=true;
			break;
		case SYNCRO: timer = new Utils.Timer(10000);
				waiting=false;
			break;
		default:
			ServidorMissatger.processaMissatge(msg);
	}
	}
}
```

// ServidorMissatger

```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.InetAddress;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Hashtable;
import java.util.Vector;

import Utils.MissatgerUtils;

import Beans.SocketMissatge;
import Beans.Usuari;

public class ServidorMissatger implements Utils.Constants {

	/**
	 * @param args
	 */

	private static ServerSocket ssocket ;
	/* Llista d'usuaris del servidor */
	private static Hashtable serverList = new Hashtable();
	/* Llista d'usuaris conectats */
    private static Hashtable userList = new Hashtable();
	/* Sockets dels usuaris conectats */
    private static Hashtable socketList = new Hashtable();
    /* Serveis en execució */
    private static Hashtable threadList = new Hashtable();

	public static void main(String[] args) {

		Socket socket=null;;
		byte[] data;
		DataInputStream entrada;
		DataOutputStream sortida;

		/* Carregar usuaris del servidor a la hashtable */
		GestorXML gest = new GestorXML();
		serverList = gest.getUsuarisXML();
```

```
            System.out.println("----------------------------------------------
");
            System.out.println("--------------  Servidor  Missatger  1.0  --------
");
            System.out.println("---------------------------------------------
");

            /* Configurar socket */
            try {
                  ssocket = new ServerSocket(gest.getPort(),BACKLOG);
            } catch (IOException e) {
                  e.printStackTrace();
                  System.out.println("Error   al   obrir   socket   al   port
"+gest.getPort());
            }

            /* Obrir thread al arribar conexió */
            while(true){
                  try {
                        socket=ssocket.accept();
                        if (socket!=null){

                              entrada                  =                  new
DataInputStream(socket.getInputStream());
                              sortida                  =                  new
DataOutputStream(socket.getOutputStream());

                              System.out.println ("------- NOVA CONEXIÓ -------
------------------");
                              System.out.println
((socket.getInetAddress()).getHostName()+"@"+(socket.getInetAddress()).getHostAdd
ress()+":"+socket.getPort());

                              /* Recepció de la classe Usuari per identificar
el client */
                              data = new byte[MAX_MISSATGE];
                              entrada.read(data);
                              SocketMissatge              msg              =
((SocketMissatge)MissatgerUtils.bytesToObject(data));
                              Usuari user = msg.getUser();

                              /* Comprova email i password i retorna VALID o
INVALID */
                              if (validaUsuari(user)){

                                    /* Borrar el password, ja no farà falta */
                                    user.setPassword("");

                                    /* Enviar missatge de confirmació: USEROK
*/
                                    msg = new SocketMissatge(USER_OK,"");
                                    msg.setDesti(user.getEmail().trim());
                                    enviaMissatgeSocket(msg, socket);

                                    /* Rebre llista de contactes del client i
afegir a l'usuari */
                                    entrada.read(data);
                                    msg                                    =
((SocketMissatge)MissatgerUtils.bytesToObject(data));
                                    user.setUClients(msg.getLlistaUsuaris());

      user.setIP((socket.getInetAddress()).getHostAddress());
```

```
                                        /* Enviar estat de contactes */
                                        msg = new SocketMissatge(USUARIS_LIST,"");
                                        msg.setUser(user);
                                        msg.setDesti(user.getEmail().trim());
                                        Hashtable               ht               =
getEstatContactes(user.getUClients(), user);
                                        msg.setLlistaUsuaris(ht);
                                        enviaMissatgeSocket(msg, socket);

                                        /* Rebre LIST_OK */
                                        entrada.read(data);
                                        msg                                      =
((SocketMissatge)MissatgerUtils.bytesToObject(data));
                                        if (msg.getHeader()==LIST_OK){

                                                /*   Donar   d'alta   el   socket   i
l'usuari */
                                                synchronized(socketList)  {

        socketList.put(msg.getEmail(),socket);
                                                }
                                                posaOnline(user);
                                                System.out.println         ("-------
CONEXIÓ ESTABLERTA -------------------");


                                                /*  Enviar  informació  d'online  als
clients */
                                                msg              =              new
SocketMissatge(CLIENT_LOGIN,"");
                                                msg.setUser(user);
                                                enviaClients(msg,
getClients(msg.getUser().getEmail()));

                                                /* Obre el servei */
                                                threadList.put(user.getEmail(),new
Servei(socket, msg.getUser()));
                                        }
                                        else{
                                                System.out.println         ("-------
TANCADA: Error al enviar contactes ---");
                                                try {
                                                        socket.close();
                                                } catch (IOException e1) {
                                                        // Res
                                                }
                                        }
                                }
                                else {

                                        /*  Enviar  missatge  de  cancelació:  USERKO
*/
                                        System.out.println   ("-------   REFUSADA:
Usuari no vàlid ------------");
                                        msg = new SocketMissatge(USER_KO,"");
                                        enviaMissatgeSocket(msg, socket);

                                        try {
                                                socket.close();
                                        } catch (IOException e1) {
                                                // res
                                        }
```

```
                                }
                        }

                } catch (Exception e) {
                        System.out.println   ("-------   REFUSADA:   Conexió   no
vàlida ---------");
                        try {
                                socket.close();
                        } catch (IOException e1) {
                                // Res
                        }
                }
        }
    }

    public static void posaOnline(Usuari user)
    {
        synchronized(userList) {
            user.setEstat(ONLINE);
            userList.put(user.getEmail().trim(),user);
        }
    }

    public static void posaEstat(String user, int Estat)
    {
        synchronized(userList) {
            Usuari usr = (Usuari) userList.get(user.trim());
            if (usr!=null){
                usr.setEstat(Estat);
                userList.put(usr.getEmail().trim(),usr);
            }
        }
    }

    public static void posaOffline(Usuari user)
    {
        try {
            /* Tanco el socket */
            Socket                        socket                        =
(Socket)socketList.get(user.getEmail().trim());
            if (socket!=null){
                if (socket.isConnected()) socket.close();
            }
            userList.remove(user.getEmail().trim());
            socketList.remove(user.getEmail().trim());
            threadList.remove(user.getEmail().trim());
        }
        catch(Exception e) {
            System.out.println("Error al destruïr el socket "+e);
        }
    }

    public static boolean validaUsuari(Usuari user){
        /* Buscar l'usuari al vector Usuaris */
        Usuari usr;
        /* Mirem que no estigui conectat */
        synchronized(userList) {
            usr = (Usuari)userList.get(user.getEmail().trim());
        }
        if (usr==null){
            synchronized(serverList) {
                usr = (Usuari)serverList.get(user.getEmail().trim());
            }
```

```
                    if (usr!=null){
                          /* Hem trobat l'usuari */
                          if
(usr.getPassword().trim().compareTo(user.getPassword().trim())==0){
                                /* Contrassenya OK */
                                return true;
                          }
                    }
             }
             return false;
      }

      public static synchronized void processaMissatge(SocketMissatge msg)
      {
             Hashtable cli=null;
             switch(msg.getHeader()) {
                    case CHAT:
                          /* Busca el servei destinatari */
                          Servei           servei           =           (Servei)
threadList.get(msg.getDesti());
                          if (servei!=null){
                                /* Envia el missatge a la cua del servei */
                                servei.addMissatge(msg);
                          }
                          else{
                                System.out.println("MSG_ERROR:        Destinatari
desconegut: "+msg.getDesti());
                          }
                    case CLIENT_LOGIN:
                          //validaUsuari(msg.getUser());

                    case CANVI_ESTAT:
                          posaEstat(msg.getEmail(),msg.getEstat());
                          cli = getClients(msg.getEmail());
                          enviaClients(msg, cli);
                          break;

                    case CLIENT_LOGOUT:
                          /* Si el client està a la pròpia llista no li enviem
res */
                          cli = getClients(msg.getUser().getEmail());
                          cli.remove(msg.getUser().getEmail());
                          /* Envia finalització */
                          enviaClients(msg, cli);
                          posaOffline(msg.getUser());
                          break;

                    default:
                          enviaClients(msg,
getClients(msg.getUser().getEmail()));
             }
      }

      public static Hashtable getClients(String user){
             synchronized(userList){
                    Usuari usr = (Usuari) userList.get(user.trim());
                    if (usr!=null){
                          /* Usuari trobat */
                          return (usr.getUClients());
                    }
             }
             //System.out.println("GetClients retorna null :(");
             return null;
```

```
	}

	public static void enviaClients(SocketMissatge msg, Hashtable Clients){
		byte[] data;
		DataOutputStream out;
		Vector vClients = new Vector(Clients.values());

		/* Recórrer vector de clients destinataris */
		for (int a=0;a<vClients.size();a++){
			Usuari usr = (Usuari) vClients.elementAt(a);
			/* Mirem si el client està admès i cal informar-li de que ens
hem conectat */
			if (usr.isAdmes()){
				/* Mirem si el client està conectat */
				Usuari		server		=		(Usuari)
userList.get(usr.getEmail().trim());
				if (server!=null){
					/* Busquem el socket de la conexió */
					Socket				_socket=(Socket)
socketList.get(usr.getEmail().trim());
					if (_socket!=null){
						try {
							/* Enviem el missatge */
							out=new
DataOutputStream(_socket.getOutputStream());

	data=MissatgerUtils.objectToBytes(msg);
							out.write(data,0,data.length);
							out.flush();
						}
						catch(java.net.SocketException se){
							/* No fer res*/
						}
						catch(Exception e) {
							System.out.println("Error		al
enviar"+e);
						}
					}
				}
			}
		}
	}

	public static Hashtable getEstatContactes(Hashtable husers, Usuari usr){
		/* Recorre tots els usuaris de la llista de l'usuari */
		Vector users = new Vector(husers.values());
		for (int a=0;a<users.size();a++){

			/* Mira si l'usuari està online al servidor */
			Usuari user=(Usuari) (users.elementAt(a));
			Usuari server=(Usuari)userList.get(user.getEmail().trim());

			if (server!=null) {
				/* Ara cerca si el té en llista */
				Usuari		uadmes		=		(Usuari)
server.getUClients().get(usr.getEmail().trim());
				if (uadmes!=null){
					/* El té admès? */
					if (uadmes.isAdmes()){
						user.setConectat(true);
						user.setEstat(server.getEstat());
						user.setNick(server.getNick());
						husers.put(user.getEmail().trim(),user);
```

```
                                }
                        }
                }
        }
        return husers;
    }

    public static synchronized void enviaMissatge(SocketMissatge msg)
    {
        byte[] data;
        DataOutputStream out;
        Socket _socket = (Socket) socketList.get(msg.getDesti());
        //System.out.println(msg.getEmail()+" envia a "+msg.getDesti());
        if (_socket!=null){
            try {
                out=new DataOutputStream(_socket.getOutputStream());
                data=MissatgerUtils.objectToBytes(msg);
                out.write(data,0,data.length);
                out.flush();
            }
            catch(Exception e) {
                System.out.println("Socket error (enviaMissatge)");
                Servei sv=null;
                if (msg!=null) {
                    if (msg.getUser()!=null){
                        sv                =                (Servei)
threadList.get(msg.getUser().getEmail());
                    }
                    else if (msg.getDesti()!=null){
                        sv                =                (Servei)
threadList.get(msg.getDesti());
                    }
                }
                if (sv!=null) sv.setAcaba(true);
            }
        }
        else {
            System.out.println("MSG_ERROR:     Destinatari     desconegut:
"+msg.getDesti());
        }
    }
    public  static  synchronized  void  enviaMissatgeSocket(SocketMissatge  msg,
Socket _socket)
    {
        byte[] data;
        DataOutputStream out;
        try {
            out=new DataOutputStream(_socket.getOutputStream());
            data=MissatgerUtils.objectToBytes(msg);
            out.write(data,0,data.length);
            //out.flush();
        }
        catch(Exception e) {
            System.out.println("SCK_ERROR:   El   socket   falla,   tancant
conexió...");
            Servei sv = (Servei) threadList.get(msg.getUser().getEmail());
            sv.setAcaba(true);
        }
    }
}
```

Paquet SWT

// AlertDialog

```
import Utils.Constants;

import com.cloudgarden.resource.SWTResourceManager;

import org.eclipse.swt.layout.FillLayout;
import org.eclipse.swt.layout.GridData;
import org.eclipse.swt.layout.GridLayout;
import org.eclipse.swt.widgets.Button;
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Label;
import org.eclipse.swt.widgets.Shell;
import org.eclipse.swt.SWT;
import org.eclipse.swt.custom.CLabel;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;

/**
* This code was generated using CloudGarden's Jigloo
* SWT/Swing GUI Builder, which is free for non-commercial
* use. If Jigloo is being used commercially (ie, by a corporation,
* company or business for any purpose whatever) then you
* should purchase a license for each developer using Jigloo.
* Please visit www.cloudgarden.com for details.
* Use of Jigloo implies acceptance of these licensing terms.
* ************************************
* A COMMERCIAL LICENSE HAS NOT BEEN PURCHASED
* for this machine, so Jigloo or this code cannot be used legally
* for any corporate or commercial purpose.
* ************************************
*/
public class AlertDialog extends org.eclipse.swt.widgets.Dialog implements
Constants {

	private Shell dialogShell;
	private Button btnTanca;
	private CLabel clblInfo;
	private String message;
	private String tittle;

	/**
	* Auto-generated main method to display this
	* org.eclipse.swt.widgets.Dialog inside a new Shell.
	*/

	public AlertDialog(int type, String msg) {
		super(MissatgerSWT.getGShell(), SWT.NULL);
		this.message=msg;
		switch (type){
			case ERROR: this.tittle="Error";
			case INFO: this.tittle="Informació";
		}
		open();
	}

	public void open() {
		try {
			Shell parent = getParent();
			dialogShell = new Shell(parent, SWT.DIALOG_TRIM |
SWT.APPLICATION_MODAL);
```

```
                        {
                                //Register     as     a     resource     user     -
SWTResourceManager will
                                //handle the obtaining and disposing of resources

        SWTResourceManager.registerResourceUser(dialogShell);
                        }


                    dialogShell.setLayout(null);
                    dialogShell.layout();
                    dialogShell.pack();
                    dialogShell.setSize(282, 134);
                    dialogShell.setText(tittle);
                    {
                            btnTanca   =   new   Button(dialogShell,   SWT.PUSH   |
SWT.CENTER);
                            btnTanca.setText("Tanca");
                            btnTanca.setBounds(215, 80, 55, 23);
                            btnTanca.addSelectionListener(new SelectionAdapter() {
                                    public void widgetSelected(SelectionEvent evt) {
                                            btnTancaWidgetSelected(evt);
                                    }
                            });
                    }
                    {
                            clblInfo = new CLabel(dialogShell, SWT.NONE);
                            clblInfo.setText(message);
                            clblInfo.setBounds(11, 10, 256, 63);
                    }
                    dialogShell.open();
                    Display display = dialogShell.getDisplay();
                    while (!dialogShell.isDisposed()) {
                            if (!display.readAndDispatch())
                                    display.sleep();
                    }
            } catch (Exception e) {
                    e.printStackTrace();
            }
        }

        private void btnTancaWidgetSelected(SelectionEvent evt) {
                dialogShell.getShell().close();
        }

}
```

// ConfigDialog

```
import java.io.IOException;
import java.util.Vector;

import org.eclipse.swt.SWT;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;
import org.eclipse.swt.widgets.Button;
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Group;
import org.eclipse.swt.widgets.Label;
import org.eclipse.swt.widgets.List;
import org.eclipse.swt.widgets.Shell;
import org.eclipse.swt.widgets.Text;
```

```
import Beans.Usuari;
import Missatger.ConfigClient;
import Missatger.GestorXML;

import com.cloudgarden.resource.SWTResourceManager;

/**
* This code was generated using CloudGarden's Jigloo
* SWT/Swing GUI Builder, which is free for non-commercial
* use. If Jigloo is being used commercially (ie, by a corporation,
* company or business for any purpose whatever) then you
* should purchase a license for each developer using Jigloo.
* Please visit www.cloudgarden.com for details.
* Use of Jigloo implies acceptance of these licensing terms.
* ************************************
* A COMMERCIAL LICENSE HAS NOT BEEN PURCHASED
* for this machine, so Jigloo or this code cannot be used legally
* for any corporate or commercial purpose.
* ************************************
*/
public class ConfigDialog extends org.eclipse.swt.widgets.Dialog {

	private Shell dialogShell;
	private Button btnSave;
	private Text txtNick;
	private Text txtMail;
	private Text txtPassword;
	private Label lblNick;
	private Text txtEmail;
	private Text txtNom;
	private Label label1;
	private Button chkAdmes;
	private Label lblNom;
	private Group grUsuaris;
	private List listUsers;
	private Label lblPassword;
	private Label lblEmail;
	private Label lblPort;
	private Label lblIP;
	private Text txtIP;
	private Text txtPort;
	private Button btnGenera;
	private Button btnModifica;
	private Button btnElimina;
	private Group grpConect;
	private Button btnCancel;
	private ConfigClient users;
	private GestorXML gest;
	private Vector lUsers;
	private String saveMail;
	private MissatgerSWT mainform;

	/**
	* Auto-generated main method to display this
	* org.eclipse.swt.widgets.Dialog inside a new Shell.
	*/


	public ConfigDialog(Shell parent, int style, MissatgerSWT _mainform) {
		super(parent, style);
		users = new ConfigClient();
		gest = new GestorXML();
		mainform=_mainform;
```

```
	}

	public void open() {
		try {
			Shell parent = getParent();
			dialogShell    =    new    Shell(parent,    SWT.DIALOG_TRIM    |
SWT.APPLICATION_MODAL);

				{
					//Register     as     a     resource     user     -
SWTResourceManager will
					//handle the obtaining and disposing of resources

	SWTResourceManager.registerResourceUser(dialogShell);
				}

			users = gest.getConfigClient();

			dialogShell.setLayout(null);
			dialogShell.layout();
			dialogShell.pack();
			dialogShell.setSize(420, 349);
			dialogShell.setText("Configuració");
			{
				grpConect = new Group(dialogShell, SWT.NONE);
				grpConect.setLayout(null);
				grpConect.setText("Conexió");
				grpConect.setBounds(10, 11, 389, 103);
				{
					lblIP = new Label(grpConect, SWT.NONE);
					lblIP.setText("IP Servidor");
					lblIP.setBounds(17, 19, 60, 13);
				}
				{
					txtIP = new Text(grpConect, SWT.BORDER);
					txtIP.setBounds(80, 17, 195, 17);
					txtIP.setTextLimit(25);
					txtIP.setText(users.getIPServidor());
				}
				{
					lblPort = new Label(grpConect, SWT.NONE);
					lblPort.setText("Port");
					lblPort.setBounds(289, 21, 20, 13);
				}
				{
					txtPort = new Text(grpConect, SWT.BORDER);
					txtPort.setBounds(316, 19, 48, 17);
					txtPort.setTextLimit(5);
					txtPort.setText(users.getPort());
				}
				{
					lblEmail = new Label(grpConect, SWT.NONE);
					lblEmail.setText("E-mail");
					lblEmail.setBounds(17, 39, 28, 13);
				}
				{
					txtMail = new Text(grpConect, SWT.BORDER);
					txtMail.setBounds(80, 36, 195, 17);
					txtMail.setTextLimit(30);
					txtMail.setText(users.getEMail());
				}
				{
					lblNick = new Label(grpConect, SWT.NONE);
```

```
                                lblNick.setText("Nick");
                                lblNick.setBounds(17, 58, 21, 13);
                        }
                        {
                                txtNick = new Text(grpConect, SWT.BORDER);
                                txtNick.setBounds(79, 55, 195, 17);
                                txtNick.setTextLimit(30);
                                txtNick.setText(users.getNick());

                        }
                        {
                                lblPassword = new Label(grpConect, SWT.NONE);
                                lblPassword.setText("Password");
                                lblPassword.setBounds(16, 77, 48, 13);
                        }
                        {
                                txtPassword = new Text(grpConect, SWT.BORDER);
                                txtPassword.setBounds(79, 74, 84, 18);
                                txtPassword.setTextLimit(10);
                                txtPassword.setText(users.getPassword());
                        }
                        {
                                btnSave  =  new  Button(grpConect,  SWT.PUSH  |
SWT.CENTER);
                                btnSave.setText("Desa");
                                btnSave.setBounds(307, 73, 69, 21);
                                btnSave.addSelectionListener(new
SelectionAdapter() {
                                        public  void  widgetSelected(SelectionEvent
evt) {
                                                btnSaveWidgetSelected(evt);
                                        }
                                });
                        }
                }
                {
                        grUsuaris = new Group(dialogShell, SWT.NONE);
                        grUsuaris.setLayout(null);
                        grUsuaris.setText("Usuaris");
                        grUsuaris.setBounds(12, 123, 386, 162);
                        {
                                listUsers  =  new  List(grUsuaris,  SWT.V_SCROLL  |
SWT.BORDER);
                                listUsers.setBounds(14, 21, 172, 135);
                                listUsers.setFont(SWTResourceManager.getFont("MS
Sans Serif",9,0,false,false));
                                listUsers.addSelectionListener(new
SelectionAdapter() {
                                        public  void  widgetSelected(SelectionEvent
evt) {
                                                listUsersWidgetSelected(evt);
                                        }
                                });
                                ompleEmails();
                        }
                        {
                                lblNom = new Label(grUsuaris, SWT.NONE);
                                lblNom.setText("Nom");
                                lblNom.setBounds(203, 25, 35, 15);
                        }
                        {
                                txtNom = new Text(grUsuaris, SWT.BORDER);
                                txtNom.setBounds(203, 41, 175, 18);
```

```
                txtNom.setEnabled(false);
            }
            {
                label1 = new Label(grUsuaris, SWT.NONE);
                label1.setText("Email");
                label1.setBounds(204, 61, 164, 14);
            }
            {
                txtEmail = new Text(grUsuaris, SWT.BORDER);
                txtEmail.setBounds(201, 76, 175, 18);
                txtEmail.setEnabled(false);
            }
            {
                chkAdmes  =  new  Button(grUsuaris,  SWT.CHECK  |
SWT.LEFT);
                chkAdmes.setBounds(205, 99, 132, 19);
                chkAdmes.setText("Admès");
                chkAdmes.setEnabled(false);
            }
            {
                btnElimina  =  new  Button(grUsuaris,  SWT.PUSH  |
SWT.CENTER);
                btnElimina.setText("Elimina");
                btnElimina.setBounds(287, 127, 59, 21);
                btnElimina.setEnabled(false);
                btnElimina.addSelectionListener(new
SelectionAdapter() {
                    public  void  widgetSelected(SelectionEvent
evt) {
                        btnEliminaWidgetSelected(evt);
                    }
                });
            }
            {
                btnModifica  =  new  Button(grUsuaris,  SWT.PUSH  |
SWT.CENTER);
                btnModifica.setText("Modifica");
                btnModifica.setBounds(221, 127, 59, 21);
                btnModifica.setEnabled(false);
                btnModifica.addSelectionListener(new
SelectionAdapter() {
                    public  void  widgetSelected(SelectionEvent
evt) {
                        btnModificaWidgetSelected(evt);
                    }
                });
            }
        }
        {
            btnCancel   =   new   Button(dialogShell,   SWT.PUSH   |
SWT.CENTER);
            btnCancel.setText("Sortir");
            btnCancel.setBounds(335, 294, 60, 23);
            btnCancel.addSelectionListener(new SelectionAdapter() {
                public void widgetSelected(SelectionEvent evt) {
                    btnCancelWidgetSelected(evt);
                }
            });
        }
        {
            btnGenera   =   new   Button(dialogShell,   SWT.PUSH   |
SWT.CENTER);
            btnGenera.setText("Genera XML");
```

```
                        btnGenera.setBounds(131, 295, 105, 19);
                        btnGenera.setVisible(false);
                }
                dialogShell.open();
                Display display = dialogShell.getDisplay();
                while (!dialogShell.isDisposed()) {
                        if (!display.readAndDispatch())
                                display.sleep();
                }
        } catch (Exception e) {
                e.printStackTrace();
        }
}

private void btnSaveWidgetSelected(SelectionEvent evt) {
        users.setIPServidor(txtIP.getText());
        users.setEMail(txtMail.getText());
        users.setPort(txtPort.getText());
        users.setNick(txtNick.getText());
        users.setPassword(txtPassword.getText());
        gest.setConfigClient(users);
}

private void btnCancelWidgetSelected(SelectionEvent evt) {
        mainform.actualitzaNick();
        dialogShell.getShell().close();
}

private void listUsersWidgetSelected(SelectionEvent evt) {
        int ind[]=listUsers.getSelectionIndices();
        Usuari oUser = (Usuari) lUsers.get(ind[0]);
        txtEmail.setEnabled(true);
        txtEmail.setText(oUser.getEmail());
        saveMail = oUser.getEmail();
        //txtNom.setEnabled(true);
        txtNom.setText(oUser.getNick());
        chkAdmes.setEnabled(true);
        chkAdmes.setSelection(oUser.isAdmes());
        txtEmail.setEnabled(true);
        btnModifica.setEnabled(true);
        btnElimina.setEnabled(true);
}

private void btnModificaWidgetSelected(SelectionEvent evt) {
        Vector vUsuaris = users.getUsuaris();
        int a = 0;
        while                                                   (((Usuari)
vUsuaris.elementAt(a)).getEmail().compareTo(saveMail)!=0) a++;
        saveMail=txtEmail.getText();
        ((Usuari) vUsuaris.elementAt(a)).setEmail(txtEmail.getText());
        ((Usuari) vUsuaris.elementAt(a)).setAdmes(chkAdmes.getSelection());
        gest.setConfigClient(users);
        ompleEmails();
}

private void btnEliminaWidgetSelected(SelectionEvent evt) {
        Vector vUsuaris = users.getUsuaris();
        int a = 0;
        while                                                   (((Usuari)
vUsuaris.elementAt(a)).getEmail().compareTo(saveMail)!=0) a++;
        vUsuaris.remove(a);
        users.setUsuaris(vUsuaris);
        btnElimina.setEnabled(false);
```

```
		btnModifica.setEnabled(false);
		txtEmail.setText("");
		txtEmail.setEnabled(false);
		txtNom.setText("");
		txtNom.setEnabled(false);
		chkAdmes.setEnabled(false);
		chkAdmes.setSelection(false);
		gest.setConfigClient(users);
		ompleEmails();
	}

	private void ompleEmails(){
		listUsers.removeAll();
		lUsers = users.getUsuaris();
		for (int a = 0; a < lUsers.size(); a++) {
			listUsers.add(((Usuari) (lUsers.get(a))).getEmail());
		}
	}
}
```

// ConversaApp

```
import org.eclipse.swt.SWT;
import org.eclipse.swt.events.ControlAdapter;
import org.eclipse.swt.events.ControlEvent;
import org.eclipse.swt.events.DisposeEvent;
import org.eclipse.swt.events.DisposeListener;
import org.eclipse.swt.events.KeyAdapter;
import org.eclipse.swt.events.KeyEvent;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;
import org.eclipse.swt.graphics.Point;
import org.eclipse.swt.widgets.Button;
import org.eclipse.swt.widgets.Composite;
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Label;
import org.eclipse.swt.widgets.ScrollBar;
import org.eclipse.swt.widgets.Shell;
import org.eclipse.swt.widgets.Text;

import Beans.SocketMissatge;
import Beans.Usuari;
import Missatger.ConexioClient;
import Missatger.GestorXML;

import com.cloudgarden.resource.SWTResourceManager;

/**
* This code was generated using CloudGarden's Jigloo
* SWT/Swing GUI Builder, which is free for non-commercial
* use. If Jigloo is being used commercially (ie, by a corporation,
* company or business for any purpose whatever) then you
* should purchase a license for each developer using Jigloo.
* Please visit www.cloudgarden.com for details.
* Use of Jigloo implies acceptance of these licensing terms.
* ************************************
* A COMMERCIAL LICENSE HAS NOT BEEN PURCHASED
* for this machine, so Jigloo or this code cannot be used legally
* for any corporate or commercial purpose.
* ************************************
*/
```

```
public class ConversaApp extends org.eclipse.swt.widgets.Composite implements
Utils.Constants {

	{
		//Set Look & Feel
		try {

	javax.swing.UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.Windo
wsLookAndFeel");
		} catch(Exception e) {
			e.printStackTrace();
		}
	}

	private Button btnEnvia;
	private Button btnTanca;
	private Text txtEnvia;

	private MissatgerSWT mainApp;
	private Label lblImage;
	private Text txtConversa;
	private String error;
	private boolean conectat;
	private ConexioClient conexio;
	private Usuari local_user;
	private Usuari remote_user;
	private boolean sync;
	public Shell shell;

	{
		//Register as a resource user - SWTResourceManager will
		//handle the obtaining and disposing of resources
		SWTResourceManager.registerResourceUser(this);
	}

	public ConversaApp(Composite parent, int style, MissatgerSWT form, Usuari
user) {
		super(parent, style);
		shell=(Shell) parent;
		Display display=shell.getDisplay();
		mainApp=form;
		initGUI();
		conexio = mainApp.getConexio();
		local_user = conexio.getUser();
		remote_user = user;
		GestorXML oGest = new GestorXML();

	local_user.setPublicKey(oGest.getUsuariClientPK(local_user.getEmail()));
		shell.open();
	}

	/**
	* Initializes the GUI.
	*/
	private void initGUI() {
		try {
			this.setSize(548, 274);
			this.setBackground(SWTResourceManager.getColor(255,	255,
255));
			this.setLayout(null);
			this.addDisposeListener(new DisposeListener() {
				public void widgetDisposed(DisposeEvent evt) {
					rootWidgetDisposed(evt);
```

```
                        }
                }});
                this.addControlListener(new ControlAdapter() {
                        public void controlResized(ControlEvent evt) {
                                rootControlResized(evt);
                        }
                });
                this.layout();
        } catch (Exception e) {
                e.printStackTrace();
        }

    getShell().setImage(SWTResourceManager.getImage("Icons/Conversa.gif"));
        Point pp = new Point(548, 300);
        {
                txtEnvia = new Text(this, SWT.MULTI
                        | SWT.WRAP
                        | SWT.V_SCROLL
                        | SWT.BORDER);
                txtEnvia.setBounds(13, 187, 412, 75);
                txtEnvia.setBackground(SWTResourceManager.getColor(208,   222,
242));
                txtEnvia.setFont(SWTResourceManager.getFont("Tahoma",   9,   0,
false, false));
                txtEnvia.setTextLimit(1024);
                txtEnvia.addKeyListener(new KeyAdapter() {
                        public void keyReleased(KeyEvent evt) {
                                txtEnviaKeyReleased(evt);
                        }
                });
        }
        {
                btnEnvia = new Button(this, SWT.PUSH | SWT.CENTER);
                btnEnvia.setText("Envia");
                btnEnvia.setBounds(437, 218, 95, 39);
                btnEnvia.addSelectionListener(new SelectionAdapter() {
                        public void widgetSelected(SelectionEvent evt) {
                                btnEnviaWidgetSelected(evt);
                        }
                });
        }
        {
                btnTanca = new Button(this, SWT.PUSH | SWT.CENTER);
                btnTanca.setText("Tanca");
                btnTanca.setBounds(437, 187, 95, 24);
                btnTanca.addSelectionListener(new SelectionAdapter() {
                        public void widgetSelected(SelectionEvent evt) {
                                btnTancaWidgetSelected(evt);
                        }
                });
        }
        {
                txtConversa = new Text(this, SWT.MULTI
                        | SWT.WRAP
                        | SWT.V_SCROLL
                        | SWT.BORDER);
                txtConversa.setBounds(11, 14, 413, 164);
                txtConversa.setFont(SWTResourceManager.getFont("Tahoma", 9, 0,
false, false));
                txtConversa.setBackground(SWTResourceManager.getColor(223,
230, 228));
                txtConversa.setEditable(false);
        }
```

```
		{
			lblImage = new Label(this, SWT.NONE);
			lblImage.setBounds(432, 18, 100, 103);

	lblImage.setImage(SWTResourceManager.getImage("Icons/Messenger.jpg"));
		}

		getShell().setSize(pp);
		getShell().
		getShell().open();

	}

	private void btnEnviaWidgetSelected(SelectionEvent evt) {
		if (txtEnvia.getText().length()>0){
			enviarDades();
		}
	}

	private void enviarDades(){
		/* Crea missatge */
		SocketMissatge missatge = new SocketMissatge(CHAT,"");
		missatge.setUser(local_user);
		missatge.setDesti(remote_user.getEmail());
		missatge.setMissatge(txtEnvia.getText());
		/* Afegeix-lo a la cua */

		mainApp.addMissatge(missatge);

	txtConversa.setSelection(txtConversa.getText().length()+1,txtConversa.getTe
xt().length()+1);
		txtConversa.insert(local_user.getNick()+" diu:\n");
		txtConversa.insert(txtEnvia.getText());
		txtConversa.insert("\n");
		/* Llest per enviar nous missatges */

		txtEnvia.setText("");txtEnvia.redraw();
	}

	private void rootWidgetDisposed(DisposeEvent evt) {
		mainApp.alliberaChat(this);
	}

	private void btnTancaWidgetSelected(SelectionEvent evt) {
		setSync(false);
		setConectat(false);
		Thread tSortir = new Thread(){
			public void run(){
				/* Esperar desconexió */
				while (isSync()) {
					try {
						Thread.sleep(200);
					} catch (InterruptedException e) {
						e.printStackTrace();
					}
				}
				/* Finalitzar */
				shell.getDisplay().syncExec(new Runnable() {
					 public void run () {
						getShell().close();
					}
				  });
			}
```

```
		};
		tSortir.start();
	}

	public void addMissatge(SocketMissatge msg){

	txtConversa.setSelection(txtConversa.getText().length()+1,txtConversa.getTe
xt().length()+1);
		txtConversa.insert(msg.getUser().getNick()+" diu:\n");
		txtConversa.insert(msg.getMissatge());
		txtConversa.insert("\n");
		ScrollBar vb = txtConversa.getVerticalBar();

	}

	public boolean isConectat() {
		return conectat;
	}

	public void setConectat(boolean conectat) {
		this.conectat = conectat;
	}

	public String getError() {
		return error;
	}

	public void setError(String error) {
		this.error = error;
	}

	public boolean isSync() {
		return sync;
	}

	public void setSync(boolean sync) {
		this.sync = sync;
	}

	public Usuari getRemote_user() {
		return remote_user;
	}

	private void txtEnviaKeyReleased(KeyEvent evt) {
		if (evt.keyCode == 13) {
			if (txtEnvia.getText().length()>0){
				/* Ha apretat ENTER, enviem */
				enviarDades();
			}
		}
	}

	private void rootControlResized(ControlEvent evt) {
		System.out.println("Quina merda");
		//this.setSize(548, 274);
	}
}
```

// MissatgerSWT

```
import java.util.Hashtable;
import java.util.Vector;
```

```
import org.eclipse.swt.SWT;
import org.eclipse.swt.custom.CCombo;
import org.eclipse.swt.events.ControlAdapter;
import org.eclipse.swt.events.ControlEvent;
import org.eclipse.swt.events.DisposeEvent;
import org.eclipse.swt.events.DisposeListener;
import org.eclipse.swt.events.MouseAdapter;
import org.eclipse.swt.events.MouseEvent;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;
import org.eclipse.swt.graphics.Point;
import org.eclipse.swt.graphics.Rectangle;
import org.eclipse.swt.layout.FillLayout;
import org.eclipse.swt.widgets.Composite;
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Label;
import org.eclipse.swt.widgets.Menu;
import org.eclipse.swt.widgets.MenuItem;
import org.eclipse.swt.widgets.Shell;
import org.eclipse.swt.widgets.Tree;
import org.eclipse.swt.widgets.TreeItem;

import Beans.SocketMissatge;
import Beans.Usuari;
import Missatger.ConexioClient;
import Missatger.ConfigClient;
import Missatger.GestorXML;

import com.cloudgarden.resource.SWTResourceManager;


/**
* This code was generated using CloudGarden's Jigloo
* SWT/Swing GUI Builder, which is free for non-commercial
* use. If Jigloo is being used commercially (ie, by a corporation,
* company or business for any purpose whatever) then you
* should purchase a license for each developer using Jigloo.
* Please visit www.cloudgarden.com for details.
* Use of Jigloo implies acceptance of these licensing terms.
* ************************************
* A COMMERCIAL LICENSE HAS NOT BEEN PURCHASED
* for this machine, so Jigloo or this code cannot be used legally
* for any corporate or commercial purpose.
* ************************************
*/
public class MissatgerSWT extends org.eclipse.swt.widgets.Composite implements
Utils.Constants {

	private Menu menu1;
	private MenuItem sobreMenuItem;
	private Menu helpMenu;
	private MenuItem helpMenuItem;
	private MenuItem sortirFileMenuItem;
	private MenuItem configuraFileMenuItem;
	private MenuItem desconectaFileMenuItem;
	private MenuItem conectaFileMenuItem;
	private Tree tree1;
	private CCombo ccEstat;
	private Label lblNick;
	private Label lblEstat;
	private MenuItem sepmenuItem2;
	private MenuItem sepmenuItem1;
```

```
	private Composite cContactos;
	private Menu fileMenu;
	private MenuItem fileMenuItem;
	private boolean Conectat;
	private boolean conSync;
	private ConexioClient conexio;
	private Hashtable usersOn;
	private Hashtable usersOff;
	private Hashtable appsChat;
	private Vector cuaMissatges;
	private static Shell gShell;
	private int Estat;
	private GestorXML oGest;
	private Label lblMessenger;

	{
		//Register as a resource user - SWTResourceManager will
		//handle the obtaining and disposing of resources
		SWTResourceManager.registerResourceUser(this);
	}

	public MissatgerSWT(Composite parent, int style) {
		super(parent, style);
		this.gShell = (Shell) parent;
		appsChat = new Hashtable();
		cuaMissatges = new Vector();
		Estat=OFFLINE;
		oGest = new GestorXML();
		initGUI();
	}

	/**
	* Initializes the GUI.
	*/
	private void initGUI() {
		try {
			this.setSize(259, 405);
			getShell().setBounds(300,150,245,350);
			this.setBackground(SWTResourceManager.getColor(255,	255,
255));
			this.setLayout(null);
			this.addDisposeListener(new DisposeListener() {
				public void widgetDisposed(DisposeEvent evt) {
					rootWidgetDisposed(evt);
				}
			});
			this.addControlListener(new ControlAdapter() {
				public void controlResized(ControlEvent evt) {
					rootControlResized(evt);
				}
			});
			{
				menu1 = new Menu(getShell(), SWT.BAR);
				getShell().setMenuBar(menu1);
				{
					fileMenuItem = new MenuItem(menu1, SWT.CASCADE);
					fileMenuItem.setText("Missatger");
					{
						fileMenu = new Menu(fileMenuItem);
						{
							conectaFileMenuItem	=	new
MenuItem(fileMenu, SWT.CASCADE);
```

```
        conectaFileMenuItem.setText("Conectar-se");
                                                conectaFileMenuItem
                                                        .addSelectionListener(new
SelectionAdapter() {
                                                        public                    void
widgetSelected(SelectionEvent evt) {

        conectaFileMenuItemWidgetSelected(evt);
                                                        }
                                                        });
                                        }
                                        {
                                                desconectaFileMenuItem     =     new
MenuItem(fileMenu, SWT.CASCADE);

        desconectaFileMenuItem.setText("Desconectar");

        desconectaFileMenuItem.setEnabled(false);
                                                desconectaFileMenuItem
                                                        .addSelectionListener(new
SelectionAdapter() {
                                                        public                    void
widgetSelected(SelectionEvent evt) {

        desconectaFileMenuItemWidgetSelected(evt);
                                                        }
                                                        });
                                        }
                                        {
                                                sepmenuItem1          =          new
MenuItem(fileMenu, SWT.SEPARATOR);
                                                sepmenuItem1.setText("menuItem1");
                                        }
                                        {
                                                configuraFileMenuItem      =      new
MenuItem(fileMenu, SWT.CASCADE);

        configuraFileMenuItem.setText("Configuració");
                                                configuraFileMenuItem
                                                        .addSelectionListener(new
SelectionAdapter() {
                                                        public                    void
widgetSelected(SelectionEvent evt) {

        configuraFileMenuItemWidgetSelected(evt);
                                                        }
                                                        });
                                        }
                                        {
                                                sepmenuItem2          =          new
MenuItem(fileMenu, SWT.SEPARATOR);

        sepmenuItem2.setText("sepmenuItem2");
                                        }
                                        {
                                                sortirFileMenuItem       =       new
MenuItem(fileMenu, SWT.CASCADE);

        sortirFileMenuItem.setText("Sortir");
                                                sortirFileMenuItem
                                                        .addSelectionListener(new
SelectionAdapter() {
```

```
                                                                public             void
widgetSelected(
                                                                       SelectionEvent
evt) {

        sortirFileMenuItemWidgetSelected(evt);
                                                                }
                                                        });
                                                }
                                                fileMenuItem.setMenu(fileMenu);
                                        }
                                }
                                {
                                        helpMenuItem = new MenuItem(menu1, SWT.CASCADE);
                                        helpMenuItem.setText("Ajuda");
                                        {
                                                helpMenu = new Menu(helpMenuItem);
                                                {
                                                        sobreMenuItem         =         new
MenuItem(helpMenu, SWT.CASCADE);
                                                        {
                                                                cContactos       =       new
Composite(this, SWT.NONE);
                                                                cContactos.setLayout(null);
                                                                cContactos.setBounds(3,    5,
250, 351);

        cContactos.setBackground(SWTResourceManager.getColor(255, 255, 255));

                                                                {
                                                                        tree1       =       new
Tree(cContactos, SWT.NONE);

        tree1.setVisible(false);
                                                                        tree1.setBounds(2,  14,
247, 326);

        tree1.addMouseListener(new MouseAdapter() {
                                                                       public      void
mouseDoubleClick(

        MouseEvent evt) {

        tree1MouseDoubleClick(evt);
                                                                       }
                                                                }});
                                                                }
                                                                {
                                                                        lblMessenger   =   new
Label(
                                                                       cContactos,
                                                                       SWT.NONE);

        lblMessenger.setBounds(59, 67, 127, 119);

        lblMessenger.setBackground(SWTResourceManager.getColor(255, 255, 255));

        lblMessenger.setImage(SWTResourceManager.getImage("Icons/Messenger.jpg"));
                                                                }
                                                        }
                                                        {
                                                                lblEstat  =  new  Label(this,
SWT.NONE);
```

```
                                                        lblEstat.setBounds(5, 10, 39,
30);

        lblEstat.setImage(SWTResourceManager.getImage("Icons/Estat Offline.jpg"));
                                                }
                                                {
                                                        lblNick   =   new   Label(this,
SWT.NONE);
                                                        lblNick.setBounds(48, 4, 202,
19);

        lblNick.setBackground(SWTResourceManager.getColor(255, 255, 255));
                                                        lblNick.setText("Gerard    ::
Projecte");

        lblNick.setFont(SWTResourceManager.getFont("Tahoma", 11, 1, false, false));
                                                }
                                                {
                                                        ccEstat  =  new  CCombo(this,
SWT.NONE);
                                                        ccEstat.setText("ccEstat");
                                                        ccEstat.setBounds(49,     27,
203, 24);
                                                        ccEstat.setEditable(false);

        ccEstat.setBackground(SWTResourceManager.getColor(255, 255, 255));

        ccEstat.setFont(SWTResourceManager.getFont("Tahoma", 10, 1, false, false));
                                                        ccEstat

        .addSelectionListener(new SelectionAdapter() {
                                                                public             void
widgetSelected(
                                                                        SelectionEvent
evt) {

        ccEstatWidgetSelected(evt);
                                                                }
                                                                });
                                                        ccEstat.add("No conectat");
                                                        ccEstat.add("Conectat");
                                                        ccEstat.select(0);

                                                }

        sobreMenuItem.setText("Informació");
                                                sobreMenuItem
                                                        .addSelectionListener(new
SelectionAdapter() {
                                                        public                   void
widgetSelected(SelectionEvent evt) {

        sobreMenuItemWidgetSelected(evt);
                                                        }
                                                        });
                                        }
                                        helpMenuItem.setMenu(helpMenu);
                                }
                        }
                }

                actualitzaNick();
                ajustar();
```

```
                this.layout();
        } catch (Exception e) {
                e.printStackTrace();
        }
    }

    /**
    * Auto-generated main method to display this
    * org.eclipse.swt.widgets.Composite inside a new Shell.
    */
    public static void main(String[] args) {
        Display display = Display.getDefault();
        Shell shell = new Shell(display);
        MissatgerSWT inst = new MissatgerSWT(shell, SWT.NULL);
        Point size = inst.getSize();
        shell.setLayout(new FillLayout());
        shell.setSize(768, 537);
        shell.setText("Client Missatger");
        shell.setImage(SWTResourceManager.getImage("Icons/Messenger.gif"));
        shell.layout();
        if(size.x == 0 && size.y == 0) {
                inst.pack();
                shell.pack();
        } else {
                Rectangle  shellBounds  =  shell.computeTrim(0,  0,  size.x,
size.y);
                shell.setSize(shellBounds.width, shellBounds.height);
        }
        shell.open();
        while (!shell.isDisposed()) {
                if (!display.readAndDispatch())
                        display.sleep();
        }
    }

    private void rootControlResized(ControlEvent evt) {
        ajustar();
    }
    private void ajustar(){
        // Ajustar cContactos i cMenuInf a la finestra //
        Point mpt1 = this.getSize();
        Rectangle rec1 = cContactos.getBounds();
        Rectangle rec2 = tree1.getBounds();
        Rectangle rec3 = lblEstat.getBounds();
        Rectangle rec4 = ccEstat.getBounds();
        rec1.x=5;
        rec1.y=56;
        rec1.width=mpt1.x-10;
        rec1.height=mpt1.y - 62;
        rec2.x=2;
        rec2.y=2;
        rec2.width=mpt1.x-14;
        rec2.height=mpt1.y-66;
        rec3.width=mpt1.x-56;
        rec4.width=mpt1.x-56;
        cContactos.setBounds(rec1);
        tree1.setBounds(rec2);
        ccEstat.setBounds(rec3);
        ccEstat.setBounds(rec4);
    }

    private void sortirFileMenuItemWidgetSelected(SelectionEvent evt) {
        if (isConectat()){
```

```
                setConSync(true);
                setConectat(false);
                conectaFileMenuItem.setEnabled(true);
                desconectaFileMenuItem.setEnabled(false);
                Thread tSortir = new Thread(){
                        public void run(){
                                /* Esperar desconexió */
                                while (isConSync()) {
                                        try {
                                                Thread.sleep(400);
                                        } catch (InterruptedException e) {
                                                e.printStackTrace();
                                        }
                                }
                                /* Finalitzar */
                                gShell.getDisplay().syncExec(new Runnable() {
                                         public void run () {
                                                        gShell.close();
                                        }
                                  });
                        }
                };
                tSortir.start();
        }
        else {
                Shell shell = this.getShell();
                shell.close();
        }
    }

        private void sobreMenuItemWidgetSelected(SelectionEvent evt) {
                sobreDialog inst = new sobreDialog(this.getShell(), SWT.NULL);
                inst.open();
        }

        private void configuraFileMenuItemWidgetSelected(SelectionEvent evt)
{
                ConfigDialog   inst   =   new   ConfigDialog(this.getShell(),
SWT.NULL, this);
                inst.open();
        }

        private void tree1MouseDoubleClick(MouseEvent evt) {
                Usuari uOnline = getUsuariSelOnline();
                if (uOnline!=null) obreConversa(uOnline);
        }

        public ConversaApp obreConversa(Usuari uOnline){
                /* Busca l'usuari al llistat de converses ja obertes */
                ConversaApp                                     app=(ConversaApp)
appsChat.get(uOnline.getEmail().trim());

                if (app==null){
                        /* No l'ha trobat, obrim nova finestra */
                        if (uOnline!=null) {
                                Display display = Display.getDefault();
                                Shell shell = new Shell(display);
                                shell.setText("Conversa amb "+uOnline.getNick()+"
("+uOnline.getEmail()+")");
                                shell.layout();
                                ConversaApp   chat   =   new   ConversaApp(shell,
SWT.NULL, this, uOnline);
                                /* Afegir la conversa al vector de converses */
```

```
                                appsChat.put(uOnline.getEmail().trim(),chat);
                                return chat;
                        }
                        else{
                                return(null);
                        }
                }
                else {
                        /* Ja hi és, activa la finestra de la conversa */
                        app.setFocus();
                        return app;
                }
        }
        private void conectaFileMenuItemWidgetSelected(SelectionEvent evt) {
                conexio = new ConexioClient(this);
                conexio.start();
                conectaFileMenuItem.setEnabled(false);
                desconectaFileMenuItem.setEnabled(false);
                conectaFileMenuItem.setEnabled(false);
                configuraFileMenuItem.setEnabled(false);
                ccEstat.setEnabled(false);
                sortirFileMenuItem.setEnabled(false);
        }

        private   void   desconectaFileMenuItemWidgetSelected(SelectionEvent
evt) {
                this.Conectat = false;
                conectaFileMenuItem.setEnabled(false);
                desconectaFileMenuItem.setEnabled(false);
                conectaFileMenuItem.setEnabled(false);
                sortirFileMenuItem.setEnabled(false);
                actualitzaCombo(false);
        }
        public void actualitzaMenu(boolean conectat){
                conectaFileMenuItem.setEnabled(!conectat);
                desconectaFileMenuItem.setEnabled(conectat);
                configuraFileMenuItem.setEnabled(!conectat);
                sortirFileMenuItem.setEnabled(true);
                if (conectat){

      lblEstat.setImage(SWTResourceManager.getImage("Icons/Estat Online.jpg"));
                        cContactos.setBackground(SWTResourceManager.getColor(0,
101, 202));
                }
                else{
                        tree1.setVisible(false);

      lblEstat.setImage(SWTResourceManager.getImage("Icons/Estat Offline.jpg"));

      cContactos.setBackground(SWTResourceManager.getColor(255, 255, 255));

                }
                ccEstat.setEnabled(true);
                ccEstat.select(0);
                if (conectat) {
                        Estat=ONLINE;
                }
                else {
                        Estat=OFFLINE;
                }

        }
```

```
            public void generaContactes(Hashtable husers){
                  TreeItem treeItem;
                  TreeItem tiOnline;
                  TreeItem tiOffline;
                  tree1.removeAll();

                  Vector users=new Vector(husers.values());
                  usersOn=new Hashtable();
                  usersOff=new Hashtable();

                  /* Crear items de nivell 1 */
                  tiOnline = new TreeItem(tree1,   SWT.NONE);
                  tiOnline.setText("Usuaris conectats");
                  tiOnline.setFont(SWTResourceManager.getFont("Tahoma",  10,  1,
false, false));

                  tiOffline = new TreeItem(tree1, SWT.NONE);
                  tiOffline.setText("Usuaris no conectats");
                  tiOffline.setFont(SWTResourceManager.getFont("Tahoma",  10,  1,
false, false));

                  /* Recórrer vector d'usuaris */
                  for (int a=0;a<users.size();a++){
                        Usuari user = (Usuari) users.elementAt(a);
                        if (user.isConectat()){
                              usersOn.put(user.getEmail().trim(),user);
                              /* Usuari online */
                              treeItem = new TreeItem(tiOnline, SWT.NONE);
                              if (user.isAdmes()){

      treeItem.setImage(SWTResourceManager.getImage("Icons/Conectat.jpg"));
                              }
                              else{

      treeItem.setImage(SWTResourceManager.getImage("Icons/No admes.jpg"));
                              }
                              treeItem.setText(user.getNick());

      treeItem.setFont(SWTResourceManager.getFont("Tahoma", 9, 0, false, false));

                        }
                        else{
                              /* Usuari offline */
                              usersOff.put(user.getEmail(),user);
                              treeItem = new TreeItem(tiOffline, SWT.NONE);
                              if (user.isAdmes()){

      treeItem.setImage(SWTResourceManager.getImage("Icons/No conectat.jpg"));
                              }
                              else{

      treeItem.setImage(SWTResourceManager.getImage("Icons/No     conectat     no
admes.jpg"));
                              }
                              treeItem.setText(user.getEmail());

      treeItem.setFont(SWTResourceManager.getFont("Tahoma", 9, 0, false, false));

                        }
                  }

                  tree1.setVisible(true);
                  tiOnline.setExpanded(true);
```

```
                tiOffline.setExpanded(true);
                ajustar();
        }

        private void posaUsuariOnline(Usuari user){
                TreeItem treeItem;
                TreeItem tiOnline=tree1.getItem(0);
                TreeItem tiOffline=tree1.getItem(1);

                /* Usuari online */
                Usuari utmp = (Usuari) usersOff.get(user.getEmail().trim());

                if (utmp!= null){
                        treeItem = new TreeItem(tiOnline, SWT.NONE);
                        if (utmp.isAdmes()){

    treeItem.setImage(SWTResourceManager.getImage(getImatgeEstat(user.getEstat(
))));
                        }
                        else{

    treeItem.setImage(SWTResourceManager.getImage("Icons/No admes.jpg"));
                        }
                        treeItem.setText(user.getNick());
                        treeItem.setFont(SWTResourceManager.getFont("Tahoma",
9, 0, false, false));

                        /* L'afegim al vector d'usuaris */
                        utmp.setNick(user.getNick());
                        utmp.setEstat(user.getEstat());
                        usersOn.put(utmp.getEmail().trim(), utmp);

                        /* Donar de baixa l'usuari offline */
                        tiOffline.removeAll();
                        usersOff.remove(utmp.getEmail().trim());
                        Vector tmp = (Vector) new Vector(usersOff.values());

                        /* Tornem a generar els usuaris i el vector en base a
tmp */
                        for (int a=0;a<tmp.size();a++){
                                /* Usuari offline */
                                Usuari tmpUser = (Usuari) tmp.elementAt(a);
                                treeItem = new TreeItem(tiOffline, SWT.NONE);
                                if (tmpUser.isAdmes()){

    treeItem.setImage(SWTResourceManager.getImage("Icons/No conectat.jpg"));
                                }
                                else{

    treeItem.setImage(SWTResourceManager.getImage("Icons/No     conectat     no
admes.jpg"));
                                }
                                treeItem.setText(tmpUser.getEmail());

    treeItem.setFont(SWTResourceManager.getFont("Tahoma", 9, 0, false, false));
                        }
                }
                tiOnline.setExpanded(true);
                tiOffline.setExpanded(true);
        }

        private void posaUsuariOffline(Usuari user){
                TreeItem treeItem;
```

```
                    TreeItem tiOnline=tree1.getItem(0);
                    TreeItem tiOffline=tree1.getItem(1);

                    /* Nou Usuari offline */
                    Usuari utmp = (Usuari) usersOn.get(user.getEmail().trim());

                    if (utmp!= null){
                          treeItem = new TreeItem(tiOffline, SWT.NONE);
                          if (utmp.isAdmes()){

      treeItem.setImage(SWTResourceManager.getImage("Icons/No conectat.jpg"));
                          }
                          else{

      treeItem.setImage(SWTResourceManager.getImage("Icons/No     conectat     no
admes.jpg"));
                          }
                          treeItem.setText(utmp.getEmail());
                          treeItem.setFont(SWTResourceManager.getFont("Tahoma",
9, 0, false, false));

                          /* L'afegim al vector d'usuaris offline */
                          usersOff.put(utmp.getEmail().trim(),utmp);

                          /* Donar de baixa l'usuari online */
                          tiOnline.removeAll();
                          usersOn.remove(utmp.getEmail().trim());
                          Vector tmp = (Vector) new Vector(usersOn.values());

                          /* Tornem a generar els usuaris i el vector en base a
tmp */
                          for (int a=0;a<tmp.size();a++){
                                /* Usuari online */
                                Usuari tmpUser = (Usuari) tmp.elementAt(a);

                                treeItem = new TreeItem(tiOnline, SWT.NONE);
                                if (tmpUser.isAdmes()){

      treeItem.setImage(SWTResourceManager.getImage(getImatgeEstat(tmpUser.getEst
at())));
                                }
                                else{

      treeItem.setImage(SWTResourceManager.getImage("Icons/No admes.jpg"));
                                }
                                treeItem.setText(tmpUser.getNick());

      treeItem.setFont(SWTResourceManager.getFont("Tahoma", 9, 0, false, false));
                                usersOn.put(tmpUser.getEmail().trim(),tmpUser);
                          }
                    }
                    tiOnline.setExpanded(true);
                    tiOffline.setExpanded(true);
              }

              private void posaUsuariEstat(String user, int Estat){
                    TreeItem treeItem;
                    TreeItem tiOnline=tree1.getItem(0);

                    /* Usuari online */
                    Usuari utmp = (Usuari) usersOn.get(user.trim());

                    if (utmp!= null){
```

```
                        /* Existex, modifiquem l'estat al vector */
                        utmp.setEstat(Estat);
                        usersOn.put(user.trim(),utmp);

                        Vector tmp = new Vector(usersOn.values());
                        tiOnline.removeAll();

                        /* Regenerem els usuaris online */
                        for (int a=0;a<tmp.size();a++){
                              /* Usuari online */
                              Usuari tmpUser = (Usuari) tmp.elementAt(a);
                              treeItem = new TreeItem(tiOnline, SWT.NONE);
                              if (tmpUser.isAdmes()){

      treeItem.setImage(SWTResourceManager.getImage(getImatgeEstat(tmpUser.getEst
at())));
                              }
                              else{

      treeItem.setImage(SWTResourceManager.getImage("Icons/No admes.jpg"));
                              }
                              treeItem.setText(tmpUser.getNick());

      treeItem.setFont(SWTResourceManager.getFont("Tahoma", 9, 0, false, false));
                        }

                  }
                  tiOnline.setExpanded(true);
            }

            public Usuari getUsuariSelOnline(){
                  int c=0;
                  int index=-1;
                  Usuari userSel=null;
                  Vector users=new Vector(usersOn.values());

                  TreeItem ti[] = tree1.getSelection();

                  if (ti.length!=0){
                        /* Busquem als online */
                        for (c=0;c<users.size();c++){

      //System.out.println(ti[0].getText()+"="+((Usuari)
users.elementAt(c)).getNick()+":"+(ti[0].getText().trim().compareTo(((Usuari)
users.elementAt(c)).getNick().trim())==0));
                              if    (ti[0].getText().trim().compareTo(((Usuari)
users.elementAt(c)).getNick().trim())==0) index=c;
                        }
                        if       (index>-1)       userSel       =       (Usuari)
users.elementAt(index);
                  }
                  return userSel;
            }

            public boolean isConectat() {
                  return Conectat;
            }

            public void setConectat(boolean conectat) {
                  Conectat = conectat;
            }
```

```
		public boolean isConSync() {
			return conSync;
		}

		public void setConSync(boolean conSync) {
			this.conSync = conSync;
		}

		public void alliberaChat(ConversaApp chat){
			synchronized(appsChat){

	appsChat.remove(chat.getRemote_user().getEmail().trim());
			}
		}

		private void rootWidgetDisposed(DisposeEvent evt) {
			if (isConectat()){
				setConSync(true);
				setConectat(false);
				Thread tSortir = new Thread(){
					public void run(){
						/* Esperar desconexió */
						while (isConSync()) {
							try {
								Thread.sleep(400);
							} catch (InterruptedException e) {
								e.printStackTrace();
							}

						}
					}
				};
				tSortir.start();
			}
		}
		public void processaMissatge(SocketMissatge msg){
			//System.out.println("RECEIVE MESSAGE "+message._header);
			switch(msg.getHeader())	{
				case USUARIS_LIST:
					generaContactes(msg.getLlistaUsuaris());
					//System.out.println("Llista	de	contactes
rebuda");
				break;
				case CLIENT_LOGIN:
					//System.out.println("Alta		usuari:
"+msg.getUser().getEmail());
					posaUsuariOnline(msg.getUser());
				break;
				case CLIENT_LOGOUT:
					//System.out.println("Baixa		usuari:
"+msg.getUser().getEmail());
					posaUsuariOffline(msg.getUser());
				break;
				case CHAT:
					/* Desxifrar el missatge */
					/* Buscar la finestra destinataria */
					ConversaApp	app	=	(ConversaApp)
appsChat.get(msg.getUser().getEmail().trim());
					if (app!=null){
						/* Finestra trobada */
						if (!app.isFocusControl()){
							app.addMissatge(msg);
						}
```

```
                                }
                                else {
                                        /* No s'ha trobat, mirem si l'usuari és
online */
                                        Usuari obre = (Usuari)
usersOn.get(msg.getUser().getEmail().trim());
                                        if (obre!=null){
                                                /* És online, obrim el xat i enviem
missatge */
                                                ConversaApp ap =
obreConversa(obre);
                                                ap.addMissatge(msg);
                                        }
                                        else{
                                                System.out.println("L'emisor del
missatge no està conectat: "+msg.getUser().getEmail());
                                        }
                                }
                                break;
                        case CANVI_ESTAT:
                                posaUsuariEstat(msg.getEmail(),msg.getEstat());
                        break;
                        default:
                        }
                }

                public void canviaEstat(){
                        switch(ccEstat.getSelectionIndex()+1){
                        case OFFLINE:
                                this.Conectat = false;
                                conectaFileMenuItem.setEnabled(false);
                                desconectaFileMenuItem.setEnabled(false);
                                conectaFileMenuItem.setEnabled(false);
                                sortirFileMenuItem.setEnabled(false);
                                actualitzaCombo(false);

        lblEstat.setImage(SWTResourceManager.getImage("Icons/Estat Offline.jpg"));
                                break;
                        case ONLINE:
                                if (!Conectat){
                                        conexio = new ConexioClient(this);
                                        conexio.start();
                                        conectaFileMenuItem.setEnabled(false);
                                        desconectaFileMenuItem.setEnabled(false);
                                        configuraFileMenuItem.setEnabled(false);
                                        ccEstat.setEnabled(false);
                                        conectaFileMenuItem.setEnabled(false);
                                        sortirFileMenuItem.setEnabled(false);
                                        actualitzaCombo(true);
                                }
                                else{
                                        enviaNouEstat(ONLINE);
                                }

        lblEstat.setImage(SWTResourceManager.getImage("Icons/Estat Online.jpg"));
                                break;
                        case AWAY:
                                enviaNouEstat(AWAY);

        lblEstat.setImage(SWTResourceManager.getImage("Icons/Estat Away.jpg"));
                                break;
                        case OCUPAT:
                                enviaNouEstat(OCUPAT);
```

```
lblEstat.setImage(SWTResourceManager.getImage("Icons/Estat Busy.jpg"));
                            break;
                }
                Estat=ccEstat.getSelectionIndex()+1;
        }

        public void enviaNouEstat(int Estat){
                SocketMissatge msg = new SocketMissatge();
                msg.setEmail(oGest.getUsuari().getEmail());
                msg.setHeader(CANVI_ESTAT);
                msg.setEstat(Estat);
                addMissatge(msg);
        }

        public ConexioClient getConexio() {
                return conexio;
        }

        public void setConexio(ConexioClient conexio) {
                this.conexio = conexio;
        }
        public void addMissatge(SocketMissatge msg){
                synchronized(cuaMissatges){
                        cuaMissatges.add(msg);
                }
        }

        public Vector getCuaMissatges() {
                 synchronized(cuaMissatges){
                        return cuaMissatges;
                 }
        }
        public Vector readCuaMissatges() {
                Vector cua = null;
                synchronized(cuaMissatges){
                        cua = new Vector(cuaMissatges);
                        cuaMissatges.clear();
                 }
                 return cua;
        }

        public static Shell getGShell() {
                return gShell;
        }

        private void ccEstatWidgetSelected(SelectionEvent evt) {
                if ((ccEstat.getSelectionIndex()+1)!=Estat){
                        /* Hi ha canvi d'estat */
                        canviaEstat();
                }
        }

        public void actualitzaNick(){
                lblNick.setText(((Usuari)(oGest.getUsuari())).getNick());
        }

        public String getImatgeEstat(int Estat){
                switch(Estat){
                case OFFLINE:
                        return("Icons/No conectat.jpg");
                case ONLINE:
                        return("Icons/Conectat.jpg");
```

```
                case AWAY:
                        return("Icons/Away.jpg");
                case OCUPAT:
                        return("Icons/Busy.jpg");
                default: return("Icons/No conectat.jpg");
                }
        }

        public void actualitzaCombo(boolean Conectat){
                ccEstat.removeAll();
                if (Conectat){
                        ccEstat.add("No conectat");
                        ccEstat.add("Conectat");
                        ccEstat.add("Ausent");
                        ccEstat.add("Ocupat");
                        ccEstat.select(1);
                }
                else{
                        ccEstat.add("No conectat");
                        ccEstat.add("Conectat");
                        ccEstat.select(0);
                }
        }
    }
```

// sobreDialog

```
import org.eclipse.swt.SWT;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;
import org.eclipse.swt.widgets.Button;
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Label;
import org.eclipse.swt.widgets.Shell;

import com.cloudgarden.resource.SWTResourceManager;

/**
* This code was generated using CloudGarden's Jigloo
* SWT/Swing GUI Builder, which is free for non-commercial
* use. If Jigloo is being used commercially (ie, by a corporation,
* company or business for any purpose whatever) then you
* should purchase a license for each developer using Jigloo.
* Please visit www.cloudgarden.com for details.
* Use of Jigloo implies acceptance of these licensing terms.
* ************************************
* A COMMERCIAL LICENSE HAS NOT BEEN PURCHASED
* for this machine, so Jigloo or this code cannot be used legally
* for any corporate or commercial purpose.
* ************************************
*/
public class sobreDialog extends org.eclipse.swt.widgets.Dialog {

    private Shell dialogShell;
    private Button btnClose;
    private Label lblUOC;
    private Label lblTreball;
    private Label lblTitol;
    private Label lblImatge;

    /**
    * Auto-generated main method to display this
    * org.eclipse.swt.widgets.Dialog inside a new Shell.
```

```
	*/
	public static void main(String[] args) {
		try {
			Display display = Display.getDefault();
			display.getActiveShell().setText("Informació");
			Shell shell = new Shell(display);
			sobreDialog inst = new sobreDialog(shell, SWT.NULL);
			inst.open();
		} catch (Exception e) {
			e.printStackTrace();
		}
	}

	public sobreDialog(Shell parent, int style) {
		super(parent, style);
	}

	public void open() {
		try {
			Shell parent = getParent();
			dialogShell    =    new    Shell(parent,    SWT.DIALOG_TRIM    |
SWT.APPLICATION_MODAL);

				{
					//Register     as     a     resource     user     -
SWTResourceManager will
					//handle the obtaining and disposing of resources

	SWTResourceManager.registerResourceUser(dialogShell);
				}


			dialogShell.setLayout(null);
			dialogShell.layout();
			dialogShell.pack();
			dialogShell.setSize(386, 146);
			dialogShell.setText("Informació");
			{
				btnClose   =   new   Button(dialogShell,   SWT.PUSH   |
SWT.CENTER);
				btnClose.setText("Tancar");
				btnClose.setBounds(313, 86, 59, 24);
				btnClose.addSelectionListener(new SelectionAdapter() {
					public void widgetSelected(SelectionEvent evt) {
						btnCloseWidgetSelected(evt);
					}
				});
			}
			{
				lblImatge  =  new  Label(dialogShell,  SWT.SHADOW_NONE  |
SWT.BORDER);
				lblImatge.setBounds(14, 16, 60, 62);

	lblImatge.setImage(SWTResourceManager.getImage("Icons/Icona_Informacio.gif"
));
			}
			{
				lblTitol = new Label(dialogShell, SWT.NONE);
				lblTitol.setText("Client Missatger. Versió 1.0.");
				lblTitol.setBounds(100, 15, 251, 15);
				lblTitol.setFont(SWTResourceManager.getFont("Tahoma",
9, 1, false, false));
			}
```

```
                    {
                          lblTreball = new Label(dialogShell, SWT.NONE);
                          lblTreball.setText("Treball fi de carrera 2005/06");
                          lblTreball.setBounds(99, 31, 265, 17);
                    }
                    {
                          lblUOC = new Label(dialogShell, SWT.NONE);
                          lblUOC.setText("Universitat    Oberta    de    Catalunya
(UOC)");
                          lblUOC.setBounds(100, 56, 237, 16);
                          lblUOC.setFont(SWTResourceManager.getFont("Tahoma",   9,
1, false, false));
                          lblUOC.setForeground(SWTResourceManager.getColor(0,   0,
128));
                    }
                    dialogShell.open();
                    Display display = dialogShell.getDisplay();
                    while (!dialogShell.isDisposed()) {
                          if (!display.readAndDispatch())
                                display.sleep();
                    }
              } catch (Exception e) {
                    e.printStackTrace();
              }
      }

      private void btnCloseWidgetSelected(SelectionEvent evt) {
            dialogShell.getShell().close();
      }

}
```

Paquet Utils

// Constants

```
public interface Constants
{

      /* Constants de comunicació */
      final int           BACKLOG                   = 10;
      final int           MAX_MISSATGE = 10000;
      final int           SLEEP_TIME          = 50;

      /* Constants de SocketMissatge */
      final int CLIENT_LOGIN    = 1;
      final int CLIENT_LOGOUT   = 2;
      final int CHAT                          = 3;
      final int SERVER_DOWN           = 4;
      final int USUARIS_LIST          = 5;
      final int CANVI_ESTAT           = 6;
      final int USER_OK               = 7;
      final int USER_KO               = 8;
      final int SYNCRO                = 9;
      final int LIST_OK               = 10;
      final int MSG_OK                = 11;

      //Estat usuari
      final int OFFLINE         = 1;
      final int ONLINE          = 2;
      final int AWAY                  = 3;
```

```
	final int OCUPAT		= 4;

	// Constants de Alerta
	final int ERROR				= 1;
	final int INFO				= 2;

}
```

// MissatgerUtils

```
import java.io.IOException;

public class MissatgerUtils
{
	public static byte[] objectToBytes (Object object)
	{
		java.io.ObjectOutputStream out=null;
		java.io.ByteArrayOutputStream bs;

		bs = new java.io.ByteArrayOutputStream ();
		try {
			out = new java.io.ObjectOutputStream (bs);
			out.writeObject(object);
			out.close ();
		} catch (IOException e) {
			e.printStackTrace();
		}
		return bs.toByteArray ();
	}

	public static Object bytesToObject (byte bytes[])
	{
		Object res=null;
		java.io.ObjectInputStream in=null;
		java.io.ByteArrayInputStream bs=null;

		bs = new java.io.ByteArrayInputStream (bytes);
		try {
			in = new java.io.ObjectInputStream(bs);
			try {
				res = in.readObject();
			} catch (ClassNotFoundException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}

			in.close ();
			bs.close ();

		} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}

		return res;
	}
}
```

// Timer

```
public class Timer {
```

```
        private long temps;
        private boolean estat;
        private Thread myTime;

        public Timer(long time){
                temps=time;
                setEstat(false);
                myTime = new Thread(){
                        public void run(){
                                try {
                                        Thread.sleep(getTemps());
                                } catch (InterruptedException e) {
                                        e.printStackTrace();
                                }
                                setEstat(true);
                        }
                };
                myTime.start();
        }

        public boolean isEstat() {
                return estat;
        }
        private void setEstat(boolean est){
                estat=est;
                if (estat==true) myTime.stop();
        }
        public long getTemps() {
                return temps;
        }
}
```