Esquema criptogràfic
Joc electrònic remot segur
Ruleta

Eulàlia Huguet Puig
Enginyeria en Informàtica

Jordi Castellà Roca

Gener 2006

Agraïments

Aquest projecte és el fruit de 4 anys de compartir la feina i els estudis, amb el que comporta d'il·lusions i preocupacions. Seria una mostra d'ingratitud oblidar-me de les persones i de les coses que, d'alguna manera, hi han estat implicades.

Vull donar les gràcies al consultor d'aquest projecte, el Sr. Jordi Castellà, per la seva paciència, per contestar totes les meves preguntes, a vegades sense gaire sentit, i sobretot per donar-me aquest marge de llibertat que es necessita per compaginar entregues, feina i família.

En el terreny personal dono les gràcies als meus pares, perquè no han dubtat mai que algun dia acabaria el 2n cicle i per tots els bons consells que m'han donat i que m'han permès arribar a presentar aquest projecte. Especialment vull donar gràcies a la meva mare que s'ha encarregat de revisar lingüísticament tot el treball, i ja és el segon. Al Pep, el meu marit, que també és el segon projecte que suporta i l'ha sabut portar més bé que el primer. Ha tingut molta paciència escoltant hores de conversa avorrida i quedant-se al meu costat en els moments tristos. També al Marc, el meu germà, i la Núria, la seva dona, i a tots els meus amics i amigues que m'han ajudat i fet costat durant aquests darrers quatre anys.

Resum

El PFC s'emmarca dins de l'àrea de seguretat informàtica. Aquesta àrea abasta tant la seguretat dels elements físics: la xarxa i els servidors, com la seguretat dels elements lògics: les dades.

Cada cop és més freqüent traslladar a l'entorn virtual les coses quotidianes de la nostra vida. A la xarxa podem trobar serveis de tota mena, entre ells els que ens aporten diversió, entreteniment i allicient: els jocs. En concret, els jocs d'un casino on podem jugar amb diners són exemples de productes adaptats i implementats en una versió de joc electrònic remot. El problema de traslladar un joc del món real a un entorn virtual planteja un seguit d'interrogants, tant per a la part que posa el joc a la xarxa com per a la part que hi juga.

En el joc de la ruleta, en un casino, hem de confiar que el crupier no aturarà la bola en algun lloc concret del cilindre de números en què ningú hagi apostat. Hem de confiar que els cilindres que permeten rodar a la ruleta estan perfectament equilibrats per no afavorir més uns números que altres. Hem de confiar que els altres jugadors de la taula no faran més apostes quan ja se sap el resultat final, que cap altre jugador s'apropiarà d'una fitxa que no es seva. Tal com fem en el món real, hem de garantir que el joc virtual sigui just i honest, tant pels jugadors com pel propi casino.

El sistema presentat proposa una solució de joc electrònic remot segur per a la ruleta, que usa criptografia de clau pública, certificats i signatures digitals. Es defineix com es faran les accions, els protocols, per assegurar als participants que el joc és just i honest.

Índex

Índex de taules

Índex de figures

Capítol 1

Introducció

1.1 Justificació del PFC i context en el qual es desenvolupa

Les xarxes de comunicació ens permeten realitzar algunes activitats amb independència del moment i del lloc on ens trobem. En són exemples ben clars, cursar estudis (la UOC), fer una operació bancària i presentar la declaració de la renda. Cada cop és més comú veure aquest tipus d'interaccions entre entitats i usuaris. El problema que totes tenen és la seguretat, ja sigui per la identitat de la persona com per les dades que s'intercanvien.

Quan hom presenta la declaració de la renda, l'Agència Tributària està segura que l'usuari que la presenta és realment qui diu que és, que les dades que hi envien els usuaris no les podran rebutjar més tard. Aquestes propietats de seguretat s'aconsegueixen mitjançant criptografia de clau pública, amb l'ajuda del certificat i la signatura digitals.

Aquesta proliferació d'activitats a la xarxa ha arribat al sector de l'oci. Ens agrada jugar. Als humans, per naturalesa, els agrada jugar i si hi ha diners l'emoció encara és millor. El 1996 van aparèixer els primers casinos en línia i des de llavors hi ha hagut un boom de creació de pàgines web, que tenen jocs de tota mena als quals pots jugar amb diners vertaders o virtuals. Els jugadors, però, no s'acaben de fiar de l'honestitat d'un casino virtual.

Hem visitat els casinos virtuals de més renom, posant especial atenció als apartats de privadesa i seguretat. Aquestes parts impliquen: la primera, l'usuari s'ha de registrar i donar les dades i, la segona, obtenir el resultat i de quina manera es fa la comunicació entre els jugadors i el casino.

Dels casinos consultats, vegeu la taula 1.1, tots tenen una aplicació client que s'ha d'instal·lar i alguns ja estan adaptant la versió en línia. L'autenticació s'ha de fer amb nom d'usuari i contrasenya. La gran majoria asseguren que les comunicacions entre l'aplicació client i els servidors es fa de manera segura, encara que no diuen com, altres usen SSL, altres RSA de 128-bits. Els casinos virtuals que portin el logotip de PlayTech (http://www.playtech.com), significa que transfereixen totes les dades codificades amb complexos algorismes matemàtics i que les transaccions monetàries estan protegides per mecanismes RSA d'encriptació de clau pública i privada. Tots tenen els servidors fortament protegits amb tecnologies de tallafocs.

El registre dels usuaris és un formulari en què s'ha d'introduir dades, com el nom, el cognom, l'adreça de correu electrònic, la ciutat, l'estat, el pais, el codi postal i la data de naixement, sense comprovar si aquestes dades són certes o falses. En el moment de registrar la targeta de crèdit, habitualment es comprova que el nom del titular de la targeta de crèdit sigui el mateix que hem introduït abans en les dades personals.

Quant a l'obtenció del resultat, tots opten per un generador de nombres aleatoris, RNG, [1] per poder simular les experiències aleatòries dels jocs. Aquests RNG són similars als que podem trobar en una màquina escurabutxaques normal. El Casino Tropez ha deixat examinar a l'OPA (On-line Players Association) el seu generador de nombres aleatoris perquè el verifiqués. L'OPA ha arribat a la conclusió que és vertader i just a l'hora de donar premis. Aquest és un fet que tots s'afanyen a demostrar: el percentatge de pagaments sobre les apostes; si el casino té un percentatge del 97,82 (Casino Tropez - juliol 31), li queda un marge del 2,18.

El casinos virtuals usen en els seus sistemes un nivell acceptable de seguretat, tant en les transferències de dades com en les tranferències de diners. De tota manera, ens demanen que confiem en punts com la gestió de la partida, la gestió dels diners i l'obtenció dels resultats del joc: el número que sortirà a la ruleta, barrejar les cartes d'una partida de pòquer o els números que aniran sortint al bingo.

[1] De les sigles en anglès: Random Number Generator

Golden Palace	http://www.goldenpalace.com [PlayTech]
Casino On Net	http://www.888.com
Casino Tropez	http://www.casinotropez.com [PlayTech]
Inter España	http://espana.intercasino.com
Golden Riviera	http://www.goldenrivieracasino.com
Amber Coast Casino	http://www.ambercoastcasino.com (20-12-1005) [PlayTech]

Taula 1.1: Casinos consultats

1.2 Objectius del PFC

L'objectiu d'aquest PFC és dissenyar i implementar un sistema de joc electrònic remot segur, que ofereixi als jugadors un nivell de seguretat similar al que poden tenir quan juguen en un casino tradicional.

Perquè sigui segur usarem criptografia de clau pública, juntament amb el certificat i la signatura digitals. També implementarem uns esquemes que ens permetran desenvolupar les accions més comunes del joc de manera segura: apostar, incrementar el dipòsit, jugar, obtenir el resultat, etc.

Perquè es pugui jugar remotament, farem servir el protocol de comunicació RMI i usarem el format XML per intercanviar dades entre el jugador i el gestor.

Enregistrar les dades del joc en una base de dades, per consultar i auditar les partides que ja han acabat.

Dissenyar una interfície per al jugador perquè la interacció amb l'aplicació sigui més senzilla i còmoda.

1.3 Enfocament del PFC i mètode seguit

El projecte s'ha dividit en fases per tal de fer una implementació incremental; és a dir, cada fase s'implementarà amb el codi corresponent, es faran proves de test unitari i s'integrarà amb la fase anterior. D'aquesta manera es van ampliant les funcionalitats de l'aplicació a poc a poc i es pot donar per acabat el projecte en una determinada fase sabent que tot funcionarà completament.

El projecte consta de 7 fases:

1. Instal·lació del programari bàsic i creació d'una PKI.
2. Esquema criptogràfic.
3. Representació de les dades.
4. Comunicació de components.
5. Gestió de la informació.
6. Interfícies: client i servidor.
7. Documentació.

1.4 Planificació del projecte

El projecte està ubicat al primer quadrimestre del curs 2005-2006. Comença el 14 de setembre del 2005 i acaba el 2 de gener del 2006. En la figura 1.1 es pot veure quan han acabat i començat les fases i les tasques en què han estat dividides.

1.4.1 Instal·lació programari. Creació d'una PKI

Instal·lació del programari bàsic per començar a treballar: el programari de desenvolupament Java Development Kit (JDK), en la seva versió 1.5.0 i la llibreria criptogràfica desenvolupada per l'Institute for Applied Information Processing and communications (IAIK). Creació una infraestructura de clau pública (PKI) per a poder emetre certificats.

1.4.2 Esquema criptogràfic

L'objectiu d'aquesta tasca és estudiar els protocols criptogràfics i com implementar-los. Les tasques de què consta són les següents:

- Estudi del funcionament dels protocols criptogràfics.
- Disseny del diagrama de classes.
- Disseny UML: casos d'ús, diagrama d'estats i diagrames de seqüència.
- Implementació dels protocols criptogràfics.
- Tests i documentació d'aquesta fase.

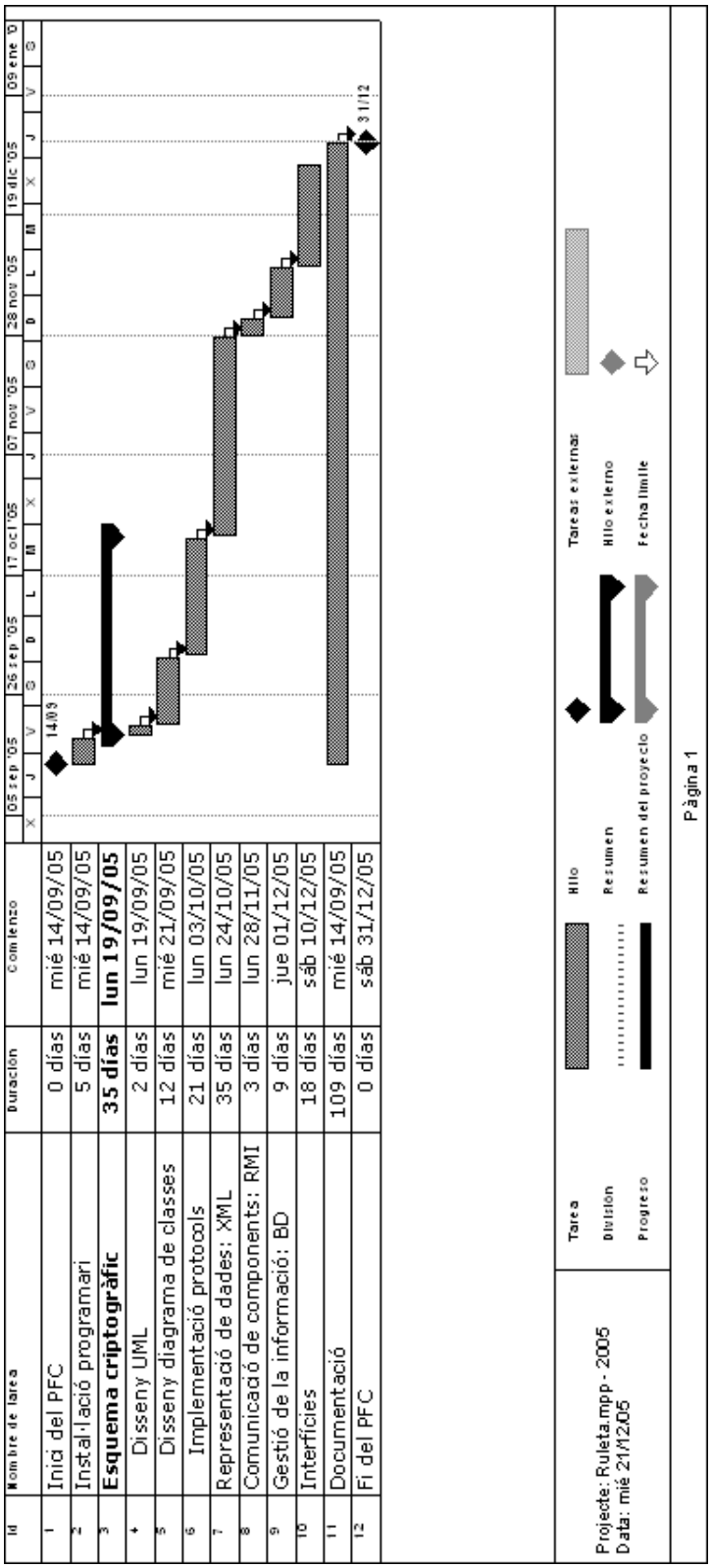

Figura 1.1: Planificació del projecte

Disseny del diagrama de classes
De l'estudi del funcionamet de l'aplicació i els protocols hem creat el diagrama inicial de classes. Aquest diagrama s'ampliarà en les següents fases ja que cadascuna afegeix noves funcionalitats al projecte.

Disseny UML
Seguint amb el procés de disseny del programari, el següent pas és fer el disseny UML de l'aplicació amb els casos d'ús, els diagrames d'estat i els diagrames de seqüència.

1.4.3 Representació de dades: XML

Els protocols criptogràfics, perquè funcionin, necessiten enviar i rebre informació. Hem definit l'estructura de dades amb format XML per a cadascun d'ells. Això ha comportat que s'afegeixin més classes al diagrama de classes, que també haurem d'implementar, provar i documentar. Les tasques d'aquesta fase són les següents:

- Disseny de l'estructura de dades XML per a cada protocol.
- Revisió del diagrama de classes.
- Implementació de les classes que gestionen les dades XML.
- Tests i integració amb la fase anterior.
- Documentació d'aquesta fase.

1.4.4 Comunicació de components: RMI

L'arquitectura de l'aplicació és client-servidor. Com a sistema de comunicació entre els clients i el servidor hem escollit Remote Method Interface (RMI). El diagrama de classes ha tornat a créixer. També haurem d'implementar, provar i documentar les noves classes. Les tasques fetes durant la comunicació de components és:

- Estudi dels diagrames de seqüència.
- Revisió del diagrama de classes.
- Implementació de les noves classes: Servidor, InterficieRemota.
- Tests i integració amb la fase anterior.
- Documentació d'aquesta fase.

1.4.5 Gestió de la informació : base de dades

Ens recolzarem en una base de dades per obtenir i recuperar informació en qualsevol moment de la partida. La base de dades també ens servirà per poder verificar què ha passat en un moment determinat. Les seves tasques són:

- Disseny del model Entitat-Relació.
- Instal·lació del programari necessari: MySQL
- Implementació del disseny amb SQL. Tests MySQL.
- Revisió diagrama de classes.
- Implementació de les noves classes: BDManager.
- Tests i integració amb la fase anterior.
- Documentació d'aquesta fase.

1.4.6 Interfície

La interfície està pensada per fer més agradable a l'usuari la interacció amb l'aplicació. La fase passa per les següents tasques:

- Disseny de la interfície.
- Implementació de les classes que gestionen la interfície.
- Tests i integració amb la fase anterior.
- Documentació d'aquesta fase.

1.4.7 Documentació

La documentació s'ha anat fent al llarg de tot el projecte. Cada fase (llevat de la documentació) es correspon aproximadament amb un capítol de la memòria. D'aquesta manera l'última setmana del projecte ha de servir per escriure les conclusions, revisar les parts escrites anteriorment i encaixar tot el document per entregar-lo.

1.5 Representació de l'escenari del Joc

L'arquitectura de l'aplicació és client - servidor, on els clients són els jugadors i els servidors són els gestors. Els jugadors fan peticions com, per exemple, iniciar una partida, fer una aposta, incrementar el crèdit, etc. al gestor, el qual accepta les peticions i enregistra, a la base de dades, la informació corresponent. A la figura 1.2 es representa gràficament l'escenari on funciona tot el sistema de joc electrònic segur de la ruleta.

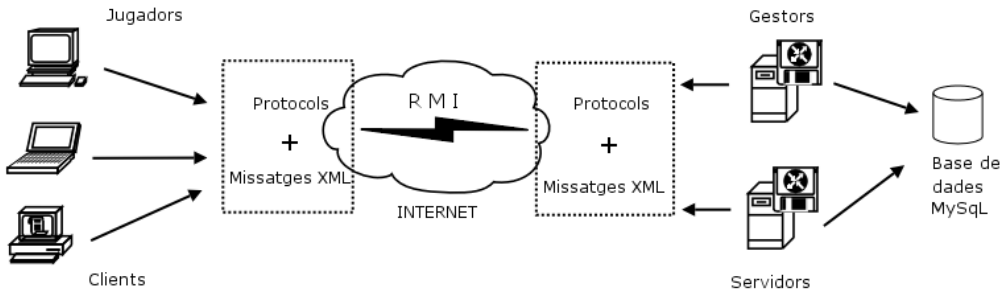

Figura 1.2: Representació de l'escenari

Un sistema com aquest es considera multijugador i funciona a través de la xarxa. Per tal de simplificar el sistema de joc, reduirem l'escenari a un sol ordinador, que executarà un cop la part servidora (1 gestor) i una o més vegades la part client (n jugadors). Els jugadors podran jugar tots a la vegada, però no en una mateixa ruleta, cadascú jugarà en una ruleta i el gestor atendrà totes les peticions dels jugadors. El gestor no permetrà que un jugador jugui dues partides a la vegada. Tota la comunicació que es fa entre els jugadors i el gestor segueix els protocols definits al capítol 3, i les dades estan estructurades en el format XML dissenyat al capítol 5. El mètode usat per executar les crides remotes és el RMI descrit al capítol 6. A la figura 1.3 podem veure la representació gràfica de l'escenari simplificat.

Figura 1.3: Representació de l'escenari simplificat

1.6 Productes obtinguts

Un cop desenvolupat, el sistema consta de:

Una insfraestructura de clau pública que permet signar peticions de certificat. Si és necessari que més de dos jugadors juguin a la ruleta a la vegada, podem generar més parells de claus i peticions de certificat i signar-les amb la PKI.

Dues aplicacions . Una per al jugador i l'altra per al gestor, i una estructura de directoris que conté totes les classes necessàries tant per al client com per al gestor, per poder jugar a la ruleta remotament de manera segura. L'aplicació client permet registrar-se, autenticar-se, iniciar una partida, apostar, etc. L'aplicació del gestor, en canvi, registra, autentica, permet iniciar una partida, recull les apostes dels jugadors, gestiona els dipòsits dels jugadors, etc.

Una base de dades que enregistra les accions que es van produint i de la qual podem obtenir tota mena de llistats d'ús de l'aplicació: els números premiats, els guanys o les pèrdues dels jugadors, les partides creades, a quins números s'aposta més, etc.

1.7 Descripció breu dels capítols de la memòria

Infraestructura de clau pública

En aquest capítol s'introdueixen els conceptes de signatura digital, de la criptografia de clau pública i de la infraestructura que es necessita per gestionar la criptografia de clau pública. S'han triat explicacions, definicions i esquemes dels mòduls didàctics 5 (Xifres de clau pública), 6 (Signatura digital) i 7 (Infraestructura de clau pública) de l'assignatura de Criptografia.

Esquema criptogràfic

Els esquemes criptogràfics vénen determinats des d'un principi per la documentació del PFC. Es descriu cada esquema i de quina manera es faran servir en l'aplicació.

Disseny de l'aplicació

En aquest capítol es repassa breument la història de la ruleta. Se n'expliquen les normes i la mecànica. A partir d'aquestes explicacions i del dos capítols anteriors es fa un primer disseny de l'aplicació.

Representació de dades: XML

S'explica el format que hauran de tenir les dades que s'intercanviaran entre el jugador i el gestor.

Comunicació de components: RMI

S'introdueixen els conceptes de comunicació RMI i com s'han aplicat al disseny del joc electrònic.

Gestió de la informació: base de dades

Es dóna el model ER que definirà la base de dades. S'expliquen les decisions de disseny i com s'ha implementat la base de dades en MySqL i amb Java.

Interfície

Una part important en totes les aplicacions és la presentació de l'aplicació i la manera que els usuaris interactuen amb ella. S'explica com s'ha dissenyat la interfície del jugador i com funciona.

Conclusions

Fem balanç dels objectius aconseguits durant el projecte.

Eines utilitzades

En cada apartat de l'apèndix s'explica una eina, com s'instal·la, quina versió hem fet servir i com l'hem usada dins del PFC.

Fem una partida?

Posada en marxa del sistema i exemple d'execució d'una partida.

Capítol 2

Infraestructura de clau pública

2.1 Introducció

La criptografia de clau pública es basa en el fet de tenir un parell de claus, una de privada que només coneix la persona i una de pública a la qual tothom ha de tenir accés.

Les signatures digitals mantenen una estreta relació amb la criptografia de clau pública. Per signar un missatge, l'usuari fa servir la seva clau privada; per verificar una signatura, qualsevol pot fer servir la clau pública de la persona que signa. El verificador queda convençut que el missatge no ha estat alterat perquè està signat. A més, amb aquest procediment, el signatari no pot repudiar més tard el fet d'haver signat el missatge, perquè ningú llevat del signatari no té la clau privada necessària per a produir la signatura.

El problema radica en el fet que la clau pública hagi d'estar disponible per a tothom; la manera com hem de distribuir-la i, quan l'obtenim, estar segurs que pertany a qui ens pensem.

Diffie i Hellman (1976) van introduir la idea de directori segur en línia, en el qual s'establia un lligam únic entre el nom d'usuari i la seva clau pública. Aquesta idea de directori deixa de ser eficient quan tractem amb quantitats mitjanes i grans de parelles nom d'usuari, clau pública. L. Kohnfelder (1978), basant-se en aquesta idea d'autoritat central de confiança, va proposar de crear uns registres de dades signades -els certificats- que permetrien que la distribució de claus es fes des de directoris públics que no requerissin confiança.

Definició 2.1 Un *certificat digital* és una estructura de dades que conté

informació del propietari de les claus criptogràfiques, la clau pública en si i de la signtatura digital dels dos camps anteriors que hi dóna validesa.

Des d'aleshores fins avui, els certificats han esdevingut el principal mitjà de distribució de les claus públiques dels sistemes de clau asimètrica. De tota manera l'ús de certificats no resol el problema de la distribució de claus. Qui ha de signar el certificat? Quins mecanismes són necessaris perquè dos usuaris que no es coneixen puguin assegurar la seva identitat en una comunicació virtual?

L'objectiu d'una infraestructura de clau pública és la gestió eficient i confiable de les claus criptogràfiques i els certificats perquè es puguin utilitzar per a funcions d'autenticació, integritat, no-repudi i confidencialitat. La infraestrutura de clau pública crea un marc segur d'intercanvi de dades en un entorn típicament insegur com Internet.

Definició 2.2 Una *infraestructura de clau pública* [1] (PKI) és el conjunt de maquinari, programari, persones, polítiques necessaris per a crear i gestionar certificats digitals basats en la criptografia de clau pública.

El component essencial de la insfraestructura de clau pública és el certificat, al voltant del qual es crea aquesta infraestructura de suport que ens permetrà registrar usuaris, emetre certificats, signar-los, revocar-los, etc. Per a poder tenir aplicacions interoperables, cal establir uns estàndards en matèria d'infraestructura de clau pública. El fet que diversos fabricants es posin d'acord en la sintaxi i l'estructura de les dades fa que el mercat creixi i millori.

Avui per avui, aquest mercat es troba entre solucions propietàries i estàndards oberts. Alguns fabricants no volen donar a conèixer les seves solucions perquè basen la seguretat en l'obscuritat i en el secret, la qual cosa no és una bona política, ja que no compleix la suposició de Kerckhoff [2]

Els *RSA Laboratories* han fet importants aportacions en els formats de dades d'intercanvi per a la infraestructura de clau pública i han aconseguit que moltes de les seves especificacions s'hagin convertit en estàndards *de facto*, com els PCKS, descrits a l'apartat 2.6.

[1] En anglès, Public key Infrastruture

[2] La suposició de Kerckhoff diu que tot el mecanisme de xifratge, excepte el valor de la clau secreta, és conegut pel criptoanalista enemic.

2.2 Components d'una infraestructura de clau pública

Com ja em vist, l'estructura clau pública és basa en la gestió de certificats digitals. Els components essencials que ens permetran aquesta gestió són l'autoritat de certificació, els subscriptors i els repositoris. N'hi ha d'altres que no són essencials, però faciliten aquesta tasca, com són les autoritats de registre, l'autoritat de validació, l'autoritat de segellat de temps.

Definició 2.3 L'*Autoritat de certificació*, CA, [3] és la responsable d'emetre i revocar certificats. És l'entitat de confiança que dóna legitimitat a la relació d'una clau pública amb la identitat d'un usuari o servei.

En una infraestructura de clau pública pot haver-hi una o més autoritats de certificació. Per crear una autoritat de certificació primer haurem de crear un parell de claus, una de pública i una de privada, que s'usaran per signar i validar els certificats. És recomanable que les claus de la CA siguin fortes perquè la probabilitat que un atacant les trenqui sigui mínima. La fortalesa de la clau dependrà de la seva longitud i de la qualitat de l'algorisme que la genera.

Definició 2.4 L'*Autoritat de registre*, RA,[4] és l'encarregada de verificar el lligam entre les claus públiques i la identitat dels seus titulars.

Les RA són components opcionals de les infraestructures de clau pública i s'usen per descarregar l'autoritat de certificació de moltes funcions administratives. Per exemple, a l'Estat espanyol les RA de la Fàbrica Nacional de Moneda i Timbre són les delegacions que l'Agència Tributària té en cada localitat. Les RA són especialment útils en organitzacions grans i geogràficament disperses.

Definició 2.5 Els *subscriptors* i les *entitats finals* són aquells que poseeixen un parell de claus i un certificat associat a la clau pública. Amb aquest parell de claus es podran fer signatures digitals, xifrar i desxifrar documents, missatges, etc. Una entitat final és una empresa o un organisme, mentre que un subscriptor és una persona.

Definició 2.6 Els *usuaris* són els agents que validen signatures digitals i la seva ruta de certificació, a partir de les claus públiques emeses per autoritats de certificació de confiança. També poden xifrar documents per a subscriptors i entitats finals.

[3] En anglès Certification Authority
[4] En anglès, Registration Authority

Els agents no necessiten cap parell de claus per desenvolupar les seves tasques. Els subscriptors i entitats finals són, en particular, usuaris.

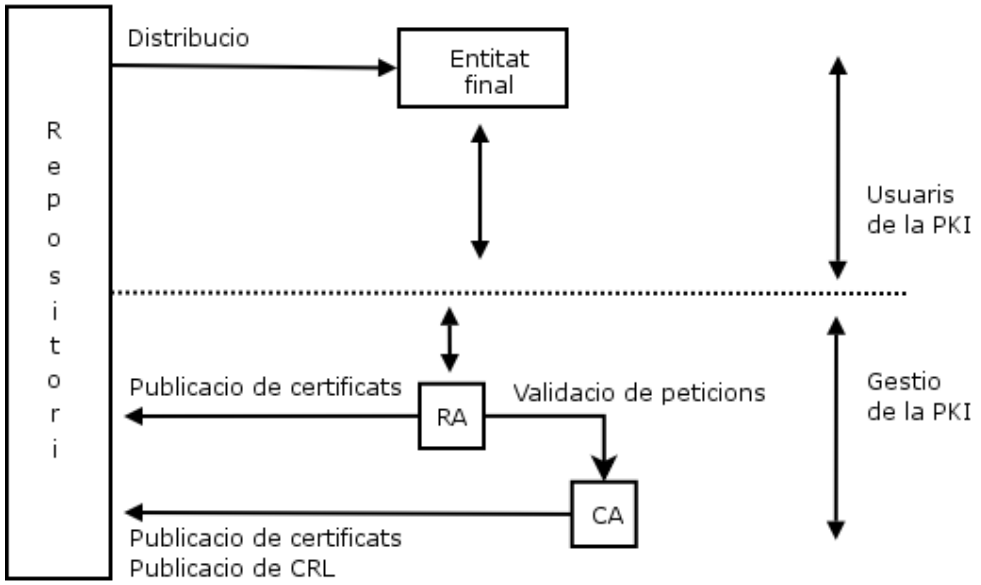

Figura 2.1: Components d'una PKI

Definició 2.7 Els *repositoris* són estructures encarregades d'emmagatzemar la informació relativa a la infraestructura de clau pública.

Els dos repositoris més importants en una infraestrutura de clau pública són el repositori de certificats i el repositori de llistes de revocació de certificats. La llista de revocació de certificats[5] (CRL) inclou tots aquells certificats que per diversos motius ja no són vàlids abans de la data de caducitat establerta en el mateix certificat.

El tipus de repositoris més usats són els directoris. Un directori és una base de dades especialitzada en la qual s'emmagatzema informació tipificada i organitzada sobre objectes. Un directori està optimitzat per fer operacions de lectura, les navegacions i les grans cerques, i el seu objectiu és donar respostes ràpides a un gran volum de peticions.

[5]En anglès, Cetificate Revocation List

Definició 2.8 *L'autoritat de validació* [6] (VA) és l'encarregada de comprovar la validesa dels certificats digitals.

Aquesta autoritat pot ser la pròpia autoritat de certificació o una autoritat externa.

Definició 2.9 L'*autoritat de segellat de temps* [7] és l'encarregada de signar un missatge amb la finalitat de provar que el missatge existeix en un determinat instant de temps.

La necessitat d'una autoritat de segellat de temps és important per a la propietat de no-repudi. Els serveis de no-repudi han de poder establir l'existència d'unes dades abans de determinats moments.

2.3 Models de confiança

L'ús de la criptografia de clau pública implica l'ús de certificats que ens assegurin el lligam entre el certificat i algun atribut, normalment és la identitat de la persona propietària del certificat. Els mètodes de certificació absoluta no són possibles, ja que un certificat no es pot certificar ell mateix. Per aquesta raó s'han proposat diversos models de confiança: el model distribuït de la xarxa de confiança, el model pla i el model jeràrquic. Hi ha altres models: el model de navegació per llista de confiança, el model de certificats creuats i el model Bridge-CA que no són a l'abast del projecte.

2.3.1 Model distribuït

El sistema distribuït de la xarxa de confiança és el model més senzill d'utilitzar; és adequat per a grups d'usuaris petits que ja tenien tractes abans de la implantació de la infraestructura de clau pública.

En aquest model, cada usuari crea i signa certificats per als usuaris que coneix. No és necessari cap infraestructura central, ni tampoc una tercera entitat de confiança que doni fe de la identitat dels usuaris. Aquest model és el que usa el programari Pretty Good Privacity, PGP. [8]

[6] En anglès, Validation Authority
[7] En anglès, Time Stamp Authority
[8] En català, privadesa prou bona

2.3.2 Model pla

El model pla és el sistema més senzill d'infraestructura de clau pública que inclou una única autoritat de certificació com a tercera part de confiança encarregada de l'emissió i la gestió dels certificats dels subscriptors. Els usuaris poden validar la identitat dels subscriptors a partir del certitficat de l'autoritat de certificació. L'autoritat de certificació posseeix un certificat que ella mateixa ha generat i en el qual els usuaris dipositen la seva confiança.

En un *certificat autosignat*, la clau pública que se certifica correspon a la clau privada que s'utilitza per signar el certificat. El nom de l'emissor i el nom titular del certificat és el mateix.

El model pla s'acostuma a fer servir en l'àmbit de les intranets. Per a entorns més extensos i oberts, aquest model s'amplia.

2.3.3 Model jeràrquic

És l'ampliació del model pla i és la implementació més típica d'una infraestructura de clau pública. En el *model jeràrquic*, els certificats dels subscriptors i les entitats finals estan signats per una entitat externa que també s'identifica amb certificats que emetrà una autoritat de certificació de jerarquia superior.

Els certificats de l'autoritat de certificació de jerarquia superior poden estar, a la vegada, certificats per altres autoritats de certificació, i així successivament, fins a arribar a una autoritat de certificació que té un certificat autosignat. Aquesta autoritat de certificació s'anomena *autoritat de certificació arrel*[9]. Si els usuaris de la infraestructura de clau pública confien en l'autoritat de certificació arrel, també confiaran en les altres autoritats de certificació de la resta de la jerarquia.

2.4 Cicle de vida de claus i certificats digitals en la PKI

2.4.1 Generació de les claus

La primera part del cicle de vida d'un parell de claus i del seu certificat digital és la generació de les claus. Les claus poden generar-les els mateixos

[9]En anglès, root CA

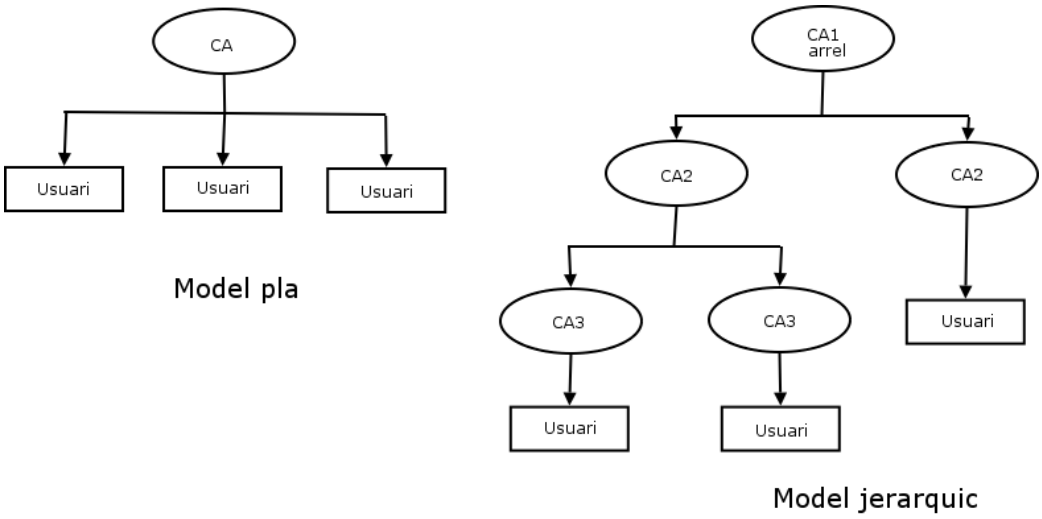

Figura 2.2: Models de confiança

subscriptors o les entitats externes. Les claus criptogràfiques s'han de generar segons les necessitats de les aplicacions i els serveis que les usaran. Pel tal de diferenciar quines claus són apropiades per certes aplicacions, els certificats contenen un camp amb el tipus de certificat que representen. Bàsicament, n'hi ha de dos tipus:

(a) En els *certificats de xifratge* pot ser necessari tenir una còpia de seguretat de la clau privada per tal de poder recuperar la informació en clar, si la clau de desxifratge es perd o es destrueix. Aquesta pràctica és comuna en l'entorn empresarial.

(b) Els *certificats de signatura* es fan servir per donar serveis d'autenticació i de no-repudi, i és important que no hi hagi cap còpia de la clau privada corresponent. Qualsevol sistema de recuperació de claus per a aquest tipus de certificats n'invalidaria les propietats.

2.4.2 Registre

El registre és el procés pel qual un subscriptor o una entitat final es dóna a conèixer per primera vegada a una autoritat de certificació (directament o des d'una autoritat de registre), per tal de demanar-li un certificat.

El subscriptor ha de proporcionar informació sobre la seva identitat, un identificador únic, alguna prova d'identitat i altra informació addicional. Si ell

mateix ha generat el parell de claus criptogràfiques, també ha de donar la clau pública que vol certificar. Tota aquesta informació s'estructura en una petició de certificat en format PKCS#10. Vegeu l'apartat 2.6.

2.4.3 Certificació

La certificació és el procés pel qual l'autoritat de certificació emet un certificat; és a dir, signa digitalment la clau pública i les dades d'un subjecte i, per tant, dóna fe d'aquesta correspondència. En aquest procés es defineix també el període de validesa del certificat i l'ús que se'n podrà fer.

2.4.4 Recuperació de claus

L'autoritat de certificació pot oferir sistemes de recuperació de claus mitjançant un sistema de còpies de seguretat al qual només puguin accedir els usuaris autoritzats per obtenir les dades. El sistema ha de garantir que les claus no es poden comprometre. Per exemple, una empresa pot necessitar accedir a documents prèviament xifrats per un treballador que s'hagi donat de baixa de l'empresa.

2.4.5 Revocació de certificats

La revocació de certificats és el procés pel qual una autoritat de certificació invalida un certificat abans del seu període d'expiració. Hi ha diverses raons per les quals un certificat pot ser revocat:

- Compromís de la clau privada associada al certificat.
- Canvi de subscriptor per un altre proveïdor de serveis d'infraestructura de clau pública.
- Canvi en la informació que porta el certificat (nom del subscriptor, companyia a què pertany, etc.).
- Substitució prematura del parell de claus de l'autoritat de certificació.

2.4.6 Renovació de certificats

La renovació d'un certificat digital pot venir donada bàsicament per dos motius:

- Pèrdua de validesa del certificat (caducitat del certificat, revocació per canvi d'informació del certificat, etc.).

- Pèrdua de validesa de les claus que certifica (revocació del certificat per compromís de la clau privada associada).

El certificat digital caduca quan expira el seu període de validesa.

2.5 Certificats digitals: el certificat X.509

El format de certificats més àmpliament acceptat en infraestructura de clau pública es coneix com X.509v3. Els certificats estan estructurats en camps, que poden ser de tres tipus:

a) *Bàsics*: camps que aporten informació sobre l'autoritat de certificació que ha emès el certificat, l'entitat/el subscriptor a què pertany la clau pública, la pròpia clau pública, el període de validesa i la identificació del certificat.

b) *Necessaris per a la signatura*: camps que utilitzarà qui rebi el certificat per comprovar que el document està signat correctament.

c) *Ampliacions*: camps que han aparegut per cobrir les noves necessitats d'atributs en un certificat.

Podeu trobar la descripció de tots els camps a la norma RFC2459. [2]

2.6 Els documents PKCS

RSA Laboratories, amb l'objectiu d'accelerar la implantació de la criptografia de clau pública, ha desenvolupat els documents PCKS [10] per establir unes normes sobre els formats de dades que s'han d'usar. En la redacció d'aquests documents hi han donat suport i collaborat altres empreses i fabricants, com Apple, Microsoft, DEC, Lotus, Sun, MIT, etc.

Des de la seva primera publicació l'any 1991 fins ara, han estat usats i implementats arreu del món. Actualment hi ha 10 documents. El PKCS#2 i #4 han estat incorporats al PKCS #1 i el PCKS#6 s'ha retirat en favor de la versió de l'estàndard X.509. El PKCS#13 i #14 encara no s'han publicat. A la taula 2.1 hi podeu trobar relacionades totes les normes PKCS i la seva descripció.

[10] En anglès, Public-Key Cryptography Standards

2.7 Ús d'una PKI en el projecte

L'intercanvi de dades entre l'aplicació client i l'aplicació servidor s'ha de fer de manera segura. La criptografia de clau pública ens permatrà signar i xifrar missatges a partir d'un parell de claus. Com hem vist en els apartats anteriors, és necessària una infraestructura (PKI) que ens permeti obtenir un certificat que demostri que la clau pública pertany a un determinat usuari.

Hem triat un model de confiança pla, tindrem, doncs, una única autoritat de certifiació (CA), que ens permetrà acceptar peticions de certificat i retornarles directament a l'usuari. Tot i que una PKI pot desenvolupar moltes altres tasques, només la usarem per generar els certificats. Cada usuari de l'aplicació client ha de tenir un parell de claus i un certificat signat per la CA de confiança, i tot emmagatzemat en una estructura d'intercanvi d'informació personal de tipus PKCS#12. L'aplicació servidor també ha de tenir els mateixos elements: un parell de claus, un certificat signat per la CA de confiança i un magatzem de tipus PKCS#12.

Construïm una petita PKI amb openssl. Obtenim un parell de claus per a la CA de 2048 bits de llargada. La clau de l'autoritat de certificació ha de ser suficientment forta perquè és una peça d'informació sensible. Si la trenquessin, tota la confiança que hi han dipositat els usuaris es veuria compromesa. Generem un certificat autosignat que serà el certificat de la CA.

Obtenim un parell de claus per a cada client de l'aplicació i també per a l'aplicació servidor. Totes amb una longitud de 1024 bits. No cal que siguin tan fortes com la clau de la CA. Generar la petició de certificat per a cadascú i enviar-la a la CA. La CA emetrà tots els certificats demanats. Un cop tenim el parell de claus i el certificat ja podem posar-los en un magatzem PKCS#12. Tot el procés està detallat a l'annex A, apartat A.1.

PKCS	Descripció
1	Defineix els mecanismes per xifrar i signar dades amb el criptosistema de clau pública RSA.
3	Defineix un protocol Diffie-Hellman per a l'intercanvi de claus.
5	Descriu un mètode per xifrar una cadena de text amb una clau secreta derivada d'una frase de pas.
7	Defineix una sintaxi general per als missatges que inclouen millores criptogràfiques, com signatures digitals o xifratge.
8	Descriu el format de la informació de la clau privada. Aquesta informació inclou una clau privada per a algun algorisme de clau pública i, opcionalment, un conjunt d'atributs.
9	Defineix els tipus d'atributs seleccionats per utilitzar en altres estàndards PKCS.
10	Descriu una sintaxi per a les peticions de certificació.
11	Defineix una interfície de programació independent de la tecnologia, anomenada Cryptoki, per a dispositius criptogràfics, com targetes intel·ligents i targetes PCMCIA.
12	Especifica un format portable per a emmagatzemar i transportar claus privades d'usuari, certificats, secrets diversos, etc.
13	Defineix els mecanismes per xifrar i signar dades utilitzant criptografia basada en corbes el·líptiques.
14	Cobreix la generació de nombres pseudoaleatoris.
15	Complement del PKCS#11 que defineix el format de les credencials criptogràfiques emmagatzemades en dispositius criptogràfics.

Taula 2.1: Documents PKCS

Capítol 3

Esquema criptogràfic

3.1 Definicions prèvies

Definim els termes esdeveniment, apostar i TTP:

Definició 3.1 Un *esdeveniment* és cadascun dels tipus de resultats que hom pot considerar una experiència aleatòria.

Definició 3.2 *Apostar* és pactar el guany o la pèrdua d'una quantitat de diners o d'una altra cosa segons quin sigui el resultat d'una altra juguesca.

Definició 3.3 *TTP* Són les sigles en anglès de Trusted Third Party, tercera part de confiança. En criptografia una tercera part de confiança és una entitat imparcial en la qual es confia per dur a terme una tasca de manera honesta.

3.2 Propietats de seguretat

A la ruleta, el fet de llançar la bola seria una experiència aleatòria. Els diversos tipus de resultats són els valors que té la ruleta. L'esdeveniment seria la casella on s'atura la bola. El resultat de la juguesca depèn d'un únic esdeveniment, per això també es diu que pertany als jocs d'una sola tirada.

3.2.1 Apostes

Quan en una partida el jugador fa una juguesca s'han de complir les propietats següents:

Autenticitat: per a qualsevol aposta, s'ha de poder demostrar quin jugador l'ha realitzada.

Integritat: una aposta no pot ser manipulada per cap part un cop ha estat realitzada.

No-repudi: si un jugador fa una aposta no pot rebutjar-la més tard. El gestor tampoc pot rebutjar-la si el jugador guanya.

Seqüència de joc: una aposta necessàriament ha de tenir aquesta informació: la partida en què ha estat realitzada, l'instant de la partida, la quantitat de diners i el concepte (en quin lloc apostem els diners).

3.2.2 Joc

L'operativa del joc de la ruleta és senzilla. El jugador fa una o més apostes. Les apostes han de complir les propietats de seguretat anteriors. A continuació s'obté un esdeveniment que determina el resultat del joc. Segons el resultat es paguen/cobren les apostes.

En un sistema de joc electrònic remot no es disposa d'una ruleta de veritat per reproduir les experiències aleatòries. Les parts que participen en el joc, el jugador i el gestor, no han de poder manipular la simulació d'experiències al seu favor. Cal garantir que la manera d'obtenir l'esdeveniment es faci tan honestament com sigui possible.

3.3 Esquema criptogràfic

Una primera aproximació per satisfer les propietats de seguretat del subapartat 3.2.2 és emprar una TTP per obtenir els esdeveniments del joc. Si la TTP participa en el joc està en situació de privilegi, perquè pot generar esdeveniments que la facin guanyar o perdre.

En aquest PFC els protocols criptogràfics no inclouen una TTP. El jugador i el gestor obtindran de manera conjunta els esdeveniments. Els protocols asseguren que els participants en el joc no poden manipular els esdeveniments a favor seu.

3.3.1 Notació

En la descripció dels protocols s'empra la notació següent:

$(P_{Entitat}, S_{Entitat})$: parell de claus asimètriques propietat d'Entitat, on P correspon a la clau pública i S a la clau privada.

$S_{Entitat}[M]$: signatura digital del missatge M amb la clau privada S d'Entitat.

$E_{Entitat}(M)$: xifratge del missatge M amb la clau asimètrica pública $P_{Entitat}$ d'Entitat.

$H(M)$: sortida d'una funció resum criptogràfica del missatge M. Aquestes funcions reben el nom de funcions hash.

3.3.2 Protocol de compromís

El protocol de compromís té dues fases: la fase de lliurament del compromís i la fase d'obertura del compromís. A l'hora d'obtenir un esdeveniment, el resultat del joc, hi intervenen dues parts: el jugador i el gestor. El jugador s'ha de comprometre davant del gestor amb un valor c. En la fase de lliurament del compromís, el jugador calcula un cert valor c^* a partir de c i l'envia al gestor. Aquesta transformació té les propietats següents:

- El gestor no pot saber res de c a partir de c^*.
- El jugador, un cop ha enviat c^*, no pot trobar un altre valor $c \neq c^{'}$ tal que es pugui obtenir c^* a partir de $c^{'}$. És a dir, el jugador no pot canviar el seu compromís.

En la fase d'obertura del compromís, el jugador lliura c al gestor i aquest verifica que s'obté c^* a partir de c. I, de la mateixa manera, el gestor també s'ha de comprometre amb el jugador amb un altre valor c.

En els escrits especialitzats en la matèria el trobarem amb el terme anglès de compromís de bit, *bit commitment*. Aquest protocol és de molta utilitat quan dues parts volen intercanviar-se certa informació de manera simultània.

En una partida P que té per identificador I_P, el jugador J i el gestor G es comprometen a un valor c_1 i c_2 respectivament. Suposem que tant el jugador com el gestor tenen un parell de claus.

Protocol 1 $[c_1, c_2, I_P]$

1. J calcula $c_1^* = H(c_1)$.
2. J signa c_1^* amb la seva clau privada, $S_J[I_P, c_1^*]$.
3. J envia $(c_1^*\ ,\ S_J[I_P, c_1^*])$ a G.

4. G calcula $c_2^* = H(c_2)$.
5. G signa c_2^* amb la seva clau privada, $S_G[I_P, c_2^*]$.
6. G envia (c_2^* , $S_G[I_P, c_2^*]$) a J.
7. J verifica la signatura $S_G[I_P, c_2^*]$ i la guarda juntament amb c_2^*.
8. G verifica la signatura $S_J[I_P, c_1^*]$ i la guarda juntament amb c_1^*.

El protocol d'obertura del compromís del jugador cap al gestor és com segueix:

Protocol 2 $[J, G, I_P]$

1. J signa c_1 amb la seva clau privada, $S_J[I_P, c_1]$.
2. J envia (c_1 , $S_J[I_P, c_1]$) a G.
3. G recupera c_1^* (del protocol anterior).
4. G verifica la signatura digital $S_J[I_P, c_1]$.
5. G verifica $c_1^* =^? H(c_1)$.

3.3.3 Inicialització

Cada jugador disposa d'un parell de claus (S_J, P_J). El gestor del joc també disposa d'un3 parell de claus (S_G, P_G), definim I_G, I_J com els identificadors del gestor i del jugador respectivament.

Un jugador J, per accedir al joc, ha d'estar registrat. En el procés de registre, hi ha les dades següents:

$Cert_J$: certificat digital del parell de claus del jugador J.

I_J: l'identificador del jugador serà el hash del certificat digital.

D_J: diners que té el jugador per fer les seves apostes. Aquest valor inicialment és zero.

I s'ha de fer d'aquesta manera:

1. El jugador J envia el seu certificat digital al gestor G i el valida.
2. El gestor crea I_J calculant el hash del certificat.

3. El gestor comprova que el jugador no s'hagi registrat amb un altre nom i el mateix certificat.

4. Emmagatzema a la base de dades el certificat, l'identificador i les dades personals del jugador.

5. El jugador rep el certificat del gestor i també el valida

3.3.4 Autenticació del jugador i del gestor del joc

Els usuaris per autenticar-se davant del gestor del joc empraran el protocol de Needham-Schroeder. El jugador ha d'obtenir la clau pública del gestor abans de començar el protocol.

Protocol 3

1. J realitza les operacions següents:
   (a) Obtenir un valor de forma aleatòria, N_i.
   (b) Xifrar N_i i I_J amb la clau pública de G, $E_G(N_i, I_J)$.
2. G realitza les operacions següents:
   (a) Desxifrar $E_G(N_i, I_J)$ amb la clau privada de G, S_G, i obtenir N_i i I_J.
   (b) Obtenir el certificat de J a partir de I_J. A partir del certificat obtindrà la clau pública del jugador, P_J.
   (c) Obtenir un valor de forma aleatòria, N_G.
   (d) Xifrar N_i, N_G i I_G amb la clau pública P_J de J, $E_J(N_i, N_G, I_G)$.
   (e) Enviar $E_J(N_i, N_G, I_G)$ a J.
3. J realitza les operacions següents:
   (a) Desxifrar $E_J(N_i, N_G, I_G)$ amb la clau privada S_J de J i obtenir N_i', N_G i I_G.
   (b) Si $N_i' =^? N_i$, J està autenticat davant de G.
   (c) Xifrar N_G amb la clau pública P_G de G, $E_G(N_G)$.
   (d) Enviar $E_G(N_G)$ a G.
4. G realitza les operacions següents:

(a) Desxifrar $E_G(N_G)$ amb la clau privada S_G de G, i obtenir N'_G.

(b) Si $N'_G =^? N_G$, G està autenticat davant de J.

3.3.5 Iniciar una partida

Un cop el jugador ha estat registrat i autenticat ja pot iniciar una partida per un joc, j.

Protocol 4 [j]

1. G calcula un identificador de partida I_P amb els passos següents:
   (a) Obtenir de forma aleatòria un valor r.
   (b) Obtenir l'instant de temps actual, T.
   (c) Obtenir el número de partides realitzades, N.
   (d) Calcular $I_P = \{j|r|T|N+1\}$.
   (e) Incrementar N en una unitat.
2. G signa I_P amb la clau privada S_G de G, $S_G[I_P]$.
3. G envia (I_P,$S_G[I_P]$) a J.
4. J verifica la signatura $S_G[I_P]$.
5. J verifica que les dades de la partida joc i T són correctes.

3.3.6 Incrementar el dipòsit

El jugador J ha d'ingressar diners al seu compte per tal de fer les apostes. L'increment del dipòsit es fa com segueix:

Protocol 5 [D_J]

1. J realitza les operacions següents:
   (a) Obtenir de forma aleatòria un valor r.
   (b) Obtenir l'instant de temps actual, T.
   (c) Obtenir el valor que es vol afegir al dipòsit, V.
   (d) Obtenir les dades de la targeta de crèdit, B.
   (e) Calcular l'identificador del dipòsit, $I_D = \{r|T|V|B\}$.

(f) Signar I_D amb la clau privada S_J, $S_J[I_D]$.

(g) Enviar (I_D,$S_J[I_D]$).

2. G realitza les operacions següents:

(a) Verificar la signatura $S_J[I_D]$.

(b) Verificar les dades del dipòsit: T, V, i B.

(c) Calcular el nou dipòsit D'_J del jugador J, $D'_J = D_J + V$.

(d) Calcular el rebut del crèdit disponible R_D, $R_D = S_G[I_J|D'_J]$.

(e) Enviar (D'_J,R_D) a J.

3. J realitza les operacions següents:

(a) Verificar la signatura digital de R_D.

(b) Verificar que el crèdit D'_J és correcte.

3.3.7 Fer una aposta

Un jugador J realitza una aposta en una partida I_P mitjançant el protocol següent:

Protocol 6 [I_P]

1. J realitza els passos següents:

(a) Obtenir l'identificador de la partida, I_P.

(b) Obtenir de manera aleatòria un valor r.

(c) Obtenir l'instant de temps actual, T.

(d) Obtenir la quantitat de diners de l'aposta, V.

(e) Obtenir el concepte de l'aposta, C.

(f) Calcular l'identificador de l'aposta $I_A = \{I_P|r|T|V|C\}$.

(g) Signar I_A amb la clau privada S_J, $I^*_A = S_J[I_A]$.

(h) Enviar (I_A, I^*_A) al gestor del joc.

2. El gestor del joc G realitza els passos següents:

(a) Verificar la signatura digital I^*_A amb la clau pública de J.

(b) Verificar les dades de l'aposta: I_P, T, C.

(c) Verificar que J disposa de crèdit suficient, $D_J - V \geq 0$.

(d) Si disposa de crèdit:

i. Actualitzar el crèdit del jugador, $D_J^{'} = D_J - V$.

ii. Calcular el rebut R_A de l'aposta I_A, $R_A = S_G[I_A^*]$.

iii. Calcular el rebut del crèdit disponible R_D, $R_D = S_G[I_J|D_J^{'}]$.

iv. Enviar ($D_J^{'}$,R_A,R_D) al jugador J.

(e) Si no disposa de crèdit, no s'accepta l'aposta.

3. J realitza els passos següents:

(a) Verificar la signatura digital de R_A.

(b) Verificar la signatura digital de R_D.

(c) Verificar que el crèdit $D_J^{'}$ és correcte.

3.3.8 Cobrar/pagar una aposta

Al finalitzar una partida, un jugador J cobra una aposta amb el protocol següent:

Protocol 7 $[I_A, I_A^*, R_A]$

1. G realitza les operacions següents:

(a) Verificar la signatura del rebut de l'aposta, R_A.

(b) Verificar la signatura de l'aposta, I_A^*.

(c) Calcular els guanys g del jugador J a la partida I_P amb l'aposta I_A.

(d) Calcular el nou crèdit disponible del jugador $D_J^{'}$, $D_J^{'} = D_J + g$.

(e) Calcular el rebut del crèdit disponible R_D, $R_D = [I_J|D_J^{'}]$.

(f) Enviar R_D al jugador J.

2. J realitza les operacions següents:

(a) Calcular els guanys $g^{'}$ de l'aposta I_A.

(b) Calcular el nou crèdit, $D_J^{''} = D_J + g^{'}$.

(c) Verificar la signatura de R_D. Si $D_J^{''} \neq D_J^{'}$, la signatura no es podrà verificar.

3.3.9 Seqüència de joc

Un jugador J empra el protocol 8 conjuntament amb el gestor del joc G. Els valors c_1 i c_2 són valors de 160 bits de llargada. Cada part en genera un i conjuntament en calculen el resultat, $c = c_1 \otimes c_2 \mod 37$, fent una XOR (OR Exclusiva) bit a bit. Fem mòdul 37 perquè aquests són els possibles resultats de la ruleta, del 0 al 36.

Protocol 8

1. J i G s'autentiquen amb el **Protocol 3**.
2. J i G inicien una partida amb el **Protocol 4**.
3. J fa una aposta i li comunica a G amb el **Protocol 6**.
4. J obté de manera aleatòria un valor c_1.
5. G obté de manera aleatòria un valor c_2.
6. J i G es comprometen a c_1 i c_2 respectivament, utilitzant el **Protocol 1**.
7. J i G obtenen c_2 i c_1 respectivament amb el **Protocol 2**.
8. J i G obtenen el resultat del joc $c = c_1 \otimes c_2 \mod 37$.
9. Si J ha obtingut un guany, G paga l'aposta a J amb el **Protocol 7**.

Capítol 4

Disseny de l'aplicació

4.1 Introducció

Ruleta ve del terme francès "roulette" que significa roda petita. El seu ús com a joc d'atzar, tot i que en configuracions diferents a l'actual, no està ben documentada fins ben entrada l'edat mitjana. La primera referència a una roda en moviment data de l'antiguitat. L'home, que des de sempre queda fascinat pels cossos en moviment constant, juga a la roda de la fortuna. La roda de la fortuna no era res més que un cercle que sempre girava on hi havia símbols esotèrics i servia per predir el destí o el futur de les persones.

La invenció d'una ruleta i de les seves normes de joc, molt similars a les que coneixem avui, s'atribueix al matemàtic, físic i filòsof francès Blaise Pascal. El 1654, incitat per un amic interessat en problemes d'apostes, Pascal va començar a mantenir correspondència amb Pierre de Fermat a qui va enviar una primera aproximació al càlcul de probabilitats. D'aquesta correspondència, i anys més tard, en sortirien les bases de la teoria de probabilitats. Aquell mateix any Pascal va tenir un accident amb un carruatge del qual miraculosament va salvar la vida.

Aquest incident va fer que Pascal canviés la seva manera de veure el món i s'acostés a la seva part més filosòfica i religiosa. La ruleta de Pascal té 36 números de l'1 al 36, en la qual s'ha estudiat curosament la posició de cadascun d'ells. Hi ha molta controvèrsia en els motius pels quals Pascal va escollir només 36 números després de l'incident del carruatge, ja que la suma de tots ells és un número amb molta simbologia religiosa, el 666.

La ruleta fou concebuda com un joc d'entreteniment i d'estudi de les proba-

bilitats matemàtiques. Quan es mira el vessant del negoci, però, no sembla gens rendible perquè donaria premis en una proporció de 36 sobre 36 (36/36). Dels 36 resultats possibles es donarien premis als 36.

A final del segle XIX, els germans Blanc, també francesos, van modificar la ruleta afegint-hi un número més: el zero i la van introduir al Casino de Montecarlo. Aquesta és la ruleta que ha arribat fins als nostres dies. Dels 37 resultats possibles només es donen premis a 36, 37/36, cosa que deixa un marge del 2,7% per al casino. En altres llocs on es juga a la ruleta, els països anglosaxons, s'hi va afegir un altre número: el doble zero, que dóna una mica més de marge al casino, 38/36, un 5,4%.

Quan surt el zero (o doble zero) significa que la casa guanya i tots els jugadors perden excepte els que han apostat al 0 o 00. Aquesta norma no la segueixen tots els casinos. Alguns apliquen la regla "de la presó": quan surt el 0 o el 00, permeten als jugadors retirar les apostes si no són el 0 o 00, o "deixar-les a la presó" fins al proper resultat. Si l'aposta queda sobre la taula i torna a sortir el 0 o 00 perden l'aposta que han fet. Els casinos que fan servir aquesta norma redueixen el seu marge a la meitat: 1,35% en les ruletes amb un 0 i 2,7% en les ruletes que tenen el 0 i el 00. Els jugadors, evidenment, prefereixen jugar als casinos que no hi ha el doble zero i apliquen la regla de la presó.

Hi ha tres tipus de joc de la ruleta: l'europea, l'americana i l'anglesa. Cadascuna difereix en aspectes com la taula de joc, la distribució i la quantitat dels números que hi ha a la ruleta i les apostes que s'hi poden fer. En aquest PFC s'ha decidit fer el joc de la ruleta europea, sense aplicar la norma de la presó; per tant, no explicarem ni les regles ni les apostes dels altres jocs de la ruleta que són força diferents.

4.1.1 Elements de la ruleta

El joc de la ruleta consta de 3 elements bàsics: la roda o cilindre, la bola i la taula (vegeu la figura 4.1).

La Roda o cilindre La roda és un cilindre de fusta dins del qual gira un altre cilindre de metall on hi ha 37 caselles, una per a cada número de l'1 al 36 i una per al zero. El zero és de color blanc o verd i els altres números rojos o negres. Quan Pascal va fer distribució dels números (vegeu figura 4.3) no ho va fer a l'atzar, sinó que va seguir les següents normes:

- Cada número que sigui més petit o igual a 18, haurà de tenir a cada

Figura 4.1: Taula de joc

costat un número més gran que 18. Excepte entre el 26 i el 32 (recordem que Pascal no va posar el zero) i entre el 5 i el 10.

- Els números la suma dels quals sigui un nombre parell seran de color negre i els que la seva suma sigui senar seran rojos. Excepte el 10 que també serà negre.
- Les caselles roges i negres han d'estar alternades.

Ara que ja sabem de quin color són i com han d'estar ordenades, vegem perquè estan en aquestes posicions i no en unes altres. Partim la ruleta per la meitat (vegeu la taula 4.1), a la part esquerra tindrem la llista de números del 26,3,35,...,5 i a la dreta la llista 32,15,19,...,10.

A cada costat de la ruleta hi ha 18 números. A cada costat hi ha 9 números negres i 9 de rojos, 9 que són parells i 9 que són senars, 9 entre l'1 i el 18 i 9 més entre el 19 i el 36. I a cada costat hi ha 6 números que pertanyen a la primera dotzena, 6 que pertanyen a la segona i 6 que pertanyen a la tercera. També n'hi ha 6 que pertanyen a la primera columna, 6 a la segona i 6 a la

tercera.

Els casinos acostumen a tenir equilibrats els dos cilindres de la ruleta, per evitar que un lleuger desequilibri faci que la bola sempre tendeixi a anar cap a un costat.

La bola és una esfera de color blanc d'uns 2 centímetres de diàmetre. Antigament eren de marfil i actualment són de plàstic. És un costum dels casinos canviar les boles cada dia.

Esquerra	Dreta
26	32
3	15
35	19
12	4
28	21
7	2
29	25
18	17
22	34
9	6
31	27
14	13
20	36
1	11
33	30
16	8
24	23
5	10

Taula 4.1: Divisió vertical de la ruleta

La taula és un espai rectangular que està situat a la vora de la roda. A la taula els números estan ordenats de manera consecutiva començant pel 0, i després de l'1 al 36 agrupats de 3 en 3. La figura 4.1 és una taula de joc. El fet que estiguin col·locats consecutivament a la taula i desordenats a la roda és perquè si es volen fer apostes a números de posicions consecutives a la roda, s'hagi de posar més d'una fitxa a la taula.

4.1.2 Regles del joc

L'objectiu del joc és encertar a quin número s'aturarà la bola que gira per la roda. Els jugadors que poden ser d'1 a 7 s'asseuen al voltant de la taula i hi col·loquen les fitxes: fan les apostes. Les fitxes de cada jugador són de colors diferents, per identificar cada color amb un jugador, i tenen escrit un número que representa el seu valor en diners. Les fitxes de la ruleta no es poden fer servir en altres llocs del casino, són especials i només per al joc de la ruleta.

El crupier ha d'accionar cada vegada la ruleta en sentit oposat al de la tirada anterior i ha de fer girar la bola en sentit contrari al que gira la ruleta. Mentre la ruleta i la bola giren, els jugadors fan les apostes fins que el crupier dóna l'ordre: "no feu més apostes". Els jugadors han de treure les mans de la taula i han d'esperar que la física faci el seu efecte i que a causa del fregament la roda deixi de girar i la bola s'aturi en alguna de les caselles numerades que té la roda.

Quan la bola s'atura, se sap el resultat i algun jugador podria caure en la temptació de posar alguna fitxa més a la taula. Per evitar aquest perill no es poden posar les mans a la taula des que el crupier dóna l'ordre fins que s'han retirat totes les fitxes; és a dir, quan s'han pagat totes les apostes. El crupier tampoc posa les mans a la taula, fa servir un bastó llarg acabat en un rectangle que li permet treure les fitxes que no han estat premiades i posar davant de cada jugador la fitxa o les fitxes premiades i el premi aconseguit.

4.1.3 Apostes i premis

Les apostes poden ser simples i múltiples. Les apostes simples són les que es fan amb una sola fitxa sobre la taula. Les múltiples són combinacions d'apostes simples.

L'aposta parell vol dir que apostarem a tots el números parells de la ruleta, anàlogament l'aposta manca vol dir que apostarem a tots els números que estiguin entre l'1 i el 18 de la ruleta. Les apostes **parell**(even), **senar**(odd), **roig**, **negre**, **manca**(1-18) i **passa**(19-36) són apostes a 18 números. És a dir, amb una sola fitxa apostem a 18 números, es paguen 1:1. Per fer una d'aquestes apostes hem de col·locar una o més fitxes a la casella corresponent de la taula de joc.

La taula està dividida en dotzenes (horitzontal) i columnes (vertical). Són apostes a 12 números i es paguen 2:1. Per fer una aposta a una **dotzena** o a una **columna** hem de col·locar una o més fitxes a les caselles on hi ha escrit 1a 12, 2a 12, 3a 12 per a les dotzenes o 2 to 1, 2 to 1, 2 to 1 per a les columnes.

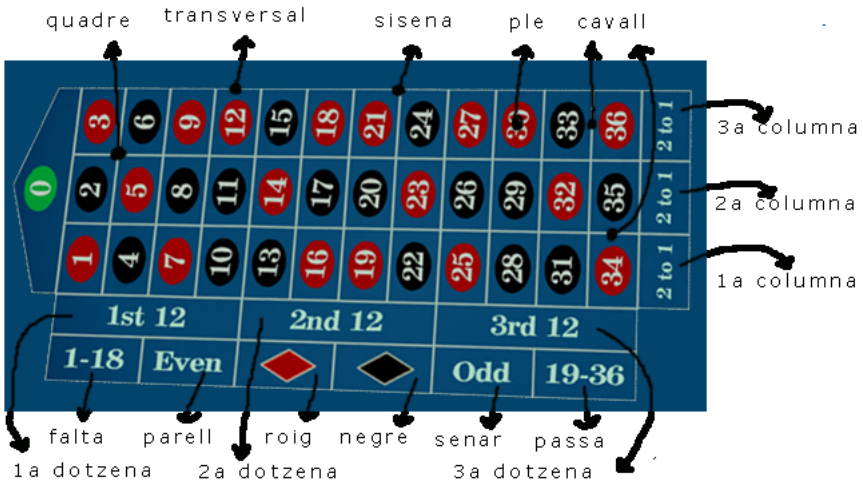

Figura 4.2: Apostes taula de joc

La **sisena** és una aposta a 6 números i es paga 5:1. Per apostar a una sisena hem de col·locar una o més fitxes a la línia que separa els dos blocs de tres números. El **quadre** és una aposta a 4 números, es paga 8:1. Per fer una aposta quadre hem de col·locar una o més fitxes al punt on s'ajunten els quatre números.

La **transversal** és una aposta de 3 números i es paga 11:1. Per fer aquesta aposta hem de col·locar una o més fitxes a la línia que marca els tres números. El **cavall** és una aposta a 2 números i es paga 17:1. Per fer l'aposta del cavall hem de col·locar una o més fitxes a la línia (horitzontal o vertical) que separa els dos números.

I el **ple** és una aposta a un sol número que es paga 35:1. Hem de col·locar una o més fitxes al quadre on hi ha el número al qual volem apostar.

En la figura 4.2, es representen els punts on s'han de col·locar les fitxes per fer cadascuna de les apostes simples. I a la taula 4.2 hi ha un resum dels

Apostes	Números	Premis
parell / senar	18	1:1
roig / negre	18	1:1
manca / passa	18	1:1
dotzena	12	2:1
columna	12	2:1
sisena	6	5:1
quadre	4	8:1
transversal	3	11:1
cavall	2	17:1
ple	1	35:1

Taula 4.2: Taula d'apostes

premis i dels números a què s'aposten. Els premis de la ruleta també tenen una lògica. Com més augmenta la probabilitat d'encertar el número, el premi que toca és menor. Els premis es calculen sobre els casos possibles: 36, tot i que n'hi ha 37. Per exemple, fem una aposta d'una fitxa als nombres senars:

$$\frac{\text{casos possibles}}{\text{casos probables}} = \frac{18}{36} = \frac{1}{2} \qquad 50\% \text{ de probabilitat}$$

El premi serà el doble del que hem apostat, o el que és el mateix: rebrem una fitxa per cada fitxa que haguem apostat(1:1). Si guanyem rebrem 2 fitxes, la de l'aposta i la del premi. Un altre exemple: fem una aposta d'una fitxa a un quadre:

$$\frac{\text{casos possibles}}{\text{casos probables}} = \frac{4}{36} = \frac{1}{9}$$

El premi serà 9 cops el que hem apostat, o el que és el mateix: rebrem 8 fitxes per cada fitxa que apostem (8:1). Si guanyem rebrem 9 fitxes, la de l'aposta i les 8 del premi.

Les apostes múltiples surten de la necessitat de fer apostes a posicions consecutives de la ruleta. L'aposta múltiple més comuna és apostar als veïns d'un número. En la figura 4.3 hi ha representada l'altra part de la taula d'apostes i com estan distribuïdes a la roda. Apostar als veïns del 9 significa apostar

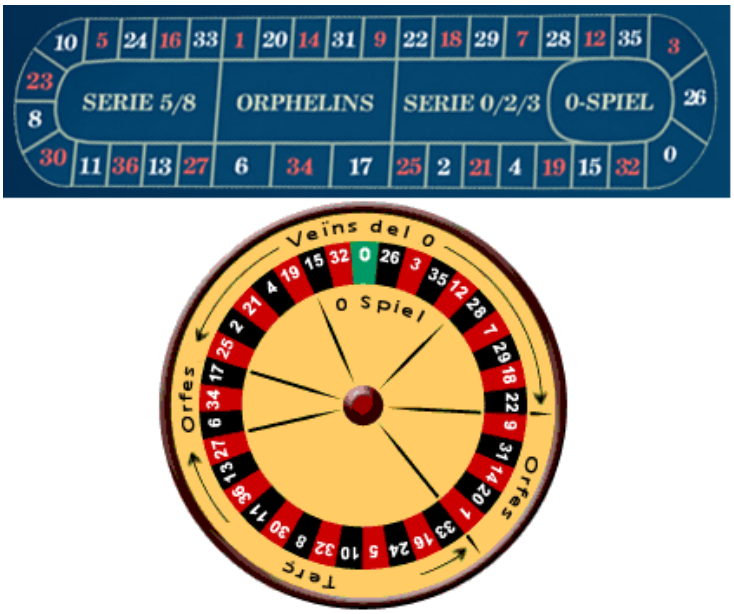

Figura 4.3: Apostes taula de joc

una fitxa al 18, una altra al 22, una altra al 9, una altra al 31 i una altra al 14; és a dir, dos números cap a l'esquerra i dos números cap a la dreta del número escollit. És el mateix que fer 5 apostes simples, una per cada número. L'aposta múltiple de veïns d'un número seria la representada en la taula 4.3.

Els premis de les apostes múltiples són els que marquen les seves apostes simples. En l'exemple dels veïns del 9, si el resultat fos el número 22, hauríem obtingut un premi perquè el 22 és un dels números veïns del 9. El premi que hauríem de cobrar seria 35:1, ja que l'aposta simple és un ple al número 22 d'1 fitxa. Es necessiten 5 fitxes (del valor que vulguem) per fer aquesta aposta o bé que el valor apostat sigui divisible entre 5, si apostem una fitxa d'1 euro estaríem apostant 0,20 cèntims per cadascuna de les apostes simples.

En les taules 4.4, 4.5, 4.6 i 4.7 hi ha les altres jugades múltiples que hi ha a la taula, per quines apostes simples estan compostes i quantes fitxes es necessiten per cobrir-les. Per cobrir les apostes múltiples fan falta tantes fitxes com apostes simples les componen o que el valor total apostat a la jugada sigui divisible també pel número d'apostes simples que la formen.

Veïns del número 9	
Números	Fitxes
18	1
22	1
9	1
31	1
14	1
Total	5

Taula 4.3: Veïns del número 9

Veïns 0	
Números	Fitxes
0/2/3	1
4/7	1
12/15	1
18/21	1
19/22	1
25/26/28/29	1
32/35	1
Total	9

Taula 4.4: Veïns del 0

Terç	
Números	Fitxes
5/8	1
10/11	1
13/16	1
23/24	1
27/30	1
33/36	1
Total	6

Taula 4.5: Terç

Orfes	
Números	Fitxes
1	1
6/9	1
14/17	1
17/20	1
31/34	1
Total	5

Taula 4.6: Orfes

0-spiel	
Números	Fitxes
0/3	1
12/15	1
26	1
32/35	1
Total	4

Taula 4.7: 0-spiel

4.2 Disseny UML

Descrita la mecànica del joc de la ruleta, i amb la informació de funcionament dels dos capítols anteriors podem començar a fer el disseny de l'aplicació.

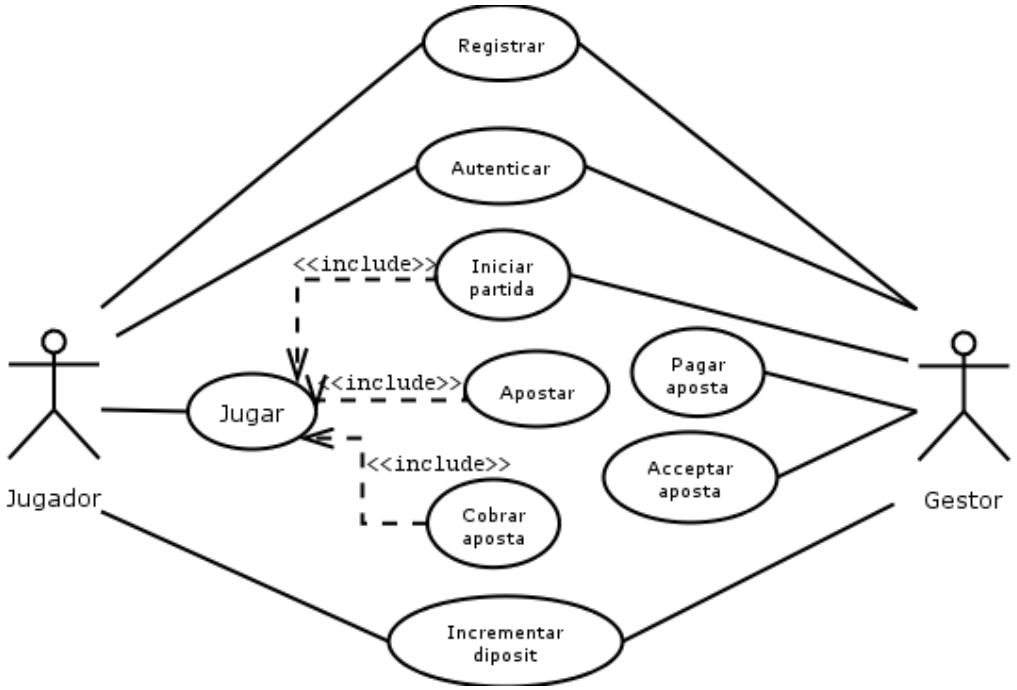

Figura 4.4: Diagrama casos d'ús general

Cas d'ús registre. El jugador ha d'estar registrat a l'aplicació per poder jugar. Per registrar-se necessitarà les seves claus i el seu certificat. El registre només es fa una vegada. Un cop el jugador està registrat el seu saldo és zero i l'haurà d'incrementar.

El gestor està a l'espera de rebre peticions de registre per part dels jugadors. Quan un jugador li envia una petició de registre, el gestor calcula l'identificador que tindrà el jugador a l'aplicació i guarda el nom, el cognom, el DNI, el certificat i l'identificador del jugador a la base de dades. El gestor retorna el seu certificat al jugador perquè pugui usar-lo durant el temps que estigui jugant. El diagrama de seqüència d'aquest cas d'ús és el de la figura 4.5.

Cas d'ús autenticar. El jugador, per autenticar-se davant del gestor, necessita el parell de claus, i el certificat. Recíprocament, el gestor també

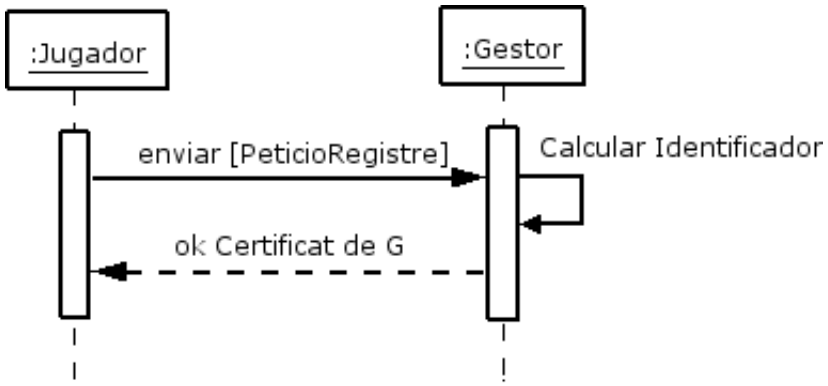

Figura 4.5: Diagrama de seqüència del cas d'ús Registrar

necessita el parell de claus i el certificat per autenticar-se davant del jugador. Tots dos han d'estar registrats a la base de dades de l'aplicació. En acabar l'autenticació el gestor haurà autenticat el jugador i el jugador també haurà autenticat el gestor. Durant l'autenticació, els missatges sempre s'enviaran xifrats. El diagrama de seqüència d'aquest cas d'ús és el de la figura 4.6.

Cas d'ús Jugar. Aquest cas d'ús només el té el jugador. El gestor està pendent de les peticions que rep dels jugadors. Jugar es compon de tres subcasos: iniciar una partida, apostar i cobrar l'aposta. Per jugar, un jugador ha d'estar autenticat i ha de tenir diners, no podrà jugar si no té el saldo més gran que zero.

Cas d'ús Iniciar una partida. El jugador ha d'estar autenticat i tenir crèdit per iniciar una partida. Demanarà al gestor que vol iniciar una partida. El gestor crearà un identificador per a la partida, el signarà i l'enviarà al jugador, i el farà servir mentre duri la partida per identificar-la. El diagrama de seqüència d'aquest cas d'ús és el de la figura 4.7.

Cas d'ús Incrementar el dipòsit. El jugador pot incrementar el seu dipòsit un cop s'ha autenticat i no ha començat a jugar o bé quan ha acabat una partida i encara no n'ha començat una altra. Per incrementar el dipòsit, el jugador crearà un identificador per al dipòsit i farà una petició al gestor. El gestor comprovarà que les dades de l'identificador del dipòsit són correctes i augmentarà el saldo del jugador. El gestor li comunicarà al jugador el nou saldo. El diagrama de seqüència d'aquest cas d'ús és el de la figura 4.8.

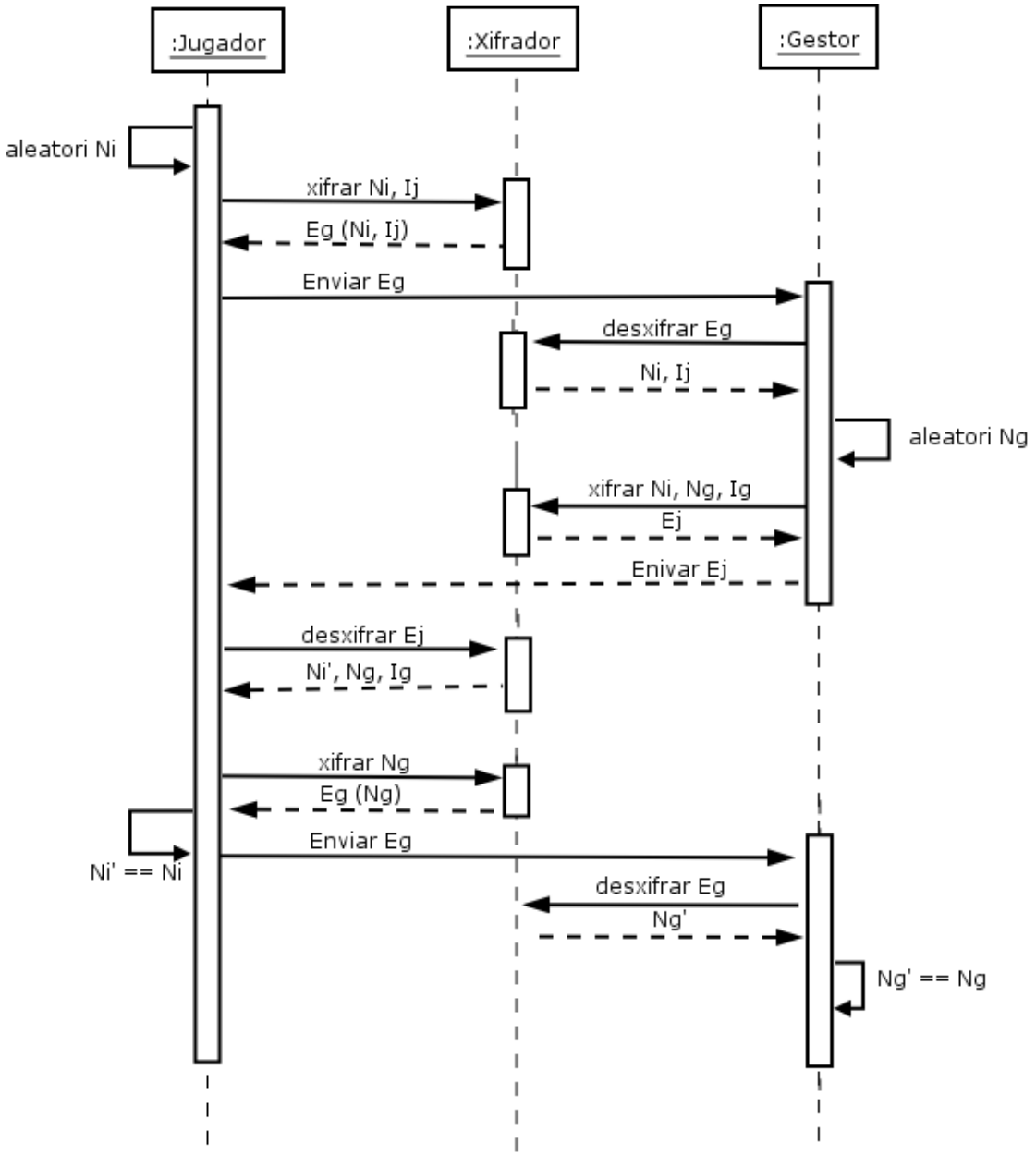

Figura 4.6: Diagrama de seqüència del cas d'ús Autenticar

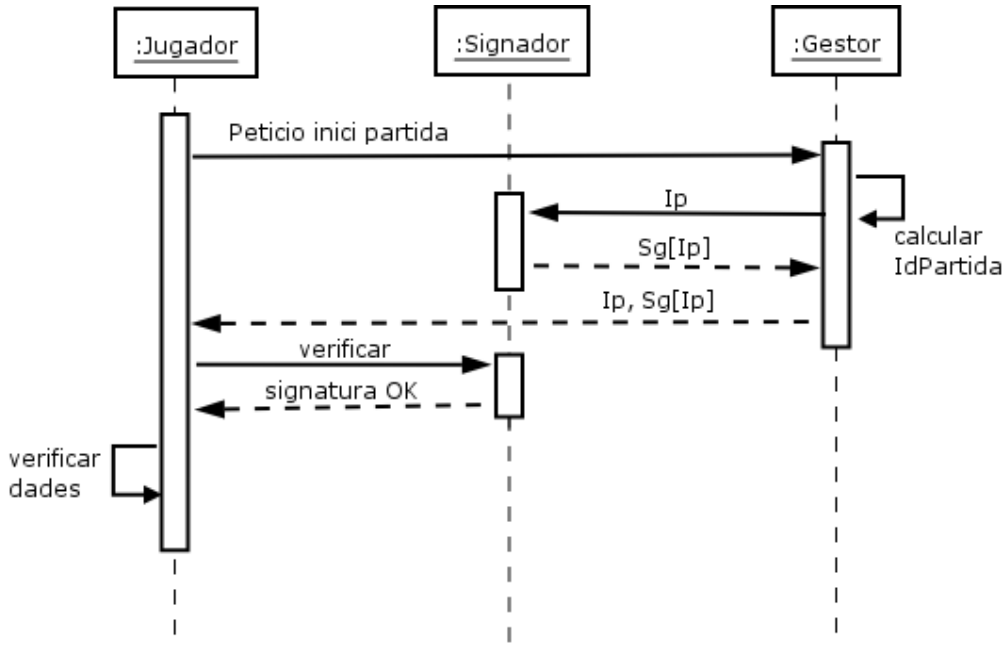

Figura 4.7: Diagrama de seqüència del cas d'ús Iniciar una partida

Cas d'ús Apostar i acceptar l'aposta. El jugador que està jugant una partida vol apostar. Posa una determinada quantitat de diners (sense superar el saldo que té) en forma de fitxa damunt d'un lloc concret de la taula. El jugador calcularà un identificador d'aposta i li demanarà al gestor que vol que li accepti l'aposta. El gestor rebrà l'aposta del jugador i calcularà un rebut per a aquesta aposta en concret i el nou saldo. El jugador ha de guardar aquest comprovant per poder cobrar aquesta aposta si guanya aquesta partida. El diagrama de seqüència d'aquest cas d'ús és el de la figura 4.9.

Cas d'ús Cobrar una aposta o pagar una aposta. El jugador ha guanyat i vol cobrar l'aposta; per tant, envia al gestor el rebut de l'aposta. El gestor comprovarà que el rebut pertany a l'aposta que el jugador havia fet i que ell mateix havia acceptat. El gestor enviarà al jugador el nou saldo amb els guanys. El jugador, per la seva banda, també calcularà els guanys que li corresponen i verificarà que el nou saldo és el que diu el gestor. El diagrama de seqüència d'aquest cas d'ús és el de la figura 4.10.

El diagrama d'estats ens permet identificar els estats per on passarà l'aplicació. És la representació del protocol de seqüència de joc (vegeu la figura 4.11). L'aplicació s'inicia i demana al jugador que s'autentiqui. Si no està

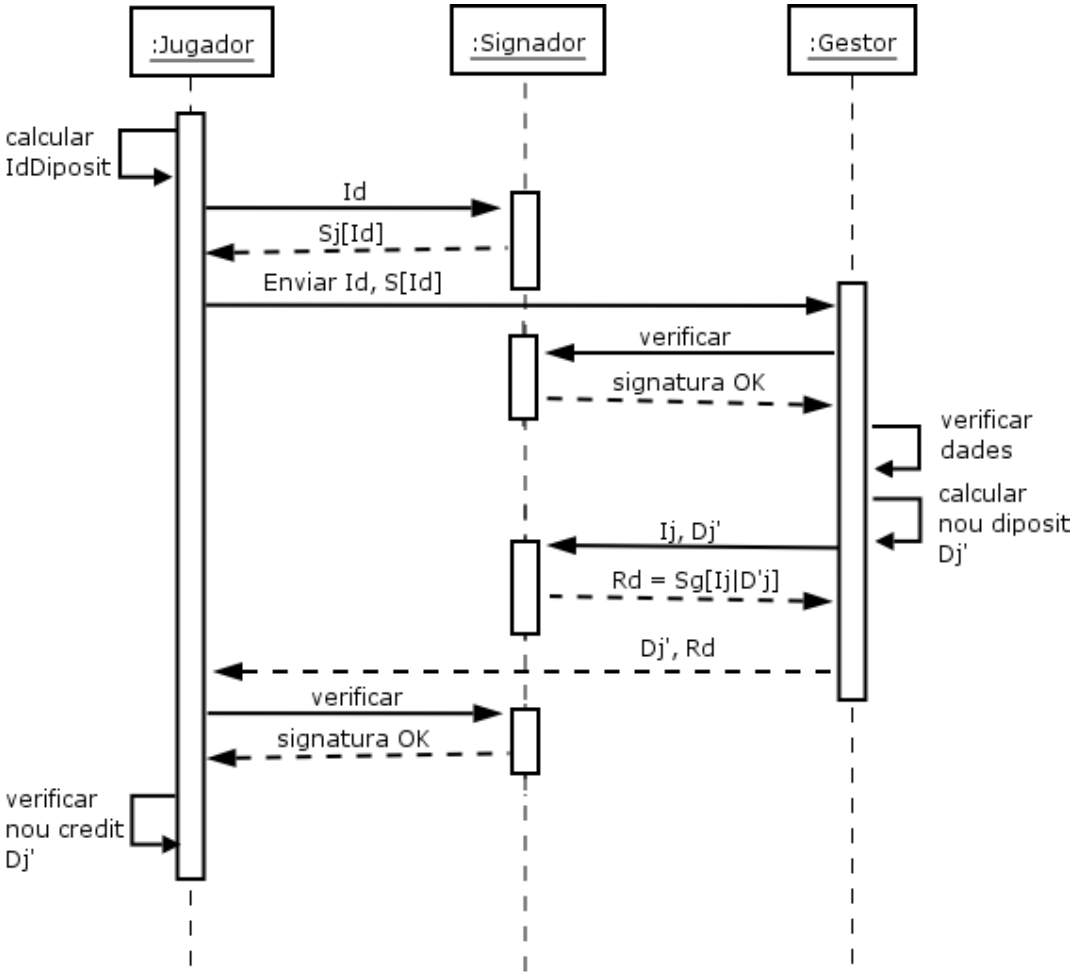

Figura 4.8: Diagrama de seqüència del cas d'ús Incrementar el dipòsit

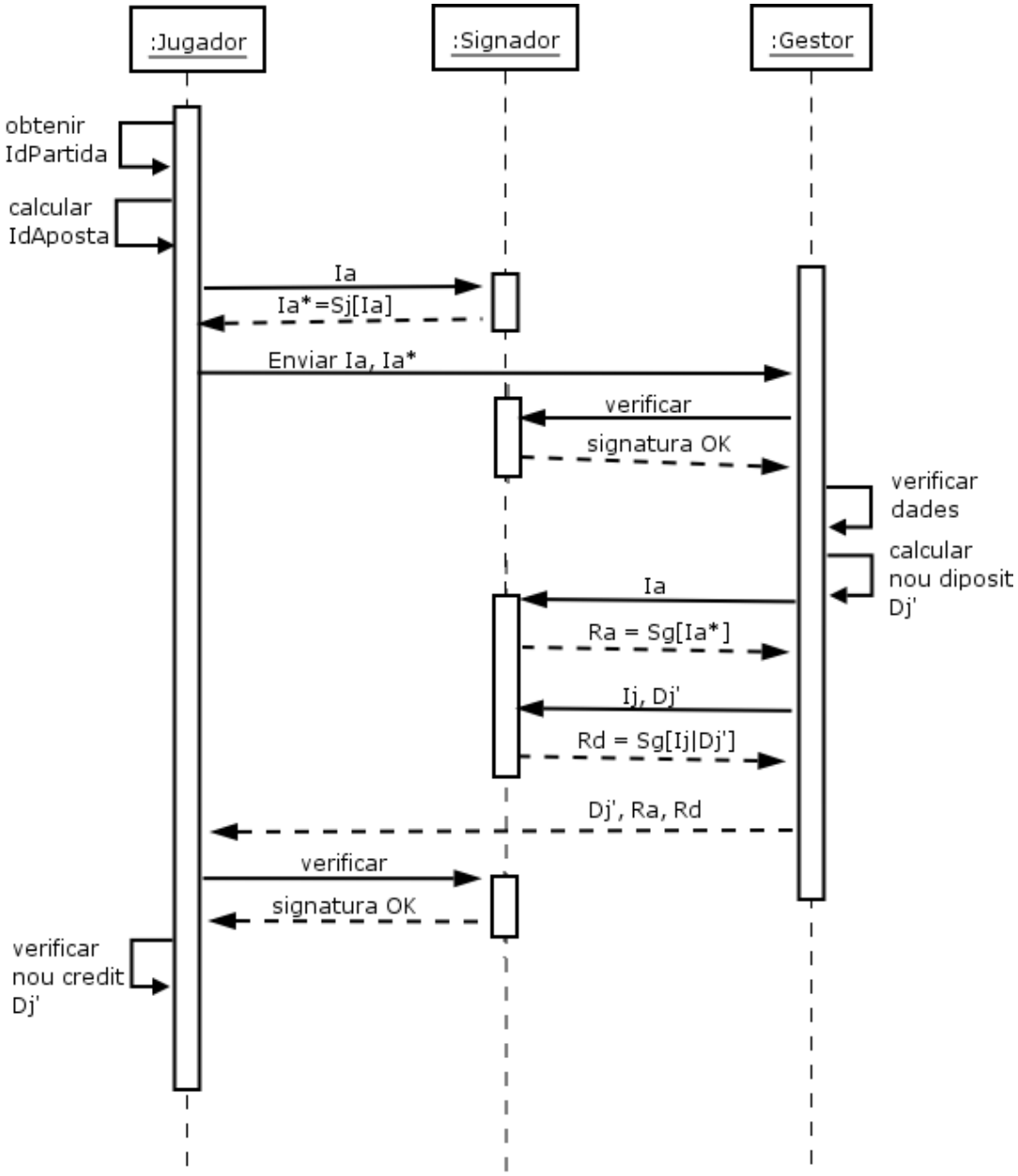

Figura 4.9: Diagrama de seqüència del cas d'ús Apostar

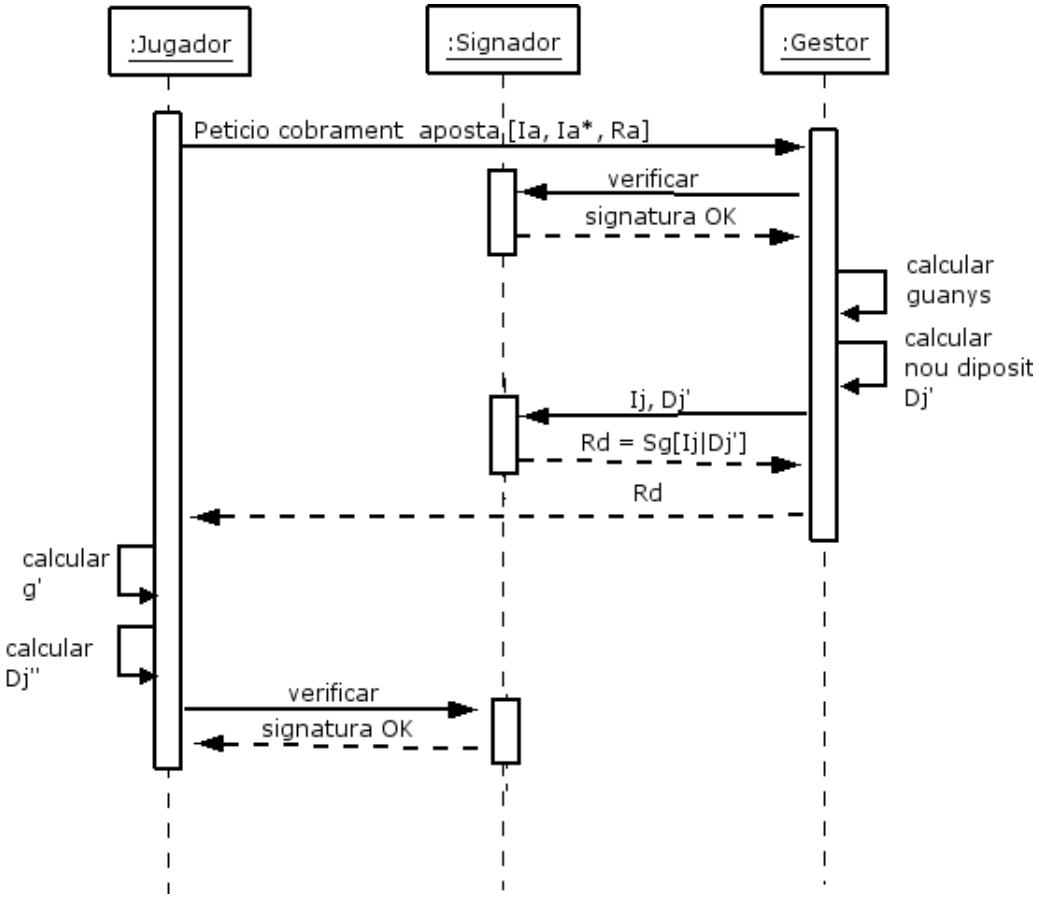

Figura 4.10: Diagrama de seqüència del cas d'ús Pagar/cobrar una aposta

registrat ha d'anar a l'estat Resgistrar. Un cop autenticat tornarà a l'estat Autenticar i s'autenticarà. Després passarà a l'estat Iniciar partida. Si el saldo del jugador és zero, ha d'incrementar obligatòriament el saldo. Pot anar a aquest estat encara que el saldo no sigui zero, però té l'oligació d'anar-hi si és igual a zero. Quan el jugador ja disposa de saldo, passa a l'estat Apostar. Es pot quedar en aquest estat mentre el seu saldo sigui més gran o igual a zero. Quan el saldo arribi a zero haurà de passar a l'estat següent, Jugar, fent girar la ruleta. Jugar és l'estat on es calcula el resultat del joc. Quan ja tenim el resultat passem a l'estat Obtenir guanys. En aquest punt podem tornar a iniciar una partida nova o podem acabar l'aplicació.

4.3 Diagrama de classes

Descrita la mecànica del joc de la ruleta, i amb la informació de funcionament dels dos capítols anteriors, podem construir el diagrama de classes que hem representat a la figura 4.12.

Totes les classes estan relaciones entre si amb associacions. Les associacions es convertiran en mètodes de les classes. La classe Aposta té un component que és la classe Concepte. Es considera que només es farà una aposta per partida, i aquesta aposta estarà composta de conceptes. Un concepte estarà compost de la quantitat de diners de l'aposta, del lloc on hem apostat (alguna de les apostes de la taula 4.2), i dels números als quals apostem. Per exemple, una fitxa als parells o una fitxa al roig són dos conceptes. Els conceptes poden formar part d'una mateixa aposta.

Les classes P12, SignerManager, ChiperManarer són components del gestor i del jugador per accedir al magatzem PKCS#12, signar dades i xifrar dades respectivament. La classe NumAleatori ens genera números aleatoris segurs i la fan servir la resta de classes. No representem totes les fletxes perquè el diagrama es faria il·legible.

4.4 Implementació

La implementació de l'aplicació s'ha fet amb el llenguatge de programació Java, en la seva versió 1.5.0. La instal·lació del programari està descrita a l'annex A, apartat A.2. S'han agrupat les classes en paquets que determinen la seva funcionalitat (vegeu la figura 8.2). En aquesta fase del projecte tenim

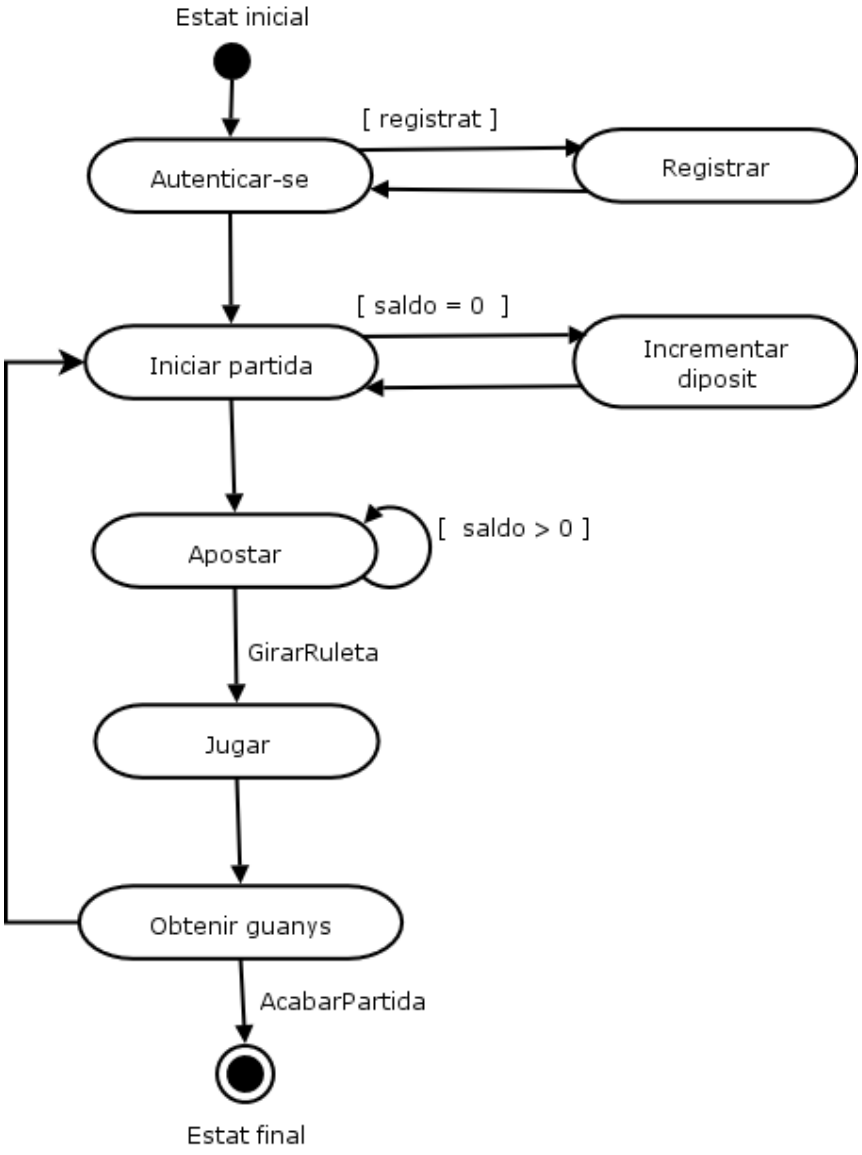

Figura 4.11: Diagrama d'estats

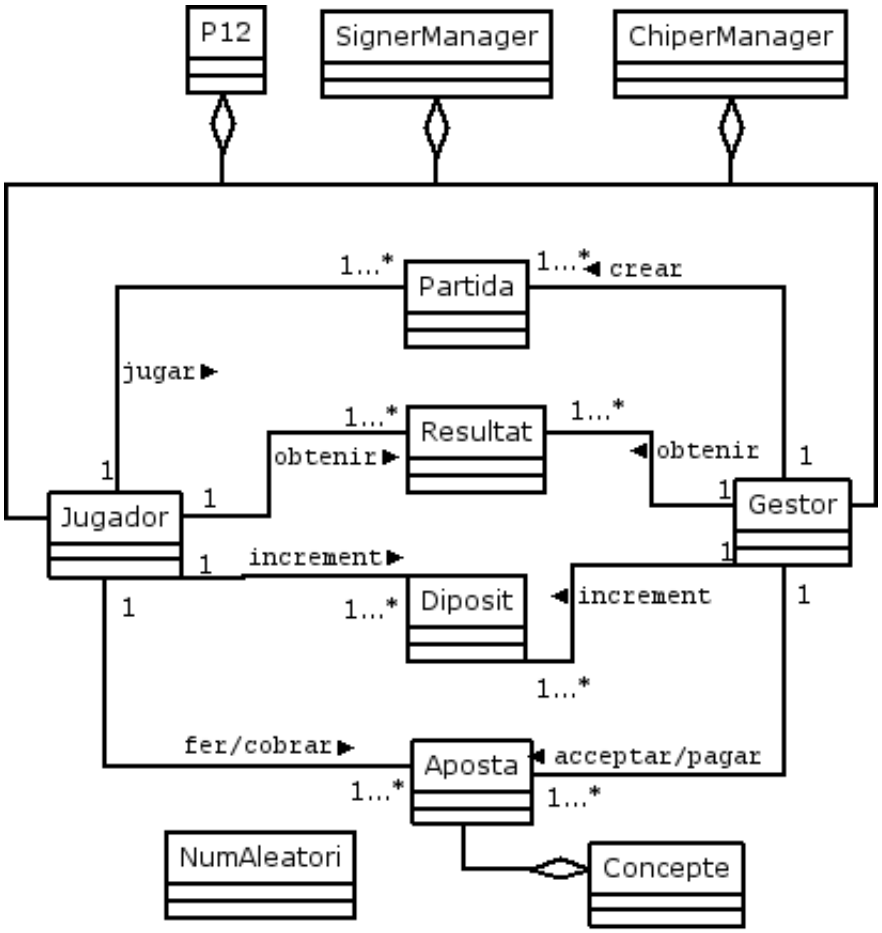

Figura 4.12: Diagrama de classes

4 paquets: criptografia, protocols, gestor i jugador.

Paquet criptografia. En aquest paquet agrupem les classes que tenen alguna cosa en comú amb l'apartat de criptografia. Les classes són:

P12. Aquesta classe ens proporciona els mètodes per accedir i manipular els magatzems de dades PKCS#12, que contenen el parell de claus i el certificat del gestor i dels jugadors.

NumAleatori. És una subclasse de la classe SecureRandom (nombre aleatori segur). Aquesta classe només té un mètode el qual ens retorna un nombre aleatori segur de la mida del número de bits que li demanem. Les classes que criden el mètode obtenirNumAleatori, ho fan amb una mida de 160 bits.

CipherManager.És la classe que ens proporciona els mètodes per xifrar i desxifrar missatges o documents.

SignerManager. Aquesta classe ens proporciona tres metodes: un mètode per signar un missatge o document, un altre per verificar la signatura i un altre per obtenir els certificats de la persona o de les persones que signen.

Resultat. Aquesta classe només té un mètode: obtenirResultat. A partir dels dos nombres aleatoris que generen el gestor i el jugador es calcula el resultat. Com que els números que s'han de tractar són grans (160 bits), s'usa la classe BigInteger (java.math.BigInteger), que permet fer operacions a nivell de bit amb números grans.

Paquet protocols En aquest paquet agrupem les classes identitat dels protocols, que són Partida, Diposit, Concepte i Aposta. Aquestes classes estan formades pels camps descrits en les taules 4.8, 4.9, 4.10 i 4.11. Els mètodes d'aquestes classes són el get i el set per a cada camp. Hem de destacar la classe Aposta que té un mètode més: obtenirGuanys(int resultat). Un cop obtenim el resultat calcularem els guanys de l'aposta amb aquest mètode.

Paquet jugador. Al paquet jugador només hi ha una classe: Jugador. Aquesta classe correspon a la part client de l'aplicació i permet jugar a la ruleta de manera segura. Els seus mètodes implementen la part del jugador dels protocols criptogràfics. El mètode jugar() implementa els protocols 1 i 2, autenticar() el protocol 3, iniciPartida() el 4, incrementarDiposit() el 5, apostar() el 6 i obtenirGuanys() el 7.

classe Partida

String joc;
byte[] numaleatori;
String T; //marca de temps
int numPartides;
String idPartida;

Taula 4.8: Classe Partida

classe Diposit

byte[] numAleatori;
String T; //marca de temps
String tarjetaCredit;
float increment;
String idDiposit;

Taula 4.9: Classe Diposit

classe Concepte
String nom; int[] numeros; float valor;

Taula 4.10: Classe Concepte

Classe Aposta

String idPartida;
String idAposta;
byte[] numAleatori;
String T; //marca de temps
float V;
Vector<Concepte> vc;

Taula 4.11: classe Aposta

Paquet gestor. Al paquet gestor només hi ha una classe: Gestor. Aquesta classe correspon a la part servidor de l'aplicació. El gestor escolta les peticions dels jugadors i les contesta. Els seus mètodes implementen la part del gestor dels protocols criptogràfics. El mètode compromis() implementa el protocol 1, obertura() el 2, autenticarGestor() i autenticarJugador() els passos 2 i 4 del protocol 3, iniciarPartida() el 4, incrementarDiposit() el 5, apostar() el 6 i obtenirGuanys() el 7.

Capítol 5

Representació de dades: XML

L'XML és l'acrònim de eXtensible Markup Language i és una especificació de la W3C, que deriva del SGML(Standard Generalized Markup Language). L'XML serveix per guardar i estructurar dades, ja sigui per enviar-les entre aplicacions a través d'Internet com per intercanviar informació entre organitzacions. Aquesta forma de representar les dades s'ha fet tan popular perquè no depèn de cap plataforma ni de cap llenguatge, i el seu ús és totalment lliure. Diem que no depèn de cap plataforma perquè un document o missatge XML és un document de text pla.

Un document XML està format per una capçalera i un conjunt d'etiquetes. L'especificació diu que com a mínim els documents han d'estar "ben formats", que totes les etiquetes que s'obren s'han de tancar: `<etiqueta></etiqueta>`. Per obrir una etiqueta la posem entre <nomEtiqueta> i per tancar-la </nomEtiqueta>, si està en una sola línia: <nomEtiqueta/>. Les etiquetes s'han de tancar en l'ordre invers al que s'han obert. Entre les etiquetes posarem les dades. Aquest seria un document buit que compleix l'especificació XML, tot i que no conté cap dada.

```
<?xml version="1.0">
<Ruleta></Ruleta>
```

Els documents XML, opcionalment, poden estar validats. Es diu validar, en el sentit que ha de complir una estructura predeterminda: l'esquema o dtd (Document Type Definition). Un document dtd defineix els components vàlids d'un document XML i els defineix mitjaçant una llista d'elements. En aquesta llista es dóna informació dels elements, dels tipus dels elements, o en quin ordre s'han de posar. Es pot veure l'esquema ruleta.dtd que han

de validar tots els documents XML de l'aplicació a l'appendix A, a l'apartat A.5.2. Per indicar-li quin esquema ha de complir, inserim la línia: `<!DOCTYPE Ruleta SYSTEM "ruleta.dtd">` que significa que un document de tipus Ruleta ha de complir l'esquema "ruleta.dtd".

```
<?xml version="1.0">
<!DOCTYPE Ruleta SYSTEM "ruleta.dtd">
<Ruleta></Ruleta>
```

Una restricció del format en documents XML és que són de text. Si les aplicacions han d'intercanviar-se algun missatge que hagi de contenir dades en binari, per exemple, una signatura digital, un missatge encriptat o un certificat, s'hauran de representar com si fossin text. En primer lloc les codificarem en base 64 i quan les hàgim rebut, les decodificarem altre cop a binari.

5.1 Format dels documents

Quan dues aplicacions s'han d'intercanviar informació primer han d'acordar com seran els documents i quines dades són les que s'han de posar. Mirant els diagrames de seqüència del capítol 4, veiem que entre el jugador i el gestor sovint s'envien missatges amb un paràmetre. Cadascun d'aquests paràmetres és un document XML, formatat i validat d'una manera adequada. En cada moment del joc necessitarem documents que continguin dades diferents. Vegem cadascun d'ells.

5.1.1 Registre

Aquest és el document XML que s'intercanvien el jugador i el gestor en el diagrama de seqüència del cas d'ús Registrar, figura 4.5.

```
<Ruleta>
    <Registre>
        <Certificat/>   certificat (codificat en base 64)
    </Registe>
</Ruleta>
```

5.1.2 Protocol Autenticació

Documents utilitzats per transmetre les dades de l'autenticació entre el jugador i el gestor en el diagrama de seqüència del cas d'ús Autenticar, figura

4.6. En el protocol d'autenticació explicat a l'apartat 3.3.4 s'envien tots els missatges xifrats. Es construeix un document XML que després es xifra i s'envia. S'aprofita la mateixa estructura per a tots els missatges. El primer missatge, que és el que envia el jugador al gestor, correspon al pas 1:

```
<Ruleta>
    <ContenidorXML>
        <Element1>      N_i (codificat en base 64)
        <Element2>      I_J (codificat en base 64)
        <Element3>      no s'usa
    </ContenidorXML>
</Ruleta>
```

El segon missatge, que és el que envia el gestor al jugador, correspon al pas 2:

```
<Ruleta>
    <ContenidorXML>
        <Element1>      N'_i (codificat en base 64)
        <Element2>      N_G (codificat en base 64)
        <Element3>      I_G (codificat en base 64)
    </ContenidorXML>
</Ruleta>
```

El tercer missatge, que és el que envia el jugador al gestor, correspon al pas 3:

```
<Ruleta>
    <ContenidorXML>
        <Element1>      N'_G (codificat en base 64)
        <Element2>      no s'usa
        <Element3>      no s'usa
    </ContenidorXML>
</Ruleta>
```

El quart missatge és la petició de saldo del jugador i també la resposta del gestor.

La petició:

```
<Ruleta>
    <ContenidorXML>
        <Element1>      I_J (codificat en base 64)
        <Element2>      no s'usa
        <Element3>      no s'usa
    </ContenidorXML>
</Ruleta>
```

La resposta:

```
<Ruleta>
    <ContenidorXML>
        <Element1>      saldo del jugador
        <Element2>      no s'usa
        <Element3>      no s'usa
</ContenidorXML>
</Ruleta>
```

5.1.3 Protocol Inciar una partida

Documents utilitzats per transmetre les dades per iniciar una partida entre el jugador i el gestor en el diagrama de seqüència del cas d'ús Iniciar partida, figura 4.7. El protocol Iniciar Partida està explicat a l'apartat 3.3.5. S'envien dos missatges: la petició d'inici de partida i la resposta del gestor.

En el primer missatge, el jugador li demana al gestor iniciar una partida:

```
<Ruleta>
    <Partida>
        <Certificat/>   I_J (codificat en base 64)
        <Joc/>          nom del joc al qual es vol jugar
    </Partida>
</Ruleta>
```

En el segon, el gestor li retorna la partida que jugarà:

```
<Ruleta>
    <Partida>
        <Signatura/>    signatura de les dades
        <IdPartida>
            <J/>        nom del joc
            <r/>        valor aleatori (codificat en base 64)
            <T/>        data actual
            <N/>        nombre de partides
        </IdPartida>
    </Partida>
</Ruleta>
```

5.1.4 Protocol Incrementar el dipòsit

Són els documents utilitzats per transmetre les dades per incrementar el dipòsit del jugador en el diagrama de seqüència del cas d'ús Incrementar dipòsit, figura 4.8. El protocol Increment de dipòsit està explicat a l'apartat 3.3.6. S'envien dos missatges: la petició d'increment i el resultat de l'increment.

En el primer missatge, el jugador demana incrementar el seu dipòsit en un valor:

```
<Ruleta>
    <IncrementDiposit>
        <Signatura/>        signatura de les dades
        <IdDiposit>
            <r/>            valor aleatori (codificat en base 64)
            <T/>            data actual
            <V/>            valor
            <B/>            número de la targeta de crèdit
        </IdDiposit>
    </IncrementDiposit>
</Ruleta>
```

En el segon missatge, el gestor li retorna el seu nou saldo:

```
<Ruleta>
    <IncrementDiposit>
        <RebutCredit/>      signatura de les dades (codificat en base 64)
        <Diposit/>          valor del nou crèdit del jugador
    </IncrementDiposit>
</Ruleta>
```

5.1.5 Protocol Fer una aposta

Són els documents utilitzats per transmetre les dades per tal de fer una aposta en el diagrama de seqüència del cas d'ús Apostar, figura 4.9. El protocol Fer una aposta està explicat a l'apartat 3.3.7. S'envien dos missatges: la petició del jugador al gestor perquè accepti l'aposta i la resposta del gestor quan retorna el rebut de l'aposta.

En el primer missatge, el jugador fa una aposta:

```
<Ruleta>
    <Aposta>
        <IdAposta>
            <IdPartida/>   identificador d'una partida
            <r/>           valor aleatori (codificat en base 64)
            <T/>           data actual
            <V/>           valor total de l'aposta
            <C/>           concepte de l'aposta
        </IdAposta>
    </Aposta>
</Ruleta>
```

L'identificador de la partida tindrà la mateixa estructura que <IdPartida> de l'apartat 5.1.3. L'element concepte tindrà aquesta forma:

```
<C>
    <Valor/>    valor de l'aposta
    <Nom/>      nom de l'aposta [vegeu la taula 4.2]
    <Numero/>   números que componen l'aposta
</C>
```

Les apostes simples estan formades per una quantitat de números determinada. L'element <Numero/> es repetirà tantes vegades com números tingui l'aposta (1,2,3,4 i 6). Només s'utilitzarà l'element <Numero/> en les apostes sisena, quadre, transversal cavall i ple. A les restants parell, senar, roig, negre, manca, passa, dotzena i columna es deixarà l'element <Numero/> en blanc. Vegeu els exemples a l'annex A, apartat A.5.1. En la ruleta es poden fer apostes múltiples. L'element <C> es repetirà tantes vegades com apostes simples vulgui fer el jugador. La suma dels elements <Valor> de tots els elements <C> serà igual a l'element V de l'aposta.

En el segon missatge, el gestor li retorna el rebut de l'aposta:

```
<Ruleta>
    <Aposta>
        <RebutCredit/>   signatura de les dades (codificat en base 64)
        <RebutAposta/>   signatura de les dades (codificat en base 64)
        <Diposit/>       nou crèdit del jugador després d'apostar
    </Aposta>
</Ruleta>
```

5.1.6 Protocol Cobrar/pagar una aposta

Documents utilitzats per transmetre les dades per cobrar una aposta en el diagrama de seqüència del cas d'ús Cobrar una aposta o pagar una aposta, figura 4.10. El protocol Cobrar/pagar una aposta està explicat a l'apartat 3.3.8. S'envien dos missatges: la petició que fa el jugador al gestor per cobrar l'aposta i la resposta del gestor amb el nou crèdit disponible.

En el primer missatge, el jugador vol cobrar una aposta:

```
<Ruleta>
    <ContenidorXML>
        <Element1>     signatura de l'aposta
        <Element2>     I_J identificador del jugador
        <Element3>     IdPartida, identificador de la partida
    </ContenidorXML>
</Ruleta>
```

En el segon missatge, el gestor retorna al jugador el rebut del nou crèdit:

```
<Ruleta>
    <IncrementDiposit>
        <RebutCredit/>   signatura de les dades (codificat en base 64)
        <Diposit/>       valor del nou crèdit del jugador
    </IncrementDiposit>
</Ruleta>
```

5.1.7 Protocols de Compromís i d'Obertura

Els protocols de Compromís i d'Obertura formen part de la seqüència de joc. Usem la mateixa estructura per a tots dos, encara que s'anomenin de manera diferent.

Protocol Compromís:

```
<Ruleta>
    <Compromis>
        <Signatura/>   signatura de les dades (codificat en base 64)
        <Valor/>       valor binari (codificat en base 64)
    </Compromis>
</Ruleta>
```

Protocol Obertura:

```
<Ruleta>
    <Compromis>
        <Signatura/>   signatura de les dades (codificat en base 64)
        <Valor/>       valor binari (codificat en base 64)
    </Compromis>
</Ruleta>
```

El valor va canviant perquè depèn del pas del protocol de compromís (vegeu apartat 3.3.2) en què ens trobem. El primer missatge l'envia el jugador al gestor, <Valor> és $H(c_1)$. El gestor li torna el missatge, en aquest cas és $H(c_2)$. En el protocol d'Obertura quan el jugador envia les dades al gestor és c_1 i és c_2 quan el gestor li respon.

5.2 Disseny UML

A la figura 5.1 hi ha el diagrama de classes ampliat per aquesta fase. Només representem les noves classes i les relacions amb les que ja existien a la fase anterior. Les classes que acaben amb les lletres XML ens permeten crear i recuperar documents XML per a cada missatge. Així, per exemple, PartidaXML permet crear els documents XML, descrits a 5.1.3, a partir d'una partida, que es representa per la classe Partida. També permet obtenir un objecte Partida a partir d'un document XML que representa una partida. Les classes que comencen per Id són subclasses de la classe Element (org.jdom.Element) i s'encarreguen exclusivament dels elements identificadors.

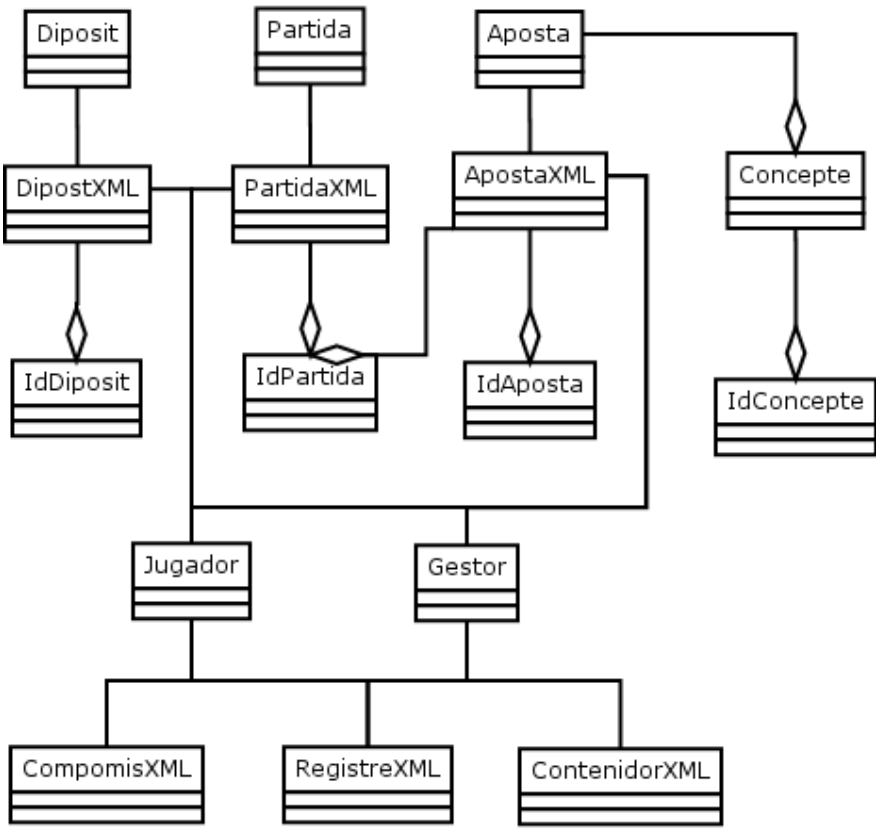

Figura 5.1: Diagrama de classes ampliat XML

5.3 Implementació

Per a la implementació amb Java s'ha utilitzat la llibreria JDOM (de lliure distribució) perquè és molt fàcil d'utilitzar. Aquesta llibreria porta les classes necessàries per generar i treballar amb documents XML. El procés d'instal·lació d'aquest programari és a l'annex A, apartat A.4.

Hem afegit un nou paquet als que ja teníem de la fase anterior. El **paquet xml** conté les classes següents:

DipositXML. Classe que permet crear i recuperar documents XML, amb l'estructura `<IncrementarDiposit></IncrementarDiposit>`.

PartidaXML. Classe que permet crear i recuperar documents XML, amb l'estructura `<Partida></Partida>`.

ApostaXML. Classe que permet crear i recuperar documents XML descrits, amb l'estructura `<Aposta></Aposta>`.

IdDiposit. Classe que permet crear un element identificador de dipòsit, amb l'estructura `<IdDiposit></IdDiposit>`, per poder-lo incloure directament al document XML. També permet fer l'acció contrària: a partir de la part IdDiposit d'un document DipositXML recupera l'element IdDiposit.

IdPartida. Té les mateixes funcionalitats que la classe anterior referides a l'identificador de la partida, amb l'estructura `<IdPartida></IdPartida>`.

IdAposta. Té les mateixes funcionalitats que la classe anterior referides a l'identificador de l'aposta, amb l'estructura `<IdAposta></IdPartida>`.

IdConcepte. Té les mateixes funcionalitats que la classe anterior referides a un concepte, amb l'estructura `<Concepte></Concepte>`.

CompromisXML. Classe que permet crear i recuperar documents XML, amb l'estructura `<CompromisXML></CompromisXML>`.

RegistreXML.Classe que permet crear i recuperar documents XML, amb l'estructura `<Registre></Registre>`.

ContenidorXML.Classe que permet crear i recuperar documents XML, amb l'estructura `<ContenidorXML></ContenidorXML>`.

Capítol 6

Comunicació de components: RMI

Fins ara, només hem dedicat esforços a fer l'aplicació segura. En aquest capítol explicarem com fer la part remota del joc electrònic.

En un primer plantejament es podria implementar un sistema de comunicació propi basat en sockets sobre un protocol com el TCP/IP. Aquesta implementació és molt costosa i hem decidit fer servir el sistema de comunicació d'objectes distribuïts de Java: RMI.

RMI són les sigles de Remote Method Invocation. Java incorpora aquesta tecnologia a l'API estàndard. Aquesta tecnologia permet, a un objecte que s'està executant en una màquina virtual de Java, cridar un mètode d'un altre objecte que també s'està executant en una màquina virtual diferent. Les màquines virtuals poden estar executant-se en un mateix dispositiu o en dispositius diferents.

Les aplicacions RMI, sovint, estan formades per dos parts: la part client i la part servidor. El servidor ha de crear objectes remots i fer-los públics mitjançant una interfície perquè els clients puguin invocar-ne els mètodes. El servidor crida el registre per tal d'associar un nom a cada objecte remot. Si un client vol executar un mètode d'un objecte remot, primer ha de buscar el seu nom al registre i després cridar el mètode concret. La implementació de la interfície queda oculta i el client no arriba mai a saber què és el que s'està executant.

6.1 Disseny UML

A la figura 6.1 hi ha el diagrama de classes ampliat per aquesta fase. La classe Servidor és la que farà el paper de servidor explicat a la secció anterior i la classe Jugador serà la part client. Una instància de la classe Gestor serà l'objecte remot i la classe InterfícieRemota serà la interfície on es faran públics els mètodes remots.

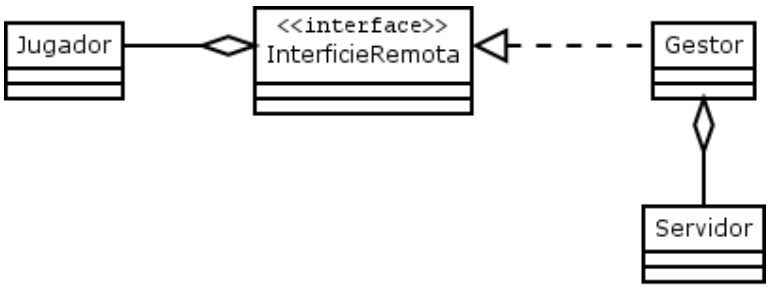

Figura 6.1: Diagrama de classes ampliat RMI

6.2 Implementació

Per a la implementació en Java no hem necessitat cap programari addicional perquè RMI està inclòs dins de l'API de Java. Tot i que no es necessària cap instal·lació addicional, sí que es necessari executar unes instruccions en un cert ordre perquè RMI funcioni. El procés d'execució per posar en marxa l'RMI està explicat a l'annex A, apartat A.6.

Hem afegit un nou paquet als que ja teníem de la fase anterior. El **paquet rmi** té una classe: Servidor, que és l'encarregada fer públic el nom de l'objecte remot, ho fa de la següent manera:

```
...
Gestor  g = new Gestor("gestor.p12", "exportgestor");
Naming.rebind("rmi://localhost/ServidorRuleta",g);
...
```

En la primera línia, la classe Servidor crea una instància de l'objecte Gestor i en la segona li dóna un nom: ServidorRuleta, aquest nom és el que haurà de fer servir el client.

Un cop hem implementat la classe Servidor, hem d'escriure la classe de la interfície: InterficieRemota. Hem de fer la tria dels mètodes del Gestor que s'han d'executar remotament; no és necessari que tots es puguin executar de manera remota.

Fent un cop d'ull als diagrames de seqüència descrits al capítol 4, veiem que els mètodes remots hauran de ser tots aquells en què el jugador demana algun servei al gestor i el gestor li torna una resposta. Els mètodes remots els descriu la classe InterficieRemota:

```
public interface InterficieRemota extends Remote {

  public void demanarRegistre(...) throws RemoteException, Exception;
  public byte[] demanarCertificat() throws RemoteException, Exception;
  public byte[] autenticarGestor(...) throws RemoteException;
  public void autenticarJugador(...) throws RemoteException;
  public String incrementarDiposit(...) throws RemoteException,
                                        Exception;
  public float obtenirSaldo(...) throws RemoteException, SQLException;
  public String iniciarPartida(...) throws RemoteException, Exception;
  public String apostar(...)throws RemoteException, Exception;
  public String compromis(...) throws RemoteException, Exception;
  public String apertura(...) throws RemoteException, Exception;
  public int obtenirResultatJoc(...) throws RemoteException,
                                    SQLException, Exception;
  public String obtenirGuanys(...) throws RemoteException, Exception;

}
```

Per tal que el Gestor pugui fer que els seus mètodes siguin remots, hem de canviar el tipus de classe que representa i els ha d'implementar. No cal escriure el codi dels mètodes perquè ja estan escrits a la fase anterior, però sí que cal modificar la capçalera de la classe Gestor i posar-hi que és subclasse d'UnicastRemoteObject i que implementa la interfície InterficieRemota:

```
public class Gestor extends UnicastRemoteObject
                    implements InterficieRemota { ... }
```

Per completar el procés, a partir del nom de l'objecte remot, el client n'obté una instància per poder exercutar-ne els mètodes.

```
private InterficieRemota g; //Instancia de l'objecte remot
....
public Jugador(String nomp12, String parauladepas) {
...
g = (InterficieRemota)Naming.lookup("rmi://localhost/ServidorRuleta");
...
}
```

Un cop el jugador té una instància de l'objecte remot ja el pot fer servir de la mateixa manera que faria servir qualsevol altre objecte.

Capítol 7

Gestió de la informació : base de dades

La gestió de la informació es fa necessària en el moment en què permetem que més d'un jugador pugui jugar, a la vegada, a la ruleta. El gestor, per atendre les peticions dels diferents jugadors, ha de poder guardar i recuperar les dades de cadascun d'ells, tant pel que fa a les dades personals, com pel que fa a les dades de les partides que juguen i de les apostes que fan durant el joc.

Una base de dades ens permetrà que les dades siguin perdurables en el temps. Si els jugadors ja han acabat les seves partides, podrem continuar consultant les dades i auditar-les en cas d'error. També amb programes de Data Ware House i tècniques de mineria de dades podem extreure informació de les dades emmagatzemades sobre el comportament dels jugadors, a quins números aposten més, a quina hora és connecten, si les apostes que fan segueixen algun patró, etc.

7.1 Model entitat relació

A la figura 7.1 hem representat el model entitat relació per a la implementació de la persistència de dades. Hi trobem dues entitats rellevants: aposta i moviments.

L'entitat dèbil aposta és necessària perquè la ruleta té apostes múltiples (formades per apostes simples) i, a més, un jugador pot fer més d'una aposta per a cada partida, ja siguin simples com múltiples. L'entitat apostes representa l'aposta total que fa el jugador en cada partida i l'entitat aposta, cadascuna

de les apostes simples.

L'entitat moviment representa el moviment que tindrà el camp saldo del jugador. Tenim moviments de tres tipus: 'increment', 'aposta' i 'pagament'. L'increment i el pagament augmentaran el saldo i l'aposta el disminuirà.

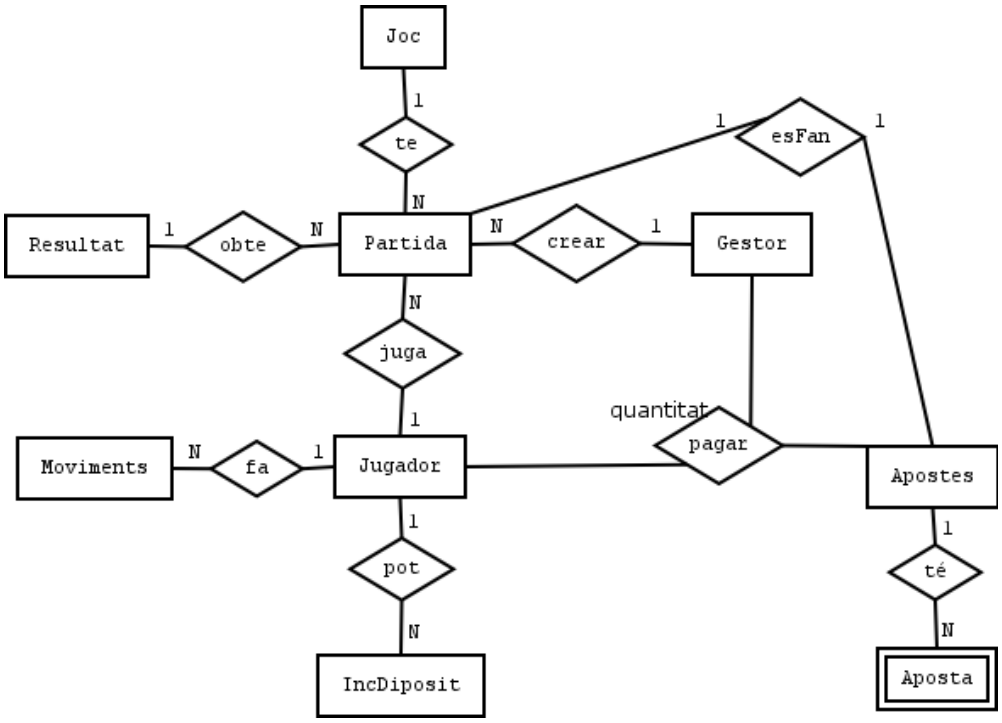

Figura 7.1: Diagrama entitat relació

7.2 Model relacional

El diagrama entitat relació es tradueix al model relacional tal com segueix:

Joc(idJoc, nomJoc)
Jugador(idJugador, dni, nom,cognom, dataAlta, dataBaixa, certificatHash, certificat, saldo)
Gestor(idGestor, dni, nom, cognom, dataAlta, dataBaixa, certificatHash, certificat)
Resultat(idResultat, resultat, c1, c2, data)
IncDiposit(idJugador, data, increment, targetaCredit)

Moviments(idMoviment,idJugador, data, valor, concepte)
Partida(idPartida, data, idXML, idJoc, idGestor, idJugador, idResultat)
Apostes(idApostes, idPartida, data, valor, rebutAposta)
Aposta(idAposta, idApostes, idPartida, valor, nom, numeros)
Pagaments(idApostes, idPartida, idGestor, idJugador, quantitat, data)

El camp saldo de la taula Jugador és un camp calculat de la base de dades. A partir dels registres que hi ha a la taula Moviments es pot calcular el saldo de cada jugador. Perquè la base de dades tingui aquesta informació actualitzada s'han creat dos procediments: actualitzar_saldo i inserir_moviment, i tres disparadors: t_insertdiposit, t_insertaposta, t_insertpagament.

Els disparadors s'executaran cada vegada que fem un INSERT a les taules que modifiquen el saldo del jugador: quan augmenta el dipòsit, quan fa una aposta i quan cobra una aposta. Cada vegada que salta un disparador s'executen els procediments d'actualitzar_saldo i inserir_moviment. El saldo no pot ser més petit que zero. Aquesta comprovació la fa l'aplicatiu del jugador igual que a la realitat són els jugadors els que es posen la mà a la butxaca per treure les seves fitxes i posar-les damunt de la taula.

7.3 Disseny UML

A la figura 7.2 hi ha el diagrama de classes ampliat per aquesta fase.

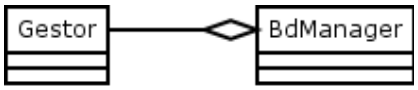

Figura 7.2: Diagrama de classes ampliat BD

7.4 Implementació

Per a la implementació d'aquesta fase hem de separar la part de base de dades de la part de l'aplicatiu.

7.4.1 Base de dades

Hem usat el gestor de base de dades MySqL perquè és de lliure distribució i està disponible en múltiples plataformes: Linux, Windows i Mac entre altres. La instal·lació del programari necessari i l'script que crea la base de dades ruleta buida està descrita a l'annex A, a l'apartat A.7.

La base de dades ha de suportar transaccions. En el moment d'aceptar una aposta, hem de tenir present que està formada per una o més apostes simples. L'acceptació de l'aposta s'ha de veure com una sola acció. Si una aposta simple no es pot acceptar, ja sigui perquè no és correcta o perquè es produeix un altre error, no s'acceptarà l'aposta completa.

Hem fet servir l'Engine Storage InnoDB perquè proporciona al MySQL un entorn transaccional amb les capacitats de commit, rollback, crash and recovery. L'InnoDB també permet fer restriccions sobre les claus foranes, a l'hora de crear-les, esborrar-les i actualitzar-les.

Hem d'emmagatzemar dades en format binari a la base de dades. Les dades que estan en aquest format són els certificats dels jugadors i del gestor i les signatures digitals, en concret el rebut de l'aposta. MySqL ens proporciona el tipus de dades BLOB (Binary Large Object) per a gestionar dades binàries grans. El tipus de dades BLOB genèric està dividit en quatre subtipus segons la seva capacitat d'emmagatzemament: tinyblob, blob, mediumblob i el largeblob. Hem escollit el blob perquè pot emmagatzemar des d' 1 byte fins a 65535 bytes.

Un altre tipus de dades que hem de tenir en compte és el temps. Hem triat el tipus de dades datetime que ens proporciona MySQL. Datetime és una combinació d'una data i d'una hora. El seu format és 'AAAA-MM-DD HH:MM:SS', i té un rang que va des de '1001-01-01 00:00:00' fins a '9999-12-31 23:59:59'.

Descripció dels camps i les taules de la base de dades

Taula joc

La taula joc emmagatzema els noms dels jocs que hi ha disponibles en la base de dades.

Nom camp	Tipus de dades	Descripció
idJoc	númeric (int)	número que identifica el joc
nomJoc	cadena de caràcters	nom del joc

Taules jugador i gestor

Les taules jugador i gestor emmagatzemen dades personals del jugador i del gestor. En aquestes dues taules a més de tenir l'índex la clau primària, hem creat un altre índex sobre el camp certificatHash. Aquest índex automàticament detectarà identificadors de jugadors repetit i ens permetrà buscar més ràpidament al jugador ja que gairebé totes les consultes sobre la taula jugadors des de l'aplicació són mitjançant el camp hashCertificat.

Nom camp	Tipus de dades	Descripció
idJugador	numèric (int)	nombre que identifica el jugador en la bd
dni	cadena de caràcters	DNI del jugador
nom	cadena de caràcters	nom del jugador
cognom	cadena de caràcters	cognom del jugador
dataAlta	temps (datetime)	data en què es va registrar per primera vegada
dataBaixa	temps (datetime)	data en què es donarà de baixa de l'aplicació
certificatHash	binari	identificador del jugador en l'aplicatiu
certificat	binari	el certificat en format X509V3
saldo	numèric (float)	saldo que té jugador

Nom camp	Tipus de dades	Descripció
idGestor	numèric (int)	nombre que identifica el gestor en la bd
dni	caràcters	DNI del gestor
nom	caràcters	nom del gestor
cognom	caràcters	cognom del gestor
dataAlta	temps (datetime)	data en què es va registrar per primera vegada
dataBaixa	temps (datetime)	data en què es donarà de baixa de l'aplicació
certificatHash	binari	identificador del gestor en l'aplicatiu
certificat	binari (blob)	el certificat en format X509V3

Taula resultat

La taula resultat emmagatzema el resultat, el número que ha sortit a la ruleta, de cada partida. També s'emmagatzema c_1 o c_2 per tornar-lo a calcular si cal.

Nom camp	Tipus de dades	Descripció
idResultat	numèric (int)	número que identifica el resultat
resultat	numèric (int)	el número que ha sortit a la ruleta
c1	binari (blob)	el número a què es compromet el jugador
c2	binari (blob)	el número a què es compromet el gestor
data	temps (datetime)	moment en el temps en què s'ha obtingut el resultat

Taula incrementdiposit

La taula incrementdiposit emmagatzema cada increment de dipòsit que fan els jugadors.

Nom camp	Tipus de dades	Descripció
idJugador	numèric (int)	jugador que fa l'increment
data	temps (datetime)	moment en què es fa l'increment
increment	numèric (float)	quantitat de l'increment
targetaCredit	cadena de caràcters	targeta de crèdit d'on s'ha fet l'increment

Taula moviments

La taula moviments emmagatzema els moviments de saldo dels jugadors.

Nom camp	Tipus de dades	Descripció
idMoviment	numèric (int)	número que identifica el moviment
idJugador	numèric (int)	jugador que fa el moviment
data	temps (datetime)	moment en què el jugador fa el moviment
valor	numèric (float)	valor del moviment
concepte	cadena de caràcters	el que genera el moviment: aposta,increment i pagament

Taula partida

La taula partida emmagatzema les dades de cada partida que es juga.

Nom camp	Tipus de dades	Descripció
idPartida	numèric (int)	identificador de partida, és el número de la partida
data	temps (datetime)	moment de creació de la partida
idXML	cadena de caràcters	identificador XML de la partida
idJoc	numèric (int)	número que identifica el joc
idGestor	numèric (int)	gestor que ha creat la partida
idJugador	numèric (int)	jugador que juga a la partida
idResultat	numèric (int)	identificador del resultat de la partida

Taula apostes i aposta

Les taules apostes i aposta emmagatzemen les apostes d'un jugador per a cada partida.

Nom Camp	Tipus de dades	Descripció
idApostes	numèric (int)	número que identifica una aposta
idPartida	numèric (int)	partida on s'ha fet l'aposta
data	temps (datetime)	moment en què s'ha fet l'aposta
valor	numèric (float)	valor total de l'aposta
rebutAposta	binari (blob)	signatura digital del gestor sobre l'aposta del jugador

Nom camp	Tipus de dades	Descripció
idAposta	numèric (int)	número que identifica l'aposta simple
idApostes	numèric (int)	identificador de l'aposta
idPartida	numèric (int)	número de la partida on s'ha fer l'aposta
valor	numèric (float)	valor d'aquesta aposta
nom	cadena de caràcters	nom de l'aposta: ple, cavall, etc.
numeros	cadena de caràcters	números als quals hem apostat separats per comes

Taula pagaments

La taula pagaments emmagatzema els pagaments que fan els gestors als jugadors cada vegada que els jugadors guanyen una aposta.

Nom Camp	Tipus de dades	Descripció
idApostes	numèric(int)	número que identifica l'aposta que s'ha pagat
idPartida	numèric(int)	número de partida en què s'ha fet el pagament
idGestor	numèric(int)	número que identifica el gestor que ha pagat l'aposta
idJugador	numèric(int)	número que identifica el jugador a qui s'ha pagat l'aposta
quantitat	numèric(float)	quantitat del pagament
data	temps(datetime)	moment en què s'ha fer el pagament

7.4.2 Java

No hem necessitat instal·lar de cap programari addicional, hem usat JDBC proporcionat per l'API estàndard de Java.

L'accés a la base de dades es fa mitjançant una classe específica a la qual només hi té accés el gestor. Hem creat un paquet nou **bd**, (vegeu el diagrama de paquets a la figura 8.2), amb una sola classe: **BdManager**. Aquesta classe té mètodes per fer la connexió a la base de dades, per afegir-hi informació i per obtenir-ne.

Mètodes per afegir informació:

afegirJugador(..), afegirGestor(..): afegeix a la base de dades un jugador o un gestor amb les dades corresponents.

afegirIncrement(...): actualitza la taula incrementdiposit.

afegirPartida(...): quan el gestor crea una nova partida, l'afegeix a la taula partides.

afegirAposta(...): quan el gestor accepta una aposta d'un jugador per una partida la registra a la taula apostes.

afegirResultat(...): quan el gestor i el jugador obtenen el resultat de la partida es registra el resultat.

afegirPagament(...): quan el gestor paga una aposta a un jugador es registra a la taula pagaments.

Mètodes per obtenir informació:

obtenirSaldoJugador(byte[] hashJ) el gestor, a partir de l'identificador del jugador, obté el saldo.

obtenirCertificatJugador(byte[] hashJ): el gestor, a partir de l'identificador del jugador, obté el certificat.

obtenirRebutAposta(String idPartida): el gestor, quan ha de pagar una aposta al jugador, recupera el rebut de l'aposta per comprovar que el jugador vol cobrar una aposta correcta.

Capítol 8

Interfície

La tasca principal d'una interfície és fer de mitjancer entre dos sistemes, que normalment són de naturalesa diferent i no es poden entendre. Com ho farà l'usuari per dir-li a l'aplicació que vol jugar? I com, que vol apostar?. Hem d'afegir a la nostra aplicació una interfície que permeti als usuaris interactuar amb ella de manera senzilla i còmoda.

Es defineix la interfície gràfica d'usuari com el conjunt de components o objectes usats pels usuaris per comunicar-se amb l'aplicació. L'usuari dirigeix el funcionament de l'aplicació a través d'instruccions. Aquestes instruccions estan sota el paradigma d'interacció **objecte - acció**: des del punt de vista de l'usuari fer clic a un botó vol dir executar l'acció que el botó té associada, i des del punt de vista de l'aplicació és executar l'acció.

Per al disseny de la interfície s'ha seguit el patró de MVC (Model Vista Controlador), ja que qualsevol modificació a la interfície, és totalment independent a la resta de components de l'aplicació.

8.1 Disseny UML

A la figura 8.1 hi ha el diagrama de classes ampliat per aquesta fase. La classe GUIJugador és la classe que farà d'interfície entre els jugadors i l'aplicació.

8.2 Implementació

Hem afegit un paquet nou, amb una sola classe: GUIJugador. Vegeu el diagrama de paquets complerts a la figura 8.2. La classe GUIJugador conté

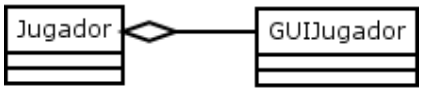

Figura 8.1: Diagrama de classes ampliat GUI

els mètodes per crear la interfície, mostrar els seus components i gestionar els canvis que es produeixen quan l'usuari interactua amb ella. Hem usat les classes que ens proporciona l'API de Java, l'AWT (Abstract Windows Toolkit) i les JFC/Swing.

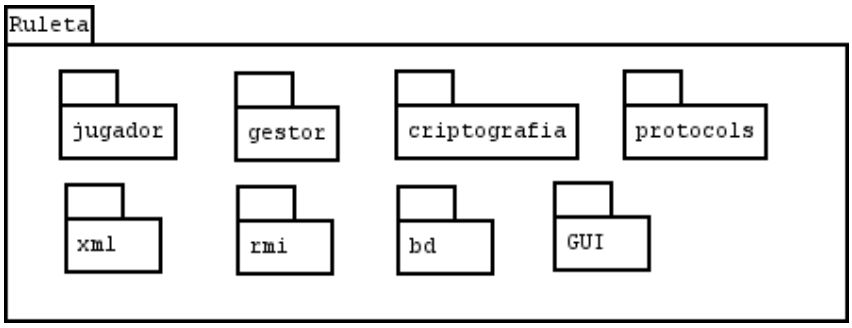

Figura 8.2: Diagrama de paquets

8.3 Descripció dels components de la interfície

8.3.1 Finestra principal

La finestra principal es pot dividir en dues parts, vegeu la figura 8.3. Una part representa la taula del joc i l'altra part representa el jugador. La part de la taula de joc conté el dibuix de la taula i un botó per fer girar la ruleta. Aquest botó és la mateixa ruleta. L'altra part té dues files: la primera, està formada per etiquetes que mostraran informació, i l'altra fila conté botons. Cada botó té una acció associada, per exemple el botó d'ajuda serveix per demanar ajuda.

La taula s'usa per fer apostes. Hem de fer click amb el ratolí al lloc concret on volem posar una fitxa.
La ruleta és un botó. En fer-li clic obtindrà el resultat de la partida.

Missatges és una etiqueta que ens mostrarà informació sobre el que passa a la ruleta o sobre les accions que podem fer. Quan passegem el ratolí per damunt de la taula, l'etiqueta missatges ens informarà de quina aposta faríem si féssim clic per on està passant el ratolí.

Figura 8.3: Finestra d'autenticació

Crèdit és una etiqueta que mostra els diners que ens queden en dipòsit al casino.
Total Aposta és una etiqueta que mostra el total de diners que hem apostat, fins al moment, en aquesta partida. Cada vegada que es comença una nova partida comença altre cop des de zero.
Resultat és una etiqueta que mostra el resultat obtingut en la partida.
Guanys és una etiqueta que mostra el guanys obtinguts de les apostes en la partida.

Les fitxes les hem representat mitjançant botons. Cadascun representa el valor que té la fitxa. Abans d'apostar s'ha de seleccionar la fitxa que volem apostar.

Increment de dipòsit és un botó que ens permetrà incrementar els diners que tenim al dipòsit del casino. Vegeu la figura 8.7.
Ajuda és un botó que mostra la finestra d'ajuda, en ella s'explica breument com s'aposta a la ruleta i quins premis es paguen. Vegeu la figura 8.8.
Sortir és un botó que ens permetrà sortir de l'aplicació.

8.3.2 Finestra de benvinguda

La finestra de benvinguda consta de dos botons: Jugar i Registrar. Vegeu la figura 8.4.

Jugar visualitzarà a la finestra d'autenticació que ens permetrà autenticar-nos a l'aplicació i, si no hi ha cap error, podrem accedir a la finestra principal on podrem començar a jugar.
Registre visualitzarà la finestra de registre que ens permetrà registrar-nos en l'aplicació i, si no hi ha cap error en el procés de registre, ens tornarà a la finestra de benvinguda. El jugador ja estarà registrat a la base de dades.

Figura 8.4: Finestra de benvinguda

8.3.3 Finestra de registre

La finestra de registre consta d'una part que serveix perquè el jugador introdueixi les dades i d'una altra part on hi ha els botons Enviar dades i Cancel·lar. Vegeu la figura 8.5.

Nom Camp per introduir el nom del jugador. El nom no pot ser més llarg de 20 caràcters.

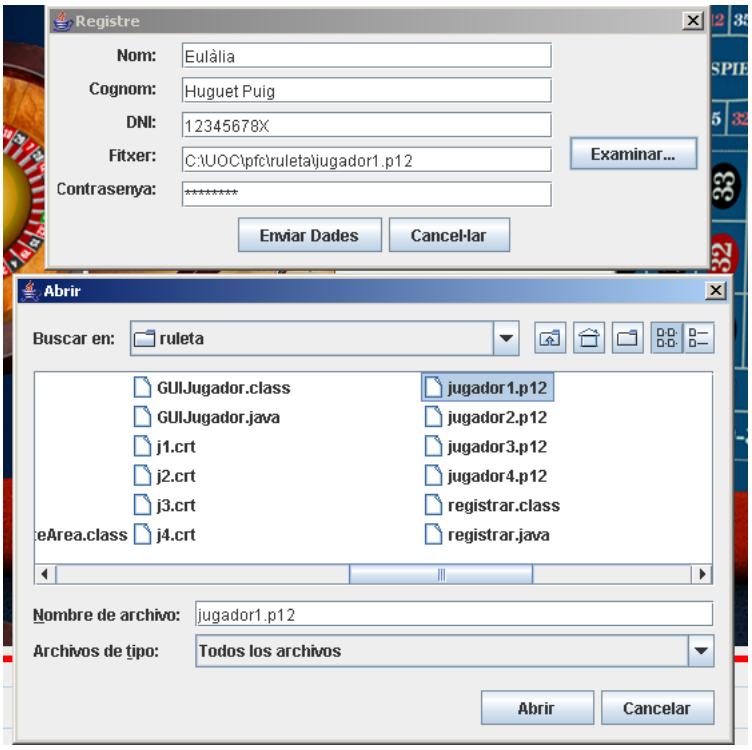

Figura 8.5: Finestra de registre

Cognom Camp per introduir el cognom del jugador. El cognom no pot ser més llarg de 20 caràcters.
DNI Camp per introduir el DNI del jugador. El DNI ha de ser de la forma 12345678X, els vuit nombres que el formen més la lletra. Ha de tenir 9 caràcters.

Fitxer Camp per introduir el camí on es troba el fitxer que conté el parell de claus i el certificat digital del jugador. Hem posat el botó Examinar... per poder buscar el fitxer si no recordem exactament el camí.
Contrasenya Camp per introduir la contrasenya d'exportació del fitxer que conté el parell de claus i el certificat digital del jugador.

Enviar Dades Un cop el jugador ha introduït les dades que li demanen, l'aplicació envia les dades al gestor. Primer es comprova que siguin correctes. Si no ho són, romandrem en aquesta finestra. Quan totes les dades siguin correctes i no es produeixi cap error de comunicació o d'altres d'inesperats el gestor registrarà a l'usuari en l'aplicació i l'aplicació visualitzarà la finestra de benvinguda perquè pugui jugar.
Cancel·lar Si el jugador no està segur de voler registrar-se pot cancel·lar el procés i tornar a la finestra de benvinguda.

8.3.4 Finestra d'autenticació

La finestra d'autenticació consta d'una part d'introducció de dades i d'una altra part d'accions. Vegeu la figura 8.6.

Fitxer Camp per introduir el camí on es troba el fitxer que conté el parell de claus i el certificat digital del jugador. Hem posat el botó Examinar... per poder buscar el fitxer si no recordem exactament el camí.
Contrasenya Camp per introduir la contrasenya d'exportació del fitxer que conté el parell de claus i el certificat digital del jugador.

Validar Un cop el jugador ha introduït les dades que li demanen, l'aplicació envia les dades al gestor. Primer es comprova que siguin correctes. Si no ho són, romandrem en aquesta finestra. Quan totes les dades siguin correctes i no es produeixi cap error de comunicació o d'altres d'inesperats s'executarà el protocol d'autenticació entre el gestor i el jugador. Si s'acaba correctament l'aplicació visualitzarà la finestra principal on el jugador podrà començar a jugar.
Cancel·lar Si el jugador no està segur de voler autenticar-se a l'aplicació en aquest moment, pot cancel·lar el procés i tornar a la finestra de benvinguda.

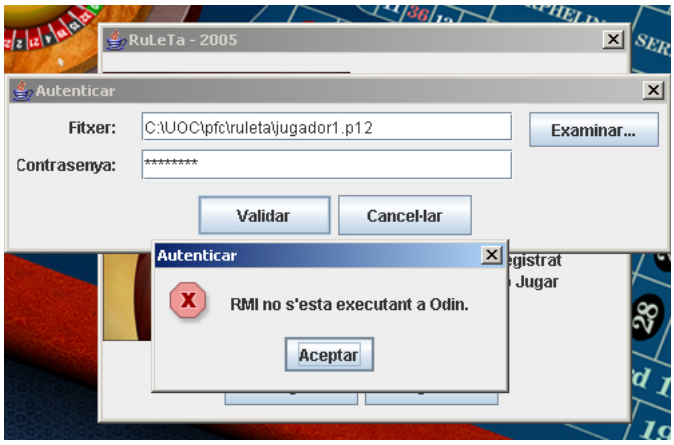

Figura 8.6: Finestra d'autenticació

8.3.5 Finestra d'increment de dipòsit

La finestra d'increment de dipòsit consta d'una part d'introducció de dades i d'una altra part d'accions. Vegeu la figura 8.6.

Increment Camp per introduir la quantitat de diners amb la que el jugador vol augmentar el seu dipòsit per jugar al casino.
Targeta de crèdit Camp per introduir el números de la targeta de crèdit del jugador. Ha de tenir 16 números.

Enviar dades Només es permès d'augmentar el dipòsit abans d'iniciar una partida o després d'acabar-la. Un cop s'ha fet una aposta a la taula no es podrà augmentar el dipòsit.
Cancel·lar Si el jugador no està segur de voler augmentar el dipòsit en aquest moment, pot cancel·lar el procés i tornar a la finestra principal .

Figura 8.7: Finestra increment de dipòsit

8.3.6 Finestra d'ajuda

Mostra informació sobre les accions que podem fer amb l'aplicació i com s'aposta i quins premis es donen. Vegeu la figura 8.8

Figura 8.8: Finestra d'ajuda

Conclusions

Hem definit un esquema criptogràfic que compleix les propietats de seguretat següents sobre les apostes:
Autenticitat: per a qualsevol aposta podem demostrar quin jugador l'ha feta.
Integritat: les apostes no les poden manipular cap de les dues parts un cop han estat acceptades.
No-repudi: un jugador no pot negar haver fet una aposta i un gestor no pot negar haver acceptat una aposta.

El compliment d'aquestes propietats ens permet detectar jugadors i gestors deshonestos, que intentin canviar les apostes o negar que les han fetes, l'aplicació ho detectarà.

L'esquema criptogràfic també ens permet garantir que els esdeveniments s'obtenen de manera honesta. Hem decidit no usar una TTP per obtenir el resultat de la ruleta, ja que si forma part del joc en compromet la seguretat. Hem optat perquè el jugador i el gestor obtinguin el resultat conjuntament.

Hem dissenyat i implementat el sistema de comunicacions perquè el joc electrònic sigui remot. També hem donar l'esquema de document (ruleta.dtd) que han de complir els missatges que s'intercanviaran els jugadors i els gestors.

Hem posat en marxa la base de dades ruleta, que ens permet emmagatzemar les dades que es van produint en usar l'aplicació. La base de dades ens permet fer una auditoria del que ha passat en una partida i amb programes especialitzats extreure informació del comportament dels jugadors.

Implementació d'una interfície pel jugador perquè la interacció entre els usuaris i l'aplicació sigui senzilla.

Bibliografia

[1] Iaik java crypto toolkit homepage, 2005. [Internet; en línia 31-desembre-2005].

[2] The internet engineering task force request-for-documents, 2005. [Internet; en línia 31-desembre-2005].

[3] The jdom xml, 2005. [Internet; en línia 31-desembre-2005].

[4] Mysql, 2005. [Internet; en línia 31-desembre-2005].

[5] Openssl home page, 2005. [Internet; en línia 31-desembre-2005].

[6] W3schools, 2005. [Internet; en línia 31-diciembre-2005].

[7] Inc. Sun Microsystems. The java tutorial. a practical guide for programmers, 2005. [Internet; en línia 31-desembre-2005]
Tutorials consultats: Essential Java Classes, Creating a GUI with JFC/Swing, JDBC Database Access, RMI, Putting It All Together, Drag and Drop.

[8] Inc. Sun Microsystems. Javatm 2 platform standard edition 5.0 api specification, 2005. [Internet; en línia 31-desembre-2005].

[9] Wikipedia. Portada — wikipedia, l'enclicopèdia lliure, 2005. [Internet; en linia 31-diciembre-2005].

Apèndix A

Eines utilitzades

A.1 Generació de la PKI

Hem construït una pepita PKI mitjançant l'eina openssl que ja ve instal·lada en la versió de Linux Fedora Core 4:

```
[eulalia@thor bin]$ openssl
OpenSSL> version
OpenSSL 0.9.7f 22 Mar 2005
OpenSSL> quit
[eulalia@thor bin]$
```

Generem el parell de claus de l'autoritat de certificació (CA) amb una llargada de 2048 bits que anomenarem CA.key. La contrasenya de la clau privada de la CA és: **ehuguet2005**

```
$ openssl genrsa -des3  -rand aleatori -out CA.key 2048
```

Detall de l'execució:

```
[eulalia@thor PKI]$ ./generarClaus  CA.key 2048
Generating RSA private key, 2048 bit long modulus
.............+++
.....................+++
e is 65537 (0x10001)
Enter pass phrase for CA.key:
Verifying - Enter pass phrase for CA.key:
[eulalia@thor PKI]$
```

Generem un certificat autosignat amb el parell de claus obtingudes. Anomenarem CA.crt l'arxiu del certificat.

```
$ openssl req -new -sha1 -x509 -key CA.key -out CA.crt -days 365
```

Detall de l'execució:

```
[eulalia@thor PKI]$ ./generaCertificatAutosignat CA.key CA.crt 365
Enter pass phrase for CA.key:
You are about to be asked to enter information that will be
incorporated into your certificate request.
What you are about to enter is what is called a Distinguished
Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:ES
State or Province Name (full name) [Berkshire]:Espanya
Locality Name (eg, city) [Newbury]:Lleida
Organization Name (eg, company) [My Company Ltd]:UOC
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:Thor
Email Address []:ehuguet@uoc.edu
[eulalia@thor PKI]$
```

Copiem el certificat de la CA al directori de la CA i movem la clau privada de la CA al directori private.

```
[eulalia@thor PKI]$ cp CA.crt CAPFC/
[eulalia@thor PKI]$ mv CA.key CAPFC/private/
```

Un cop ja tenim el certificat que ens permet signar certificats a altres entitats que ens ho demanin, generem un parell de claus per al jugador1 i un altre parell per al gestor del joc amb noms j1.key i gestorjoc.key respectivament i totes dues de 1024 bits de llargada.
La contrasenya de j1.key és: **jugador1**.
Anàlogament la contrasenya de gestorjoc.key és: **gestorjoc**.

```
$ openssl genrsa -des3  -out j1.key 1024
$ openssl genrsa -des3  -out gestorjoc.key 1024
```

Un cop tenim les claus generem una petició de certificat per a cadascú:

```
openssl req -new -sha1 -config openssl.cnf -key j1.key -out j1.csr
openssl req -new -sha1 -config openssl.cnf -key gestorjoc.key
-out gestorjoc.csr
```

Detall de l'execució:

```
[eulalia@thor PKI]$./generaPeticioCertificat j1.key j1.csr openssl.cnf
Enter pass phrase for j1.key:
You are about to be asked to enter information that will be
incorporated into your certificate request.
What you are about to enter is what is called a Distinguished
Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [ES]:ES
State or Province Name (full name) [Catalunya]:Catalunya
Locality Name (eg, city) [Barcelona]:Lleida
Organization Name (eg, company) [Universitat Oberta de Catalunya]:UOC
Organizational Unit Name (eg, section) [Consultors]:Alumnes
Common Name (eg, YOUR name) []:Jugador1
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:j12005
An optional company name []:
[eulalia@thor PKI]$
```

Amb la petició de certificat (.csr) li demanem a la CA que ens generi el certificat del jugador i del gestor del joc.

```
openssl ca -config openssl.cnf -out j1.crt -infiles j1.csr
openssl ca -config openssl.cnf -out gestorjoc.crt -infiles gestorjoc.csr
```

Detall de l'execució:

```
[eulalia@thor PKI]$ ./generaCertificat j1.csr j1.crt openssl.cnf
Using configuration from openssl.cnf
Enter pass phrase for ./CAPFC/private/CA.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'ES'
stateOrProvinceName   :PRINTABLE:'Catalunya'
localityName          :PRINTABLE:'Lleida'
```

```
organizationName      :PRINTABLE:'UOC'
organizationalUnitName:PRINTABLE:'Alumnes'
commonName            :PRINTABLE:'Jugador1'
Certificate is to be certified until Dec 27 05:46:11 2006 GMT
(365 days)
Sign the certificate? [y/n]:y
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
[eulalia@thor PKI]$
```

Tot seguit generem l'arxiu PKCS12 que contindrà el parell de claus del jugador, el certificat del jugador i el certificat de la CA. També generarem l'arxiu PKCS12 per al gestor del joc. Les contrasenyes d'exportació per a cadascun d'ells són: **exportj1** i **exportgestor**

```
openssl pkcs12 -export -in j1.crt -inkey j1.key
       -certfile CA.crt -out jugador1.p12

openssl pkcs12 -export -in gestorjoc.crt -inkey gestorjoc.key
       -certfile CA.crt -out gestorjoc.p12
```

Detall de l'execució:

```
[eulalia@thor PKI]$ ./generaPKCS12 j1.key j1.crt CA.crt jugador1.p12
Enter pass phrase for j1.key:
Enter Export Password:
Verifying - Enter Export Password:
[eulalia@thor PKI]$
```

A.2 Instal·lació del JDK 1.5.0

Linux: Descarreguem de http://java.sun.com l'arxiu:
`jdk-1_5_0_05-linux-i586-rpm.bin`. Només cal executar-lo. La versió de Linux, Fedora Core 4, porta per defecte altres versions de Java per problemes de llicències als Estats Units. Un cop instal·lat el rpm, s'han de modificar els enllaços simbòlics del sistema pels que acabem d'instal·lar. El directori on habitualment estan els executables és el /usr/bin.

```
$ cd /usr/bin/
$ ls -l jav*
java -> /usr/java/jdk1.5.0_05/bin/java
java.old -> /etc/alternatives/java
javac -> /usr/java/jdk1.5.0_05/bin/javac
javac.old -> /etc/alternatives/javac
javadoc -> /usr/java/jdk1.5.0_05/bin/javadoc
javadoc.old -> /etc/alternatives/javadoc
javah -> /usr/java/jdk1.5.0_05/bin/javah
javah.old -> /etc/alternatives/javah
```

Windows: Descarreguem de http://java.sum.com l'arxiu:
`jdk-1_5_0_03-windows-i586.exe` i l'executeu. Poseu a la variable PATH del sistema el directori dels executables i a la variable CLASSPATH els directoris on estaran les classes.

A.3 Instal·lació de la llibreria IAIK

Cal registrar-se per poder descarregar aquesta llibreria. És gratuita per a finalitats educatives. Un cop hem descarregat l'arxiu: `iaik_jce_full.jar` l'hem de copiar al directori:
Linux: `$JAVA_HOME/jre/lib/ext`
Windows: `c:\Archivos de programa\Java\jdk1.5.0\jre\lib\ext`
`c:\Archivos de programa\Java\jre1.5.0\lib\ext`

També hem de descarregar l'arxiu: `jce_policy-1.5.0.zip`. L'arxiu conté les polítiques de seguretat de Java que permeten emprar qualsevol longitud de clau, Java Cryptography Extensions (JCE) Unlimited Strength Jurisdiction Policy Files 5.0 RC. El descomprimim i copiem els arxius: `local_policy.jar` i `US_export_policy.jar` a:
Linux: `$JAVA_HOME/jre/lib/security`.
Windows: `c:\Archivos de programa\Java\jdk1.5.0\jre\lib\security`
`c:\Archivos de programa\Java\jre1.5.0\lib\security`

A.4 Instal·lació del Jdom i del Xalan

Descarreguem l'arxiu `jdom-contrib-1.0.zip` de l'adreça http://jdom.org/ i el descomprimim. Al directori `built` hi ha el paquet jdom.jar que hem d'afegir al CLASSPATH.

Descarreguem l'arxiu de l'adreça http://xml.apache.org/xalan-j i el descomprimim. Al directori arrel hi ha el paquet xalan.jar que hem d'afegir al CLASSPATH.

A.5 XML

A.5.1 Exemples de documents XML

Us presentem alguns exemples de parts dels documents xml, deixant de banda els camps de les signatures que, per la seva codificació, resulten poc interessants.

```
<IdPartida>
  <J>RuLeTa</J>
  <r>sijq2Z1B5AAUNx+IJxCsuelEs9s=</r>
  <T>2006-01-01 08:50:26</T>
  <N>173</N>
</IdPartida>
```

```
<IdAposta>
  <IdPartida>
    <J>RuLeTa</J>
    <r>ArGAJvrHSTYMVGMCHSh4R9o96wQ=</r>
    <T>2005-12-31 20:23:48</T>
    <N>154</N>
  </IdPartida>
  <r>tWJ9G2x0pexdkkQQGPGlz5KaSfk=</r>
  <T>2005-12-31 20:23:56</T>
  <V>9.0</V>
  <C>
    <Nom>parells</Nom>
    <Valor>2.0</Valor>
    <Numero> </Numero>
  </C>
  <C>
    <Nom>ple</Nom>
    <Valor>2.0</Valor>
    <Numero>30</Numero>
  </C>
  <C>
    <Nom>cavall</Nom>
    <Valor>5.0</Valor>
    <Numero>1</Numero>
    <Numero>2</Numero>
  </C>
</IdAposta>
```

Un dels missatges del protocol autenticar:

```
<?xml version="1.0" encoding="UTF-8"?>
<Ruleta>
  <Contenidor>
    <Element1>HXIH3SNcyZXwOVNvP6B6SCHa2P8=</Element1>
    <Element2>1UkD0gdyKSRQnELHA2b/x8+3VK4=</Element2>
    <Element3>b31mvom1cqDD0MDFcScouXBxQtM=</Element3>
  </Contenidor>
</Ruleta>
```

A.5.2 Esquema de definició de document - DTD

L'esquema de definició de document és el següent:

```
<?xml version='1.0' encoding='UTF-8'?>

<!--- Ruleta pot estar format pels elements següents: -->
<!ELEMENT Ruleta (Registre|ContenidorXML|Partida|
                  IncrementDiposit|Aposta|Compromis)>

<!--- Registre -->
<!ELEMENT Registre (Certificat)>

<!--- Protocol autenticar -->
<!ELEMENT ContenidorXML (Element1,Element2,Element3)>

<!--- Protocol Iniciar Partida -->
<!ELEMENT Partida((Certificat,Joc)|(Signatura,IdPartida))>
<!ELEMENT IdPartida (J,r,T,N)>

<!--- Protocol Incrementar diposit -->
<!ELEMENT IncrementDiposit((Signatura,IdDiposit)|
                           (RebutCredit,Diposit))>
<!ELEMENT IdDiposit (r,T,V,B)>

<!--- Protocol Fer aposta -->
<!-- el símbol + : una o mes vegades -->
<!ELEMENT Aposta(idAposta|(RebutCredit,RebutAposta,Diposit))>
<!ELEMENT IdAposta (IdPartida, r,T,V,C+)>
<!ELEMENT C (Valor,Nom,Numero+) >
```

```
<!--- Protocol Cobrar/pagar una aposta -->
<!-- ja estan definits ContenidorXML, IncrementDiposit -->

<!--- Protocol Compromis i obertura -->
<!ELEMENT Compromis (Signatura,Valor)>

<!-- Elements  -->
<!ELEMENT Certificat  (#PCDATA)>
<!ELEMENT Element1    (#PCDATA)>
<!ELEMENT Element2    (#PCDATA)>
<!ELEMENT Element3    (#PCDATA)>
<!ELEMENT Joc         (#PCDATA)>
<!ELEMENT Signatura   (#PCDATA)>
<!ELEMENT J           (#PCDATA)>
<!ELEMENT r           (#PCDATA)>
<!ELEMENT T           (#PCDATA)>
<!ELEMENT N           (#PCDATA)>
<!ELEMENT V           (#PCDATA)>
<!ELEMENT B           (#PCDATA)>
<!ELEMENT RebutCredit (#PCDATA)>
<!ELEMENT Diposit     (#PCDATA)>
<!ELEMENT Valor       (#PCDATA)>
<!ELEMENT Nom         (#PCDATA)>
<!ELEMENT Numero      (#PCDATA)>
<!ELEMENT RebutAposta (#PCDATA)>
```

A.6 RMI: Seqüència d'arrencada

Hem de generar els Stub i l'Skeleton amb l'ordre: `rmic ruleta.gestor.Gestor`.

Després hem de posar en marxa el registre:
Linux: `rmiregistry &`.
Windows: `start rmiregistry`.

I per finalitzar hem d'executar el servidor: java ruleta.rmi.Servidor.

A.7 Instal·lació del MySqL i càrrega de l'script de la base de dades

Linux: Descarreguem l'arxiu `mysql-connector-java-3.1.12.tar.gz` de l'adreça http://www.mysql.com a l'apartat Downloads->MySqL Connectors i escolliu ConnectorJ, el connector de Java. El descomprimim i obteniu el paquet `mysql-connector-java-3.1.12-bin.jar` que hem de posar a la variable de sistema CLASSPATH.

Windows: Descarreguem l'arxiu `mysql-connector-java-3.0.17-ga.zip` de l'adreça http://www.mysql.com a l'apartat Downloads->MySqL Connectors i escolliu ConnectorJ, el connector de Java. Ho descomprimiu El descomprimim i obteniu el paquet `mysql-connector-java-3.1.12-bin.jar` que hem de posar a la variable de sistema CLASSPATH.

Script que genera la base de dades ruleta buida:

```
-- Creacio de la base de dades buida
--create database if not exists ruleta;
--use ruleta;
drop database if exists ruleta;
create database if not exists ruleta;
use ruleta;

-- Creació de les taules
-- Taula JOC
create table joc(
  idJoc int unsigned not null auto_increment,
  nomJoc varchar(10) not null,
  primary key (idJoc)
) engine=InnoDB;
```

```
-- Taula JUGADOR
create table jugador(
  idJugador int unsigned not null auto_increment,
  dni varchar(9) not null,
  nom varchar(20) not null,
  cognom varchar(20) not null,
  dataAlta datetime not null,
  dataBaixa datetime default null,
  certificatHash varbinary(20) not null unique,
  certificat blob not null,
  saldo float not null,
  index (certificatHash),
  primary key ( idJugador )
) engine=InnoDB;

-- Taula GESTOR
create table gestor(
  idGestor int unsigned not null auto_increment,
  dni varchar(9) not null,
  nom varchar(20) not null,
  cognom varchar(20) not null,
  dataAlta datetime not null,
  dataBaixa datetime default null,
  certificatHash varbinary(20) not null unique,
  certificat blob not null,
  index(certificatHash),
  primary key ( idGestor )
) engine=InnoDB;

-- Taula RESULTAT
create table resultat(
  idResultat int unsigned not null auto_increment,
  resultat int unsigned not null,
  c1 blob not null,
  c2 blob not null,
  data datetime not null,
  primary key (idResultat)
) engine=InnoDB;
```

```
-- Taula INCREMENT DIPOSIT
create table incrementdiposit(
  idJugador int unsigned,
  data datetime not null,
  increment float not null,
  targetaCredit varchar(16) not null,
  primary key (data, idJugador),
  foreign key (idJugador) references jugador(idJugador)
) engine=InnoDB;

-- Taula MOVIMENTS DE SALDO
create table moviments(
  idMoviment int unsigned not null auto_increment,
  idJugador int unsigned,
  data datetime not null,
  valor float not null,
  concepte varchar(10),
  primary key (idMoviment),
  foreign key (idJugador) references jugador(idJugador)
) engine=InnoDB;

-- Taula PARTIDA
-- quan es crea la partida encara no sabem el resultat
create table partida(
  idPartida int unsigned not null auto_increment,
  data datetime not null,
  idXML varchar(200),
  idJoc int unsigned,
  idGestor int unsigned,
  idJugador int unsigned,
  idResultat int unsigned,
  primary key (idPartida),
  foreign key (idJoc) references joc(idJoc),
  foreign key (idResultat) references resultat(idResultat),
  foreign key (idGestor) references gestor(idGestor),
  foreign key (idJugador) references jugador(idJugador)
) engine=InnoDB;
```

```
-- Taula APOSTES
-- signatura amb la clau del gestor de l'aposta que ha fet el jugador
-- ha ser null perque primer s'ha de crear l'aposta abans de poder
-- obtenir el rebut de l'aposta
create table apostes(
  idApostes int unsigned not null auto_increment,
  idPartida int unsigned,
  data datetime not null,
  valor float not null,
  rebutAposta blob,
  foreign key (idPartida) references partida(idPartida),
  primary key (idApostes, idPartida)
) engine=InnoDB;

-- Taula APOSTA
create table aposta(
  idAposta int unsigned not null auto_increment,
  idApostes int unsigned,
  idPartida int unsigned,
  valor float not null,
  nom varchar(15) not null,
  numeros varchar(25) not null,
  foreign key (idApostes) references apostes(idApostes),
  foreign key (idPartida) references partida(idPartida),
  primary key (idAposta, idApostes, idPartida)
) engine=InnoDB;

-- Taula PAGAMENTS
create table pagaments(
  idApostes int unsigned,
  idPartida int unsigned,
  idGestor int unsigned,
  idJugador int unsigned,
  quantitat float not null,
  data datetime not null,
  foreign key (idApostes) references apostes(idApostes),
  foreign key (idPartida)  references partida(idPartida),
  foreign key (idGestor)  references gestor(idGestor),
  foreign key (idJugador) references jugador(idJugador),
  primary key (idApostes, idPartida)
) engine=InnoDB;
```

```
-- Procediment per actualitzar el saldo del jugador
delimiter //
create procedure actualitzar_saldo(in i int, in s float)
  begin
    update jugador set saldo = saldo + s where idJugador = i;
  end
//

create procedure inserir_moviment(in ij int, in v float, in c varchar(10))
  begin
    insert into moviments values(null, ij, now(), v, c);
  end
//

-- Triggers que avisaran per actualitzar el saldo del jugador
create trigger t_insertdiposit before INSERT ON incrementdiposit
  for each row
  begin
    call actualitzar_saldo(new.idJugador, new.increment);
    call inserir_moviment(new.idJugador, new.increment, 'increment');
  end
//

create trigger t_insertaposta before INSERT ON apostes
  for each row
  begin
    declare i int;
    select idJugador into i from partida where
new.idPartida = partida.idPartida;
    call actualitzar_saldo(i, (new.valor)*(-1));
    call inserir_moviment(i,(new.valor)*(-1), 'aposta');
  end
//
```

```
create trigger t_insertpagament before INSERT ON pagaments
  for each row
  begin
    call actualitzar_saldo(new.idJugador, new.quantitat);
    call inserir_moviment(new.idJugador, new.quantitat, 'pagament');
  end
//

delimiter ;

-- Posarem algun valor inicial a les taules
insert into joc(nomJoc) values('RuLeTa');
insert into joc(nomJoc) values('BiNGo');
insert into joc(nomJoc) values('DauS');
```

Apèndix B

Fem una partida?

Suposem que teniu instal·lat tot el programari necessari descrit a l'annex A.

Heu de compilar el projecte **Linux**: Executar el fitxer `compilacio.sh`
Windows: Executar el fitxer `compilacio.bat`

Heu de posar en marxa el gestor de base de dades: MySQL.
Linux: Comproveu l'estat del servei i si no està funcionant activeu-lo, com a superusuari:

```
[root@thor ~]# /etc/init.d/mysqld status
mysqld está parado
[root@thor ~]# /etc/init.d/mysqld start
Iniciando MySQL:                                           [  OK  ]
[root@thor ~]#
```

Carregueu l'scrip de la base de dades. Creeu un usuari ehuguet amb password uoc2005 i doneu-li permisos per accedir a la base de dades que acabeu de crear.

```
[eulalia@thor ruleta]$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 18 to server version: 4.1.11

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> source ruleta.sql
```

```
mysql> grant all privileges on ruleta.* to 'ehuguet'@'localhost'
-> identified by 'uoc2005' with grant option;
Query OK, 0 rows affected (0.00 sec)
mysql>

mysql>
mysql> quit
Bye
[eulalia@thor ruleta]$
```

Windows: La instal·lació del MySQL crea un servei que s'arranca cada cop que arranquem el Windows XP. Comproveu que s'està executant el procés: mysqld-nt a l'administrador de tasque, a la pestanya processos. Carregeu l'scrip de la base de dades i creeu un usuari ehuguet amb password uoc2005. Doneu-li permisos per accedir a la base de dades que acabeu de crear.

```
C:\>mysql -u root -p
Enter password: ******
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 30 to server version: 5.0.16-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> source ruleta.sql

mysql> create user ehuguet identified by 'uoc2005';
Query OK, 0 rows affected (0.05 sec)
mysql>

mysql> grant all privileges on ruleta.* to ehuguet;
Query OK, 0 rows affected (0.00 sec)
mysql>
mysql> quit
Bye
C:\>
```

Heu d'arrancar el servidor RMI:
Linux: Executar el fitxer `servidor.sh`
Windows: Executar el fitxer `servidor.bat`

Heu de registrar els usuaris, es registren el jugador1, jugador2, jugador3 i el gestor. Es deixa el jugador4 per si voleu registrar-lo: `java registrar`

Heu d'executar l'aplicació que farà funcionar la ruleta:
`java ruleta.gui.GUIJugador`.

Es visualitzarà la finestra de benvinguda. Feu clic a jugar i valideu-vos, amb alguns dels jugadors registrats.

Figura B.1: Figura

Com que el vostre saldo és zero, haureu d'augmentar el dipòsit. Poseu la quantitat que desitgeu i un número de targeta de credit, pe. 1234 1234 1234 1234.

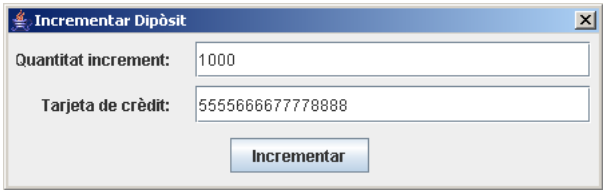

Figura B.2: Finestra de benvinguda

Feu clic a la fitxa del valor que vulgueu apostar. Després feu clic en el lloc determinat on vulgueu posar la fitxa. Repetiu el procés tantes vegades com apostes vulgueu fer. Si sempre aposteu fitxes del mateix valor no cal que feu

clic a les fitxes, només en el cas que vulgueu apostar diferents quantitats de diners cada vegada.

Feu clic a la ruleta per obtenir un resultat i cobrar les apostes en el cas que hagueu guanyat.

Figura B.3: Finestra principal

Podeu tornar a apostar i obtenir un resultat. Finalment, per sortir de l'aplicació, feu clic al botó Sortir.

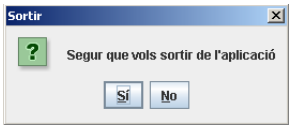

Figura B.4: Quadre de diàleg sortir