DOMOTROLA
Aplicació web pel control domòtic

Juanjo Cano Guirao
Enginyeria Tècnica d'Informàtica de Sistemes

Vicenç Font Sagrista
14 de Gener 2013

Mai havia comprès la veritable importància de les dedicatòries i agraïments que llegia en els llibres o treballs d'altres persones, fins ara que he realitzat el meu treball de final de carrera i abans de completar la seva realització ja sentia la necessitat d'escriure aquestes línies.

Dedico aquest treball a la meva dona, la Mónica, per la seva paciència i tot el suport rebut durant la carrera i en especial en les intensives hores dedicades a aquest projecte.

Al meu fill, el Dani, que amb 5 anys ha posat el seu granet posant títol al projecte i a qui li dec un munt d'hores de joc.

Agrair a tota la comunitat UOC tant companys com consultors el suport rebut durant aquests anys de carrera.

Gracies al Vicenç Font per la seva excepcional disponibilitat.

RESUM DEL TREBALL

La idea del treball va sorgir aprofitant un projecte personal pendent de realitzar. El projecte consisteix en la instal·lació de dispositius electrònics i/o mecànics en la meva llar per convertir-la en un habitatge domotitzat. Una part important del projecte és la part de software necessari pel control dels dispositius. Aquesta part és la que s'ha desenvolupat en aquest treball de final de carrera.

Per tant la finalitat d'aquest projecte és construir un aplicatiu per administrar i accionar remotament els controladors de maquinari dels dispositius d'un habitatge domotitzat. Es fa referència a l'accionament de controladors de maquinari de dispositius, i no de dispositius directament, per que es pot donar el cas que un mateix controlador gestioni varis dispositius, com pot ser el cas del reg automàtic on un mateix controlador de maquinari acciona diverses electrovàlvules.

L'aplicatiu disposa d'una interfície web i acciona els controladors mitjançant la xarxa local de l'habitatge. Per tant l'abast de l'aplicatiu serà una decisió de l'administrador del sistema, donant o no accés a la xarxa des de l'exterior.

Per a la realització del treball s'ha escollit l'àrea **TFC-J2EE**. La plataforma Java proporciona prou portabilitat i independència per a evitar problemes de cara a la futura elecció de hardware i sistemes operatius. El treball consisteix en la creació d'una aplicació web que gestioni i controli els dispositius de l'habitatge.

En la implementació de l'aplicatiu s'han emprat diferents eines disponibles sota la plataforma Java, com pot ser:

- **Eclipse** com a IDE per al desenvolupament del projecte.
- Bastiment **Hibernate** per al mapatge objecte-relacional (**ORM**) de la base de dades i el model d'objectes.
- Bastiment **Struts2** basat en el patró model-vista-controlador (**MVC**).
- Llibreria **JQuery**.
- **Tomcat** com a servidor d'aplicacions.
- **MySql** per a la base de dades.
- **Wireshark** per a realitzar els jocs de proves i constatar que l'aplicació envia les ordres als controladors a traves de la xarxa.

Els recursos de maquinari es basen simplement en un ordinador personal.

Índex de continguts.

Índex de figures.

1. Introducció

1.1. Justificació del TFC: punt de partida i aportació

Com a punt de partida tenim els coneixements adquirits durant els estudis d'enginyeria tècnica d'informàtica de sistemes, remarcant les assignatures enfocades al programari com poden ser POO, Enginyeria del programari, Base de Dades I i II, etc.

Una vegada plantejat el projecte s'han identificat alguns riscos com el desconeixement de la plataforma ja que malgrat que conec el llenguatge de programació estudiat durant la carrera, no he treballat mai amb algunes de les eines descrites anteriorment com Hibernate, Struts o Tomcat. Com a aportació addicional tindré la de buscar informació i assolir els coneixements necessaris sobre la plataforma J2EE i les eines de desenvolupament descrites, realitzar algunes proves prèvies, i algun curs per al desenvolupament d'aplicacions webs amb Java.

El temps serà un altre risc. Tenint en compte l'escàs temps de dedicació disponible per al projecte dintre de les dates pel lliurament del mateix i la feina suplementària pel desconeixement de la plataforma, es fa imprescindible un seguiment molt estricte de la temporització proposada per detectar en el menor temps possible les desviacions que es puguin produir.

1.2. Objectius del TFC

L'objectiu principal del present treball és realitzar l'anàlisi, disseny i implementació d'una aplicació web sota la plataforma J2EE utilitzant els bastiments Struts2 i Hibernate.

Com ha objectiu també entenem la demostració de l'assoliment per part de l'alumne de les tècniques estudiades durant els estudis d'enginyeria tècnica d'informàtica de sistemes. A part el present treball te com a objectiu l'aprenentatge en l'ús dels bastiments esmentats anteriorment i el patró MVC.

1.3. Enfocament i mètode seguit

Aquest projecte informàtic està enfocat sota el marc de la programació orientada al objecte. Per aquest motiu ha estat necessària la utilització del bastiment Hibernate mitjançant el qual s'ha realitzat el mapatge objecte-relacional (ORM) entre els objectes (classes Java) i les taules de la base de dades relacional.

Com tot projecte informàtic aquest ha de tenir un cicle de vida, i partint dels coneixements de la assignatura Enginyeria del Programari s'ha escollit un cicle de vida iteratiu per que fàcilment es produiran modificacions tant en els requisits com en el disseny. Les etapes seguides són recollida i documentació de requisits, anàlisi, disseny, implementació i proves.

1.4. Planificació del projecte

Durant el pla de treball es realitza la següent temporització inicial:

Fase del projecte	Durada	Data inici	Data fi
Reunió d'inici	1 dia	26-9-12	26-9-12
PLA DE TREBALL	**7 dies**	**27-9-12**	**3-10-12**
-Descripció del projecte	1 dia	27-9-12	27-9-12
-Definició de funcions	2 dies	28-9-12	29-9-12
-Definició de recursos i riscos	1 dia	30-9-12	30-9-12
-Temporització	1 dia	1-10-12	1-10-12
-Entrega document Pla de treball PAC1	0 dies	3-10-12	3-10-12
ANÀLISI I DISSENY	**35 dies**	**4-10-12**	**8-11-12**
-Definició de casos d'ús i actors	3 dies	4-10-12	6-10-12
-Fitxes de casos d'ús	1 dia	7-10-12	7-10-12
-Model de pantalles prototipus	2 dies	8-10-12	9-10-12
-Disseny relacional de la base de dades	5 dies	11-10-12	14-10-12
-Diagrama de classes principals	5 dies	15-10-12	19-10-12
-Diagrama d'arquitectura	5 dies	20-10-12	24-10-12
-Entrega document anàlisi i disseny PAC 2	0 dies	5-11-12	8-11-12
IMPLEMENTACIÓ	**39 dies**	**9-11-12**	**17-12-12**
-Base de dades	5 dies	9-11-12	13-11-12
-Interfície	3 dies	14-11-12	16-11-12
-Connexió	2 dies	17-11-12	18-11-12
-Usuaris	15 dies	19-11-12	3-12-12
-Controladors	5 dies	4-12-12	8-12-12
-Accionament dispositius	5 dies	9-12-12	13-12-12
-Entrega document implementació PAC3	0 dies	14-12-12	17-12-12
MEMORIA I PRESENTACIÓ	**28 dies**	**18-12-12**	**14-1-13**
-Revisió document memòria	5 dies	19-12-12	23-12-12

-Presentació	5 dies	26-12-12	30-12-12
-Instal·lable	5 dies	2-1-13	6-1-13
-Entrega final	0 dies	7-1-13	14-1-13

Il·lustració 1 - Taula de planificació del projecte

Malgrat es van detectar inicialment uns riscos, ha estat impossible evitar el desajustament d'aquesta temporització. El desajustament més greu es produeix en la fase d'implementació degut al desconeixement de les eines, tal i com es va predir.

1.5. Productes obtinguts

Una vegada acabat el treball obtenim varis documents o productes de la feina realitzada. El primer és el document Memòria del treball que correspon al present document, en el qual es descriu tot el procés que ha sofert el TFC. Un altre producte és la presentació. Es tracta d'una presentació virtual realitzada en un document presentació (tipus Power Point) la qual sintetitza de forma clara i concisa el treball realitzat al llarg del TFC i els resultats obtinguts. Per últim el producte resultant del TFC és l'aplicació obtinguda, que demostra d'una manera pràctica l'assoliment dels estudis d'Enginyeria Tècnica d'Informàtica de Sistemes.

1.6. Descripció de la resta de capítols

En els capítols següents veurem la feina realitzada durant les etapes del cicle de vida del projecte. En la recollida i documentació de requisits es realitza tal i com el seu nom indica la recollida d'informació mitjançant entrevistes amb els usuaris de l'aplicació per tal de documentar els requeriments que ha de complir l'aplicació. En la etapa d'anàlisi es descriuen els requeriments, obtingut en la etapa anterior, en un llenguatge més formal com els models i diagrames UML. La etapa de disseny, que fa de pont entre l'anàlisi i la implementació, es el primer pas en la elaboració d'una solució als requeriments de l'aplicació.

2. Recollida i documentació de requisits

2.1. Descripció

La finalitat d'aquest projecte és construir un aplicatiu per administrar i accionar remotament els controladors de maquinari dels dispositius d'un habitatge domotitzat. Es fa referència a l'accionament de controladors de maquinari de dispositius, i no de dispositius directament, per que es pot donar el cas que un mateix controlador gestioni varis dispositius, com pot ser el cas del reg automàtic on un mateix controlador de maquinari acciona diverses electrovàlvules.

L'aplicatiu disposa d'una interfície web i acciona els controladors mitjançant la xarxa local de l'habitatge. Per tant l'abast de l'aplicatiu serà una decisió de l'administrador del sistema, donant o no accés a la xarxa des de l'exterior.

2.2. Descripció de les funcionalitats

Volem elaborar una aplicació amb la qual poder controlar els dispositius mecànics i electrònics d'un habitatge domotitzat. Segons la informació recollida en el pla de treball, l'aplicació tindrà una funció principal de executar les accions dels CP i les macros i diverses funcions administratives, com són la gestió d'usuaris, CP, programa, plantilles i macro.

2.2.1. Connexió al sistema

Quan un usuari es connecta a l'aplicació el sistema demana el nom d'usuari i la paraula de pas. Disposem de dos tipus d'usuari, amb drets d'administració o sense. Quan instal·lem l'aplicació disposem d'un usuari administrador. Amb aquest usuari podem accedir a totes les funcions de l'aplicació.

Com a possible funció opcional o futura es planteja la identificació de l'usuari amb certificat digital o DNIe.

2.2.2. Gestió d'usuaris

Aquesta funcionalitat ens proporciona les eines necessàries per gestionar els usuaris que tenen accés a l'aplicació. Només tenen accés els usuaris amb drets d'administrador. Dintre d'aquesta gestió d'usuaris ens trobem les eines més bàsiques com ara:

- **Alta d'usuaris**. On podem afegir un nou usuari a la base de dades del sistema. S'hauran de omplir els camps obligatoris: tipus d'usuari, nom d'usuari i paraula de pas. Es disposaran d'altres camps opcionals com per exemple un camp descripció.
- **Baixa d'usuaris**. Ens dona l'opció de eliminar un usuari de la base de dades. L'usuari principal administrador no es pot eliminar.
- **Consulta i modificació d'usuaris**. On podem consultar i modificar les dades d'un usuari.

2.2.3. Modificació de la clau d'accés

Funció amb la que podem modificar la clau d'accés que ens dona accés a l'aplicació. A aquesta funció tenen accés tots els usuaris. (Durant el projecte s'estudiarà la necessitat o no de canviar la clau d'accés obligatòriament cada cert temps).

2.2.4. Gestió de CP (controladors de programari)

De manera anàloga a la gestió d'usuaris en la gestió de CP ens trobem amb les eines per gestionar els controladors de maquinari que estan connectats al sistema i evidentment només tenen accés els usuaris amb drets d'administrador. Disposem de les opcions següents:

- **Alta de CP**. Ens permet afegir un nou CP a la base de dades del sistema. Aquest controlador de maquinari associat al CP ha d'estar connectat a la xarxa local. Els camps obligatoris són:
    - Id: nombre identificatiu del CP.
    - IP: adreça IP del controlador de maquinari dintre de la xarxa on s'enviaran els senyals de cada acció.

- Descripció: Una petita descripció que facilitarà la identificació del dispositiu per part de l'usuari.
- Llista d'accions: Es tracta d'un recull de les accions que admet el controlador de maquinari. Cada acció estarà definida per una descripció. Per exemple el CP *Llum cuina* tindrà dues accions, la acció *Encendre* i la acció *Apagar*. En un altre exemple, el CP *Persiana dormitori 1* tindrà tres accions, *Pujar*, *Baixar* i *Parar*.

- **Baixa de CP**. Opció que possibilita la eliminació de CP de la base de dades.
- **Consulta i modificació de CP.** Per consultar i modificar la informació de un CP existent a la base de dades.

2.2.5. Plantilles de CP

La funció d'aquestes plantilles és la de facilitar la creació de CP, amb característiques similars, de manera repetitiva sense haver de definir totes les accions. Per exemple si creem una plantilla *llum* amb els controls de les dues accions (encendre i apagar), quan vulguem crear els CP de tots els llums de l'habitatge, a partir de la plantilla, només haurem d'anar indicant l'adreça IP i la descripció del controlador de maquinari.

El registre de plantilles és una eina administrativa per a la creació de CP, per tant el seu accés estarà limitat als usuaris amb drets d'administrador. La gestió de plantilles serà molt similar a la de CP (altes, baixes, consulta i modificació), l'única diferencia és que les plantilles no tindran adreça IP ja que no estan assignades a cap controlador de maquinari en concret.

2.2.6. Accionament dels dispositius

La funció principal serà la de accionar els dispositius de l'habitatge a traves dels controladors de maquinari connectats a la xarxa local. L'usuari disposarà, en una interfície senzilla i amigable, de varis controls que dependran del CP que estigui seleccionat en cada moment. Cada control estarà associat a una acció del CP . També es tindrà accés a la llista de les macros que hagin estat creades per poder executar-les.

2.2.7. Macros

Les macros són una altra eina, no administrativa, però que la seva gestió estarà limitada als usuaris administradors. Els usuaris no administradors només tindran dret a executar les que hagin estat creades prèviament per algun usuari administrador.

Aquesta eina consisteix com el seu nom indica en una macro de accions. En les macros podem emmagatzemar un conjunt d'accions de diferents CP que s'executaran seqüencialment amb una sola ordre.

Les macros tindran tres camps: Identificador, descripció i llista d'accions. Aquesta llista d'accions estarà composta per parelles de identificació de CP i acció corresponent. I la seva gestió proporcionarà eines per afegir macros, donar de baixa, consultar i modificar.

2.2.8. Programació

L'aplicatiu disposarà d'una programació setmanal per automatitzar encara més el control dels CP. En un calendari setmanal i en franges de 1 minut podrem programar el CP desitjat i la acció que realitzarà. No cal dir que l'accés a aquesta programació serà amb drets d'administració.

2.2.9. Càmeres

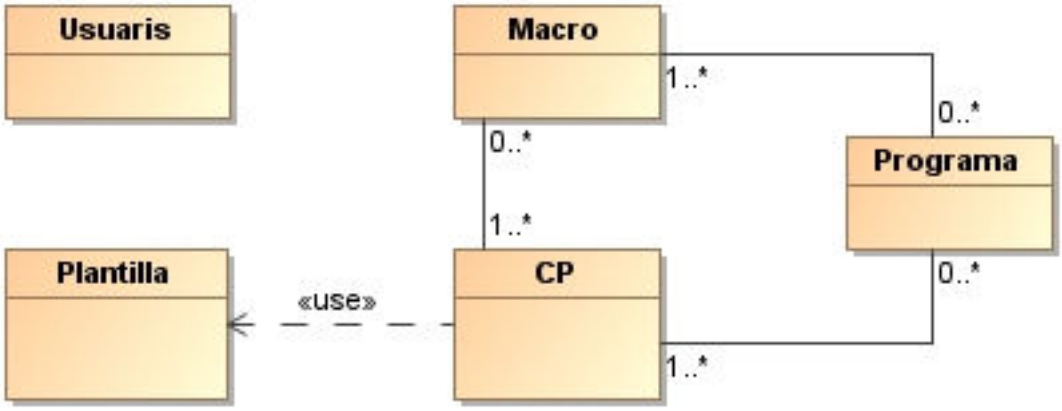
Com a funció futura existeix la possibilitat de incorporar un tipus de controladors per rebre les imatges de una o varies càmeres instal·lades a la xarxa. A través d'aquestes càmeres es podrà visualitzar el que ocorre a cada lloc de l'habitatge. Funció obligatòria que es demana actualment en tots els sistemes de alarmes de seguretat.

2.3. Model del domini

En una primera ullada identifiquem els objectes CP i programa que es corresponent a objectes del món exterior.

Com a objectes del negoci identifiquem dos objectes més, les macros i les plantilles.

Com a esdeveniments identifiquem la execució automàtica dels CP i macros que estan anotats en el programa.



Il·lustració 2 - Diagrama del model de domini

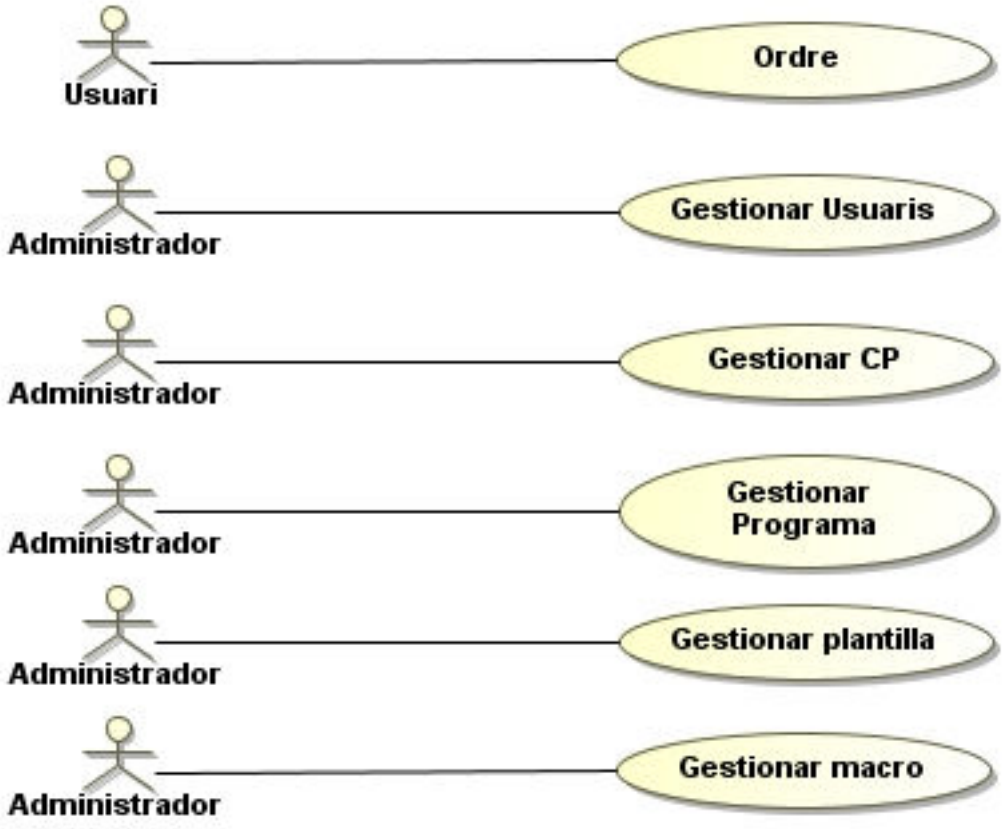
2.4. Model del negoci

Amb la informació anterior podem elaborar uns primers casos d'ús: Quan un usuari entra a l'aplicació pot executar CP o macros. Si aquest usuari és un usuari administrador, disposa de unes quantes funcions més. Com la gestió de usuaris, CP, plantilles, macros i programa amb la que podem programar la execució automàtica dels CP i macros.

La relació entre els objectes identificats i aquest casos d'ús és:

- Executar acció: CP, Macro
- Gestionar usuaris: Usuaris
- Gestionar CP: CP, plantilla
- Gestionar programa: Programa, CP, Macro
- Gestionar plantilla: plantilla
- Gestionar macro: Macro, CP

El diagrama de casos d'ús del model del negoci és aquest:



Il·lustració 3 - Diagrama de casos d'ús del model de negoci

2.5. Guions

S'han identificat dos guions relacionats amb els casos d'ús:

- D'una part el guió del usuari que vol accionar algun dels dispositius de l'habitatge. L'usuari entra a l'aplicació, prèvia identificació, i escull l'acció del CP seleccionat, o la macro que vol executar.
- El segon guió correspon a l'usuari administrador que vol gestionar algun dels objectes. L'usuari administrador entra a l'aplicació, prèvia identificació, i selecciona la opció del menú desitjada corresponent al objecte (usuari, CP, plantilla, macro o programa) que vol gestionar.

2.6. Casos d'ús

A continuació en els següents subapartats farem la identificació i descripció dels casos d'ús corresponents a la situació plantejada.

2.6.1. Actors

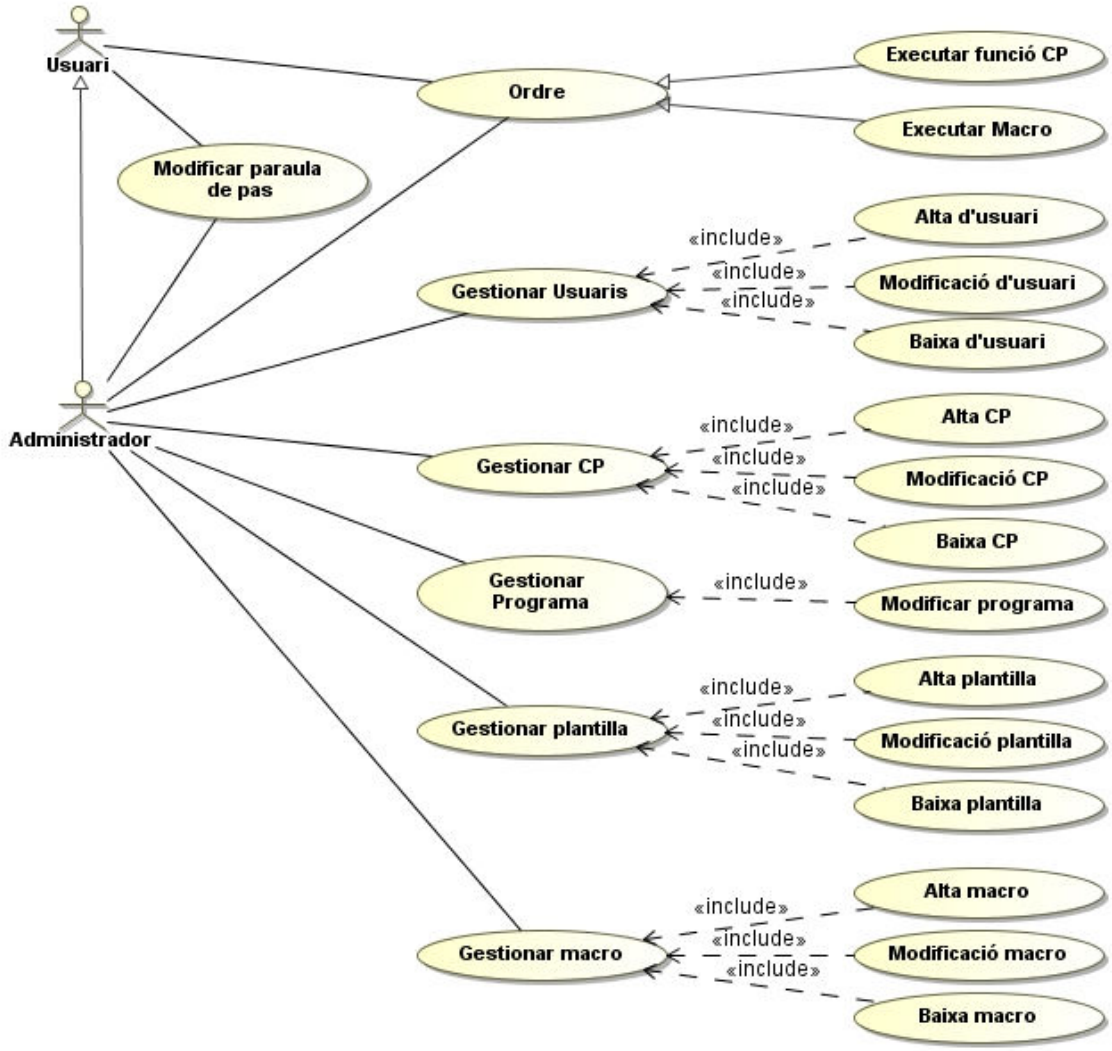
Hem identificat dos actors corresponents a les úniques entitats externes al programari que interactuen amb ell. Aquestes entitats, l'usuari i l'usuari administrador, corresponent a persones físiques del mon exterior.

L'usuari només té un paper, el de executar les accions dels dispositius. En canvi l'usuari administrador a més de executar accions igual que l'usuari normal, té cinc papers més corresponents a la gestió dels usuaris, CP, plantilles, macros i programa.

Podem apreciar que entre els dos actors existeix una relació d'especialització, ja que, tenim un cas d'ús que poden fer els dos.

2.6.2. Diagrama de casos d'ús

El diagrama de casos d'ús resultant de tot aquest plantejament és el següent:



Il·lustració 4 - Diagrama de casos d'ús

2.6.3. Documentació textual

2.6.3.1. Cas d'ús: "Ordre"

- Resum de la funcionalitat: Executa o be una funció de CP o be una macro.
- Paper dins del treball de l'usuari: és el cas d'ús principal en el que intervenen els *usuaris*. En el cas dels *administradors* aquest paper es pot considerar com a secundari.
- Actors: **usuari** i **administrador**.
- Casos d'ús relacionats: cap.
- Precondició: CP: El CP existeix a la base de dades amb la funció desitjada.

Macro: La macro existeix a la base de dades.

- Postcondició: El sistema envia el senyal al dispositiu.

L'usuari o administrador una vegada s'han identificat accedeixen a l'apartat d'ordres on disposen d'una llista de CP i una llista de macros. En cas de seleccionar un CP poden veure les accions que te disponibles i executar una de elles. En cas de seleccionar una macro poden veure la llista d'accions que estan incloses en la macro i executar la macro (s'executa la llista d'accions sencera).

2.6.3.2. Cas d'ús: "Modificar la paraula de pas"

- Resum de la funcionalitat: L'usuari modifica la seva paraula d'accés al sistema.
- Paper dins del treball de l'usuari: és un cas d'ús que s'aconsella fer servir amb freqüència per raons de seguretat del sistema.
- Actors: **usuari** i **administrador**.
- Casos d'ús relacionats: cap.
- Precondició: L'usuari a entrat al sistema identificant-se amb el nom i la paraula de pas.
- Postcondició: La paraula de pas ha estat modificada.

L'usuari o administrador una vegada s'han identificat accedeixen a l'apartat modificar paraula de pas on veuran un petit formulari que hauran d'omplir. Aquest formulari constarà de tres camps: paraula de pas antiga, paraula de pas nova i repetició de la paraula de pas nova. Una vegada omplerts disposarà de dos botons un per cancel·lar i el segon per acceptar la modificació.

2.6.3.3. Cas d'ús: "Gestionar usuaris"

- Resum de la funcionalitat: Gestiona la informació dels usuaris existent a la base de dades.
- Paper dins del treball de l'usuari: és cas d'ús molt esporàdic dins del treball de l'administrador. Ja que aquesta aplicació no requereix de un gran nombre de usuaris ni de massa informació sobre ells.
- Actors: **administrador**.
- Casos d'ús relacionats: cap.

- Precondició: Alta: L'usuari no existeix a la base de dades.

  Baixa o modificació: L'usuari ja existeix a la base de dades.

- Postcondició: Alta: Les dades de l'usuari s'han afegit a la base de dades.

  Baixa: Les dades de l'usuari s'han eliminat de la base de dades.

  Modificació: Les dades de l'usuari han estat modificades.

L'administrador accedeix a l'apartat de gestió d'usuaris on disposa de la llista d'usuaris emmagatzemats a la base de dades. Seleccionant un usuari d'aquesta llista pot veure la informació sobre aquest com: Nom, tipus usuari, paraula de pas i descripció. Una vegada seleccionat un usuari el sistema ens dona la possibilitat de modificar les dades d'aquest usuari, donar-lo de baixa o de crear un usuari nou. El valor del camp tipus usuari el fa servir el sistema per diferenciar els *usuaris* dels *administradors*.

2.6.3.4. Cas d'ús: "Gestionar CP"

- Resum de la funcionalitat: Gestiona la informació dels CP existent a la base de dades.

- Paper dins del treball de l'usuari: aquest també és un cas d'ús molt esporàdic dins del treball de l'administrador. No serà gaire freqüent que la informació dels dispositius de l'habitatge canviï una vegada introduïda tota aquesta informació.

- Actors: **administrador**.

- Casos d'ús relacionats: cap.

- Precondició: Alta: El CP no existeix a la base de dades.

  Baixa o modificació: El CP ja existeix a la base de dades.

- Postcondició: Alta: Les dades del CP s'han afegit a la base de dades.

  Baixa: Les dades del CP s'han eliminat de la base de dades.

  Modificació: Les dades del CP han estat modificades.

La funcionalitat d'aquest cas d'ús és igual al cas d'ús anterior, l'administrador accedeix a l'apartat de gestió de CP on disposa de la llista de CP emmagatzemats a la base de dades. Seleccionant un CP d'aquesta llista pot veure la informació sobre aquest com: ID, IP, descripció i llista d'accions. Cada acció estarà definida per una descripció. Una vegada seleccionat un CP el sistema ens dona la possibilitat de modificar les dades d'aquest CP, donar-lo de baixa o de crear un CP nou. A l'hora de crear un de nou

tindrem la possibilitat de crear-lo partint d'una de les plantilles existents a la base de dades. En cas de seleccionar aquesta opció tindrem accés a la llista de plantilles per seleccionar una.

2.6.3.5. Cas d'ús: "Gestionar plantilles"

- Resum de la funcionalitat: Gestiona la informació de les plantilles existents a la base de dades.
- Paper dins del treball de l'usuari: aquest també és un cas d'ús molt esporàdic dins del treball de l'administrador. La informació d'una plantilla correspon a un tipus de dispositiu, encara que afegim dispositius a l'habitatge, les plantilles no canviaran gaire.
- Actors: **administrador**.
- Casos d'ús relacionats: cap.
- Precondició: Alta: La plantilla no existeix a la base de dades.

  Baixa o modificació: La plantilla ja existeix a la base de dades.
- Postcondició: Alta: Les dades de la plantilla s'han afegit a la base de dades.

  Baixa: Les dades de la plantilla s'han eliminat de la base de dades.

  Modificació: Les dades de la plantilla han estat modificades.

La funcionalitat d'aquest cas d'ús és idèntica al cas d'ús anterior, amb la única diferencia que les plantilles no disposen de camp IP, ja que no estan relacionades directament a cap dispositiu en concret. A l'hora de crear una plantilla nova tindrem la possibilitat de crear-la partint d'un dels CP existents a la base de dades. En cas de seleccionar aquesta opció tindrem accés a la llista de CP per seleccionar un.

2.6.3.6. Cas d'ús: "Gestionar macro"

- Resum de la funcionalitat: Gestiona la informació de les macros existents a la base de dades.
- Paper dins del treball de l'usuari: es preveu que aquest serà un cas d'ús més freqüent que els altres. Ates a la flexibilitat que proporciona hi haurà una gran demanda per part dels usuaris de crear noves macros i modificar les existents.

Aquesta previsió està basada en la diversitat que normalment existeix entre els usuaris en relació a horaris de estada a l'habitatge, zones més habitades, etc, etc.

- Actors: **administrador**.
- Casos d'ús relacionats: cap.
- Precondició: Alta: La macro no existeix a la base de dades.

  Baixa o modificació: La macro ja existeix a la base de dades.
- Postcondició: Alta: Les dades de la macro s'han afegit a la base de dades.

  Baixa: Les dades de la macro s'han eliminat de la base de dades.

  Modificació: Les dades de la macro han estat modificades.

La funcionalitat d'aquest cas d'ús segueix el mateix patró que les altres, disposem d'una llista de macros on al seleccionar una podem veure la seva informació. Una macro té un camp ID, una descripció i una llistra d'accions de CP. De igual manera disposem de les opcions de modificar, baixa i crear nova.

2.6.3.7. Cas d'ús: "Gestionar programa"

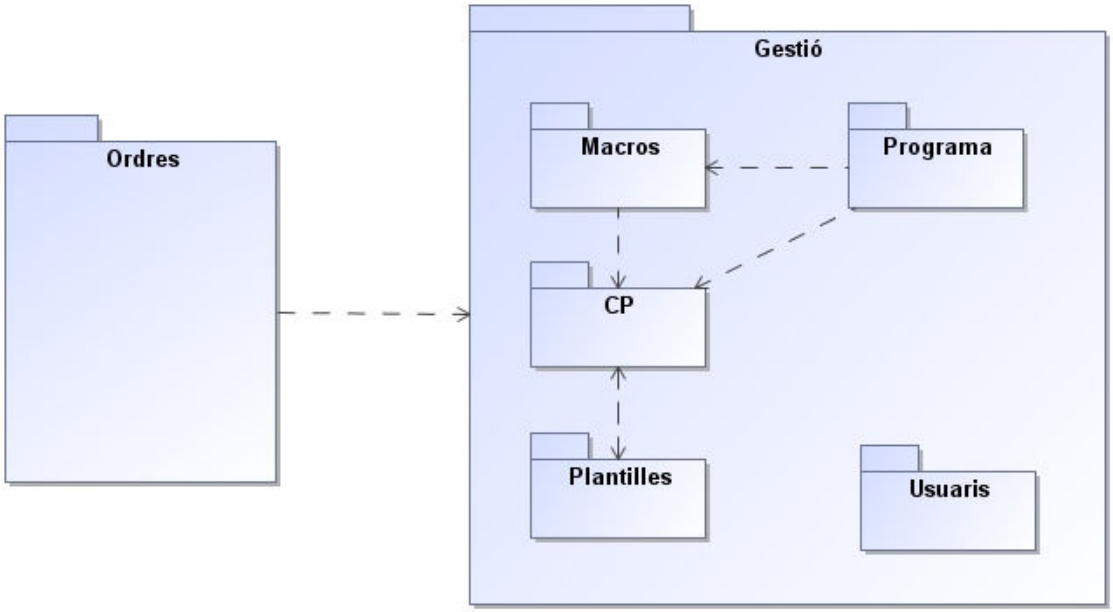
- Resum de la funcionalitat: Gestiona la programació automàtica dels dispositius.
- Paper dins del treball de l'usuari: la freqüència d'aquest cas d'us dependrà de les variacions que es produeixin ens els costums i horaris dels usuaris. Per exemple si un usuari canvia el seu horari de feina, possiblement voldrà modificar l'hora en que s'encén el llum del seu dormitori.
- Actors: **administrador**.
- Casos d'ús relacionats: cap.
- Precondició: Existeixen CP o macros a la base de dades.
- Postcondició: El programa ha estat modificat.

Quan l'administrador entra en l'apartat de programa disposa d'un calendari setmanal on pot programar que accions de CP o macros s'executaran automàticament el dia i l'hora indicats al programa. Seleccionant dintre del calendari veurem el detall de la entrada i podrem eliminar-la o modificar-la. També hi haurà la opció per afegir una entrada nova al programa.

3. Anàlisi

3.1. Paquets

A pesar de ser una aplicació molt senzilla, atenen al grau de dependència, podem diferenciar dos paquets d'anàlisi: el paquet d'ordres que inclou el cas d'ús *Ordres* i el paquet de gestió que inclou la resta de casos. Dintre del paquet de gestió podem diferenciar cinc paquets de serveis corresponents a les cinc classes de les que podem realitzar una gestió. L'esquema de les relacions entre els paquets és el següent:



Il·lustració 5 - Esquema de paquets d'anàlisi

3.2. Classes d'entitats

3.2.1. Identificació

A primera vista identifiquem les següents classes d'entitats: *Usuaris, CP, Plantilles, Macros, Programa*. Però analitzant les funcionalitats dels casos d'ús, trobem necessari afegir una nova classe anomenada *accions*, on inclourem la informació de les diferents accions dels CP.

3.2.2. Atributs de les classes d'entitats

A continuació presentem els atributs corresponents a cada classe d'entitat.

- Classe *Usuaris*: id (integer), nom (string), tipus (string), pass (string), descripcio (string)
- Classe *CP*: id (integer), ip (string), descripcio (string), descripcio2 (string)
- Classe *Plantilles*: id (integer), descripcio (string), descripcio2 (string)
- Classe *Macros*: id (integer), descripcio (string)
- Classe *Programa*: id (integer), estat (integer), descripcio (string)
- Classe *Accions*: id (integer), descripcio (string)

3.3. Relacions

3.3.1. Herència

Els atributs comuns a les classes *CP* i *Macros* es posen dins una superclasse abstracta, *Programable*. Així facilitem les funcions de la classe *Programa*.

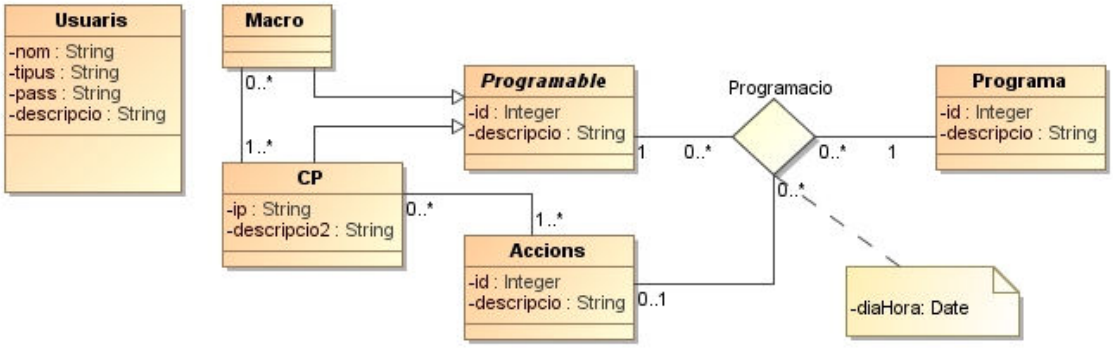
3.3.2. Associacions

Trobem varies associacions entre classes:

- La classe *Accions* té una associació binaria amb les classes *Plantilles* i *CP*.
- Existeix una associació ternària entre *Programa*, *Programable* i *Accions*. Aquesta associació afegeix un atribut *diaHora* del tipus date.

3.3.3. Diagrama de classes

El diagrama de classes resultant és el següent:

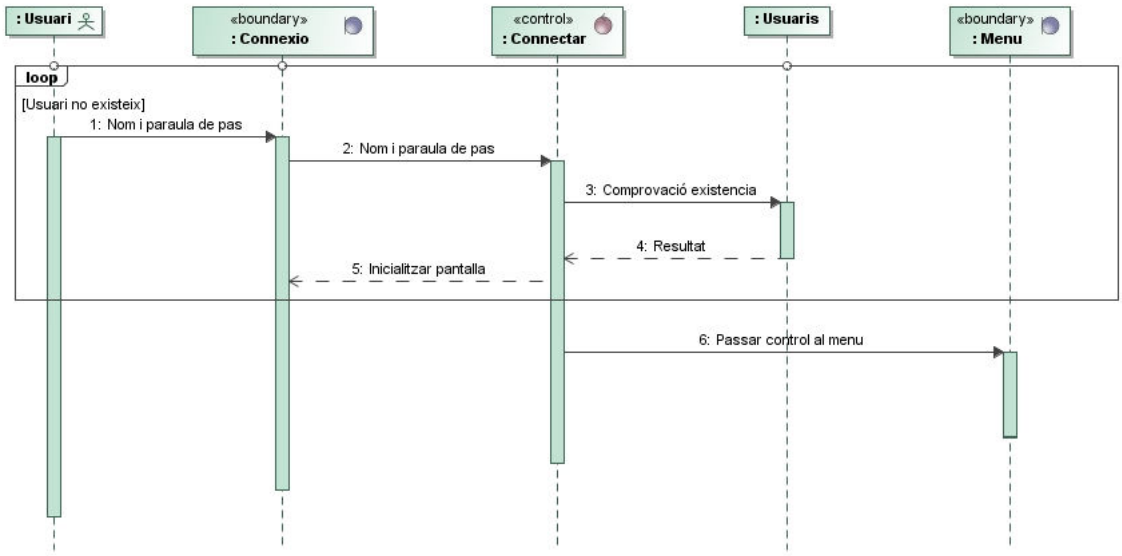


Il·lustració 6 - Diagrama de classes

3.4. Revisió dels casos d'ús

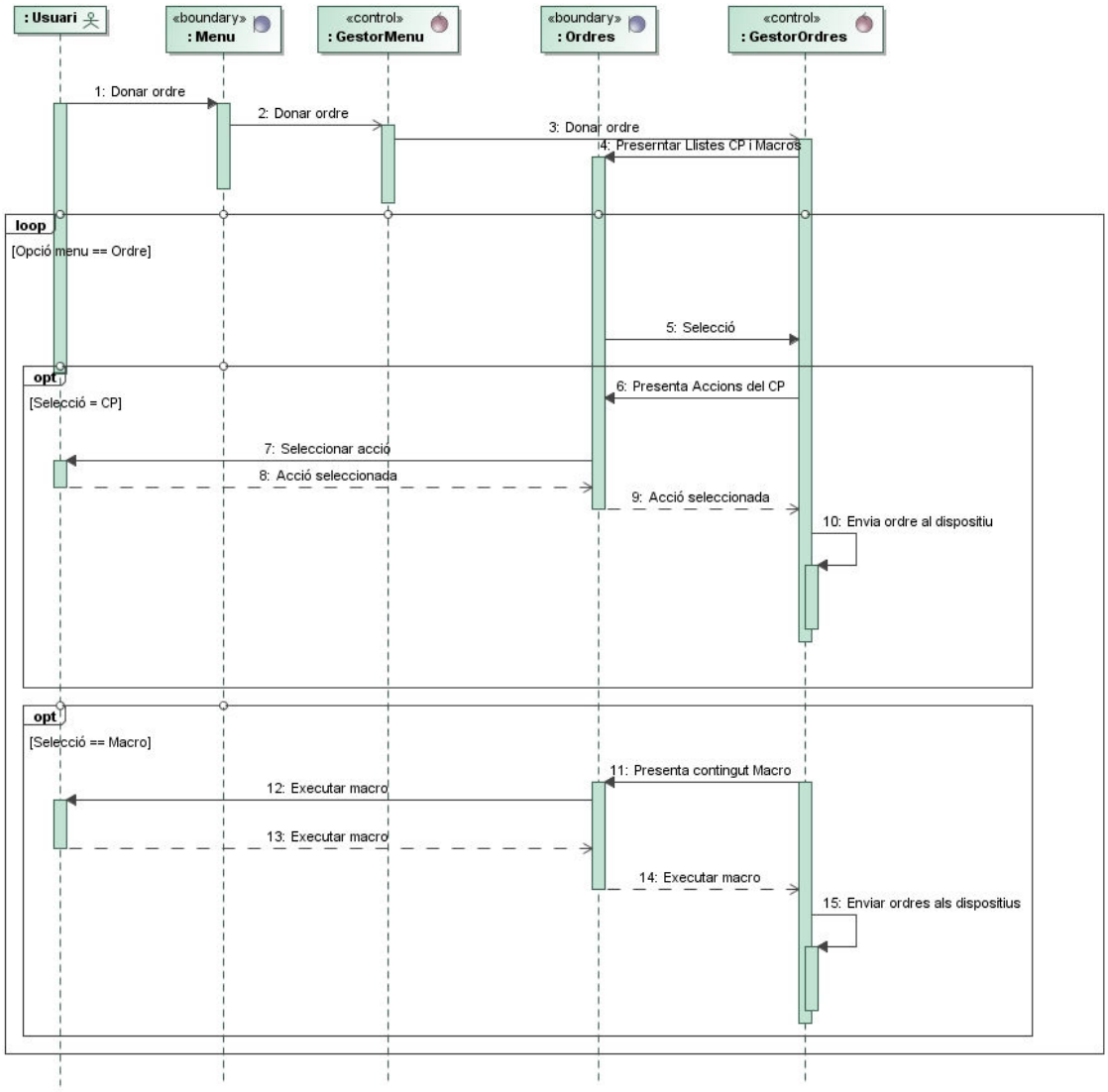
En els casos d'ús descrits en els requeriments s'ha comentat la necessitat de una identificació prèvia per part de l'usuari. També hem de recordar que abans hem afegit la classe *accions*, per tant ara toca afegir el cas d'ús *Gestionar Accions*. Tot això ens porta a fer una revisió dels casos d'ús ja que aquests fets no queden ben reflectits. A més a més aquests casos d'ús són molt senzills. A continuació proposarem uns diagrames de seqüència per definir una mica més els casos d'ús i aprofitarem per identificar les classes de frontera, les classes de control i les operacions en cada cas d'ús.

3.4.1. Cas d`ús connexió



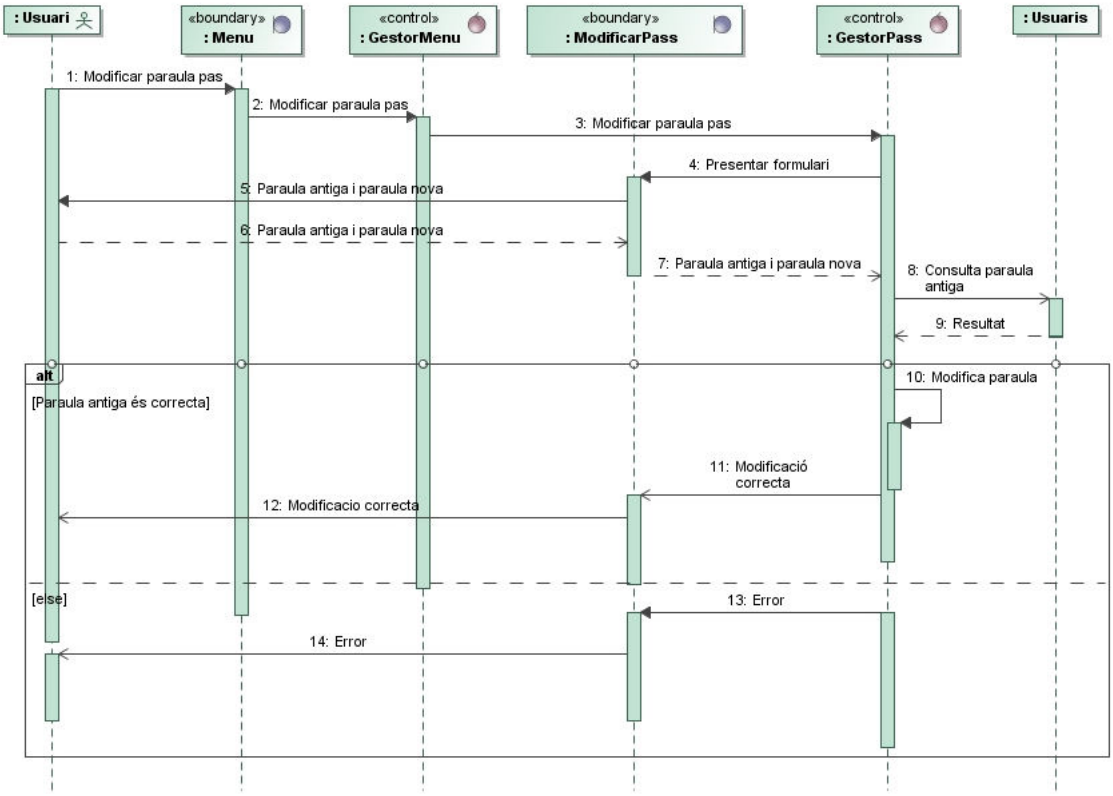
Il·lustració 7 - Diagrama de seqüència Cas d'ús connexió

3.4.2. Cas d'ús Ordre



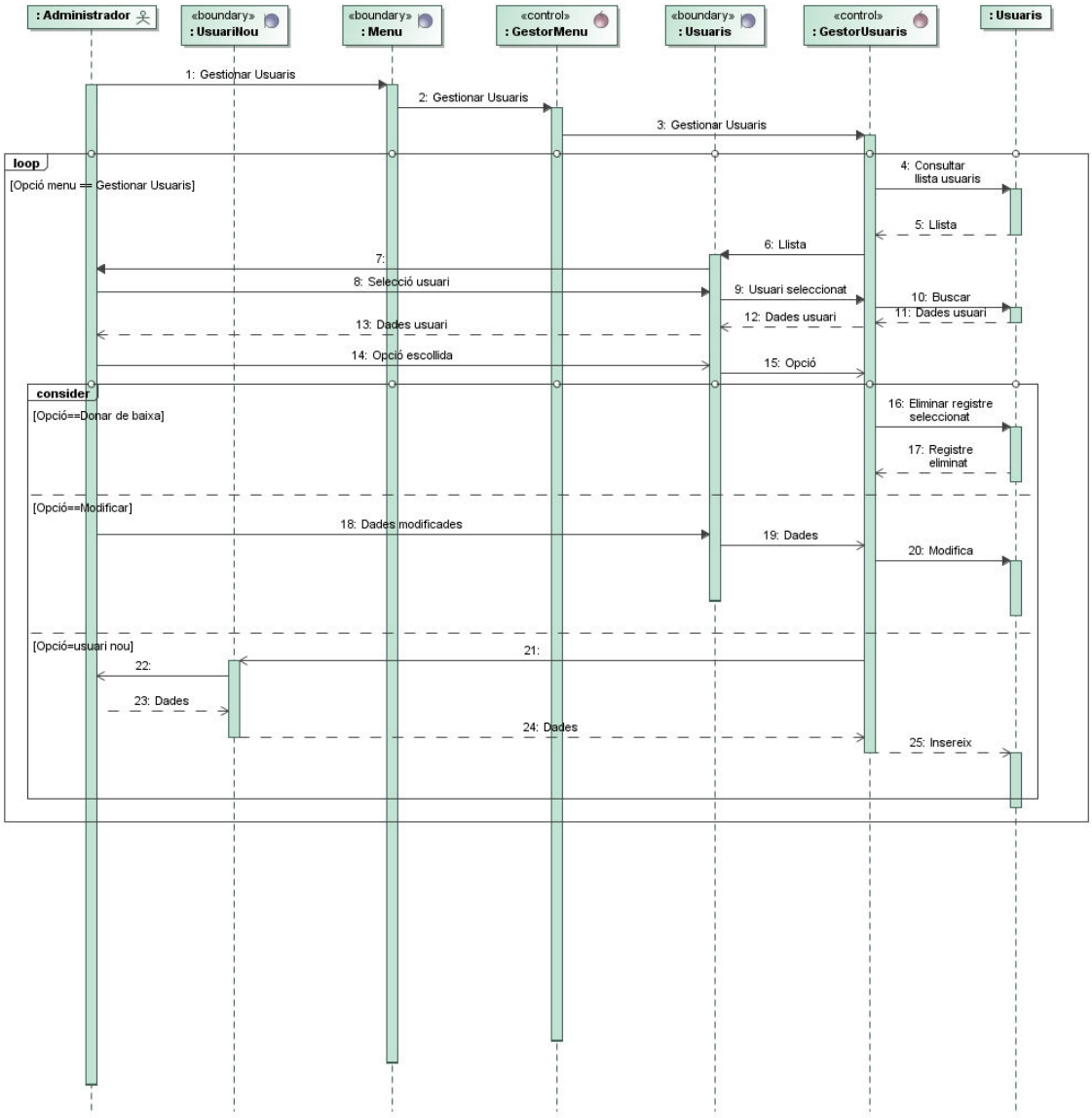
Il·lustració 8 - Diagrama de seqüència Cas d'ús Ordre

3.4.3. Cas d'ús Modificar clau d'accés



Il·lustració 9 - Diagrama de seqüència Cas d'ús Modificar clau d'accés

3.4.4. Cas d'ús Gestionar usuaris

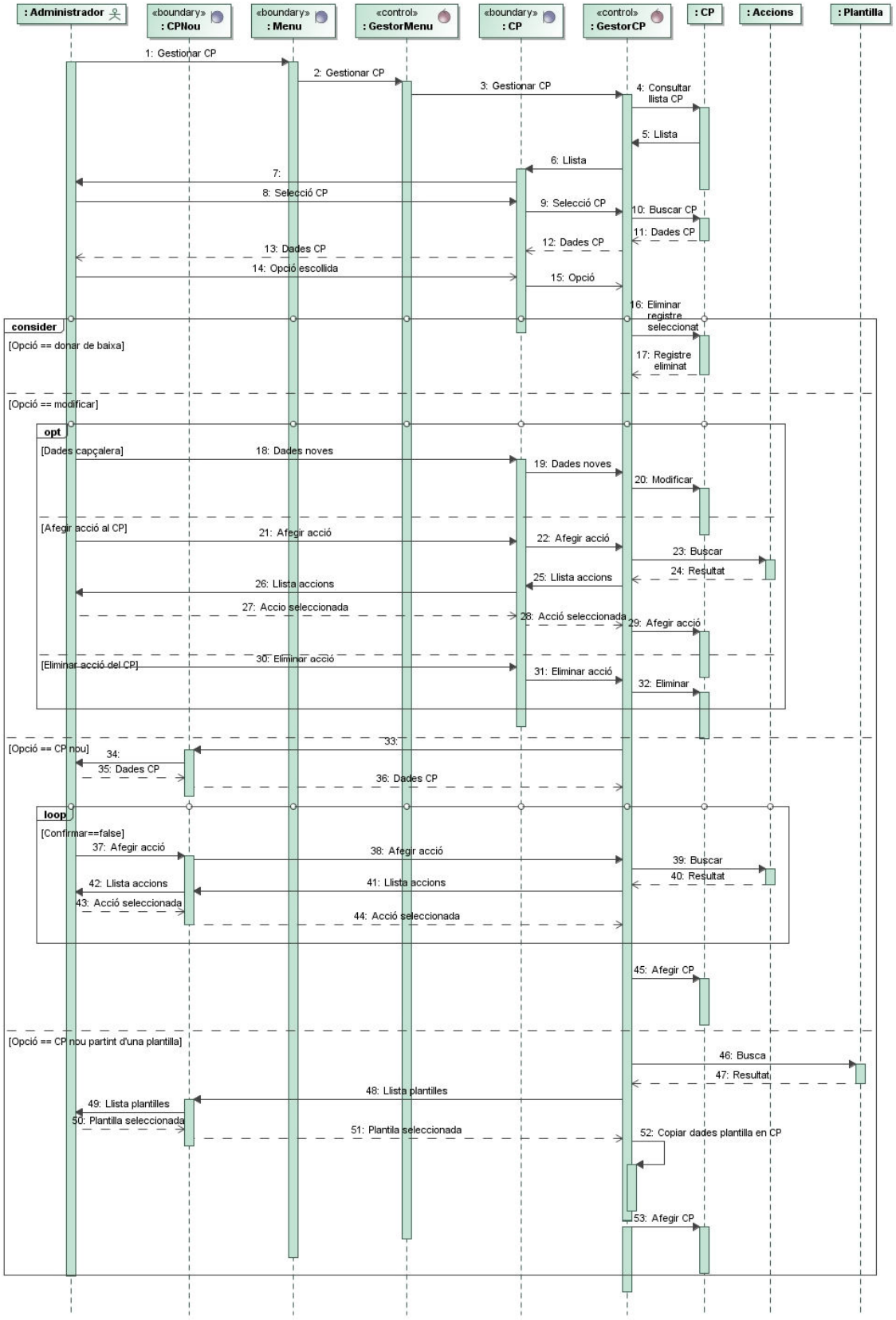


Il·lustració 10 - Diagrama de seqüència Cas d'ús Gestionar usuaris

3.4.5. Cas d'ús Gestionar Accions

Aquest cas d'ús funciona exactament com el cas d'ús *Gestionar Usuaris*, per tant hem considerat que el diagrama anterior és vàlid també per a aquest cas d'ús.

3.4.6. Cas d'us Gestionar CP

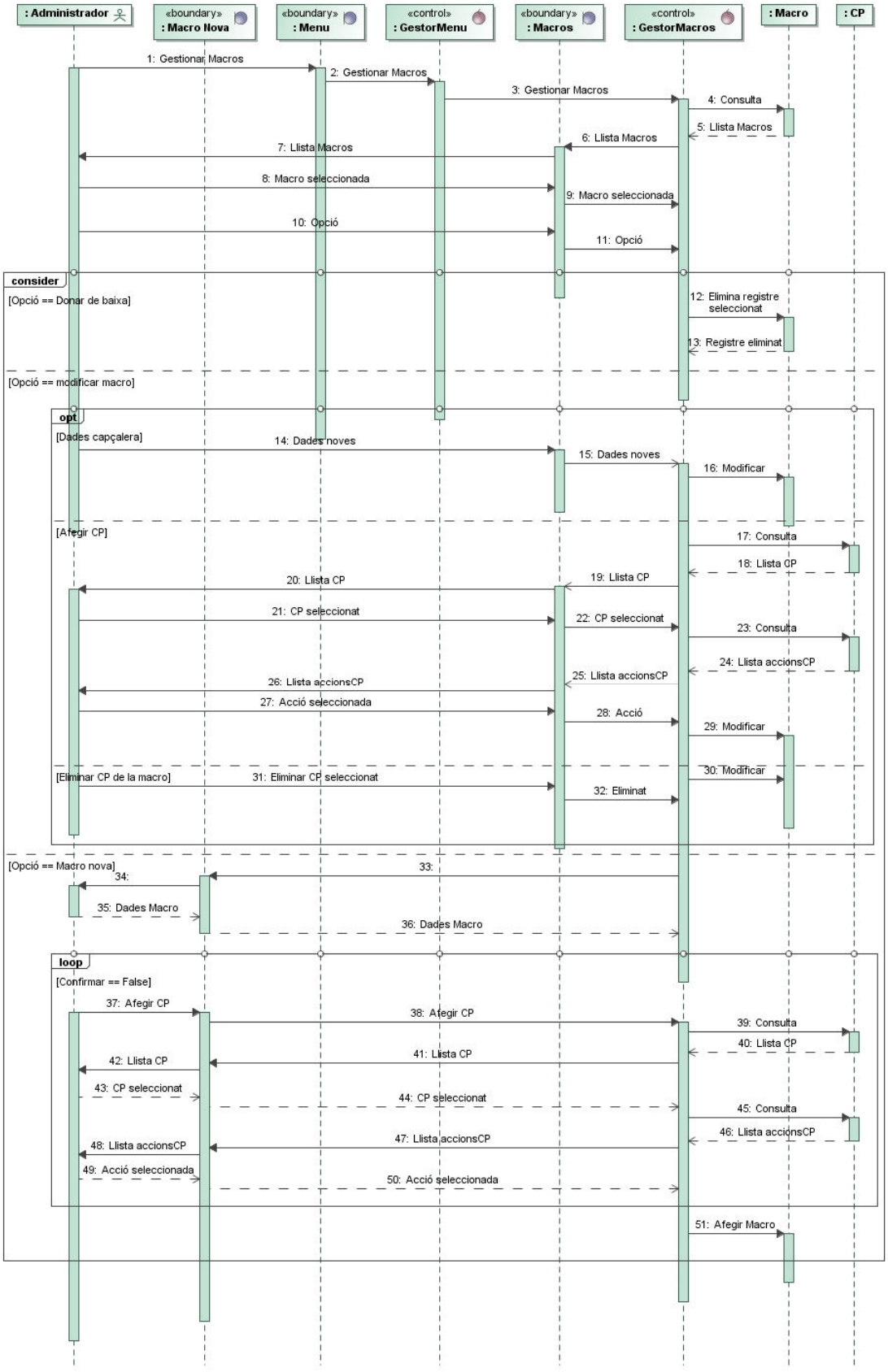


Il·lustració 11 - Diagrama de seqüència Cas d'ús Gestionar CP

3.4.7. Cas d'ús Gestionar Plantilles

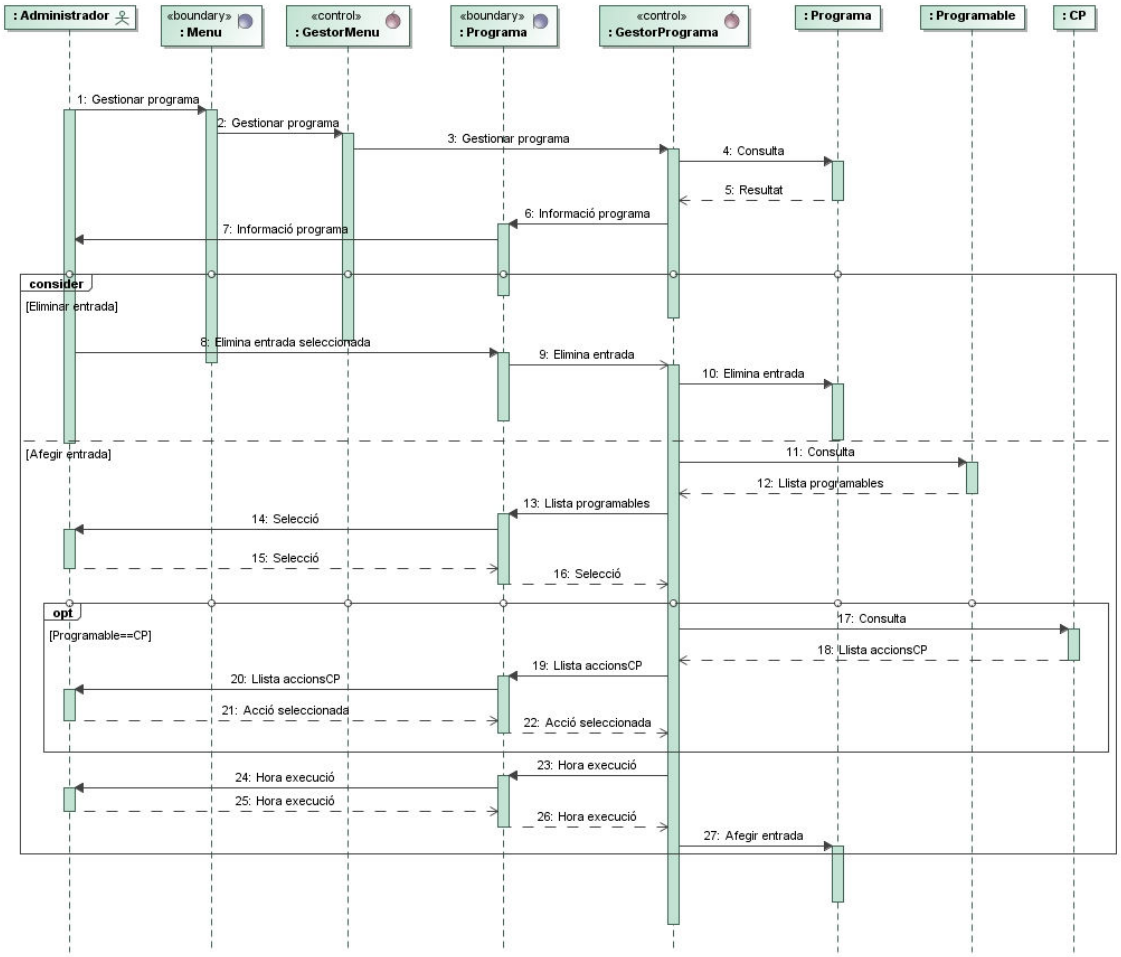
En aquest cas d'ús també podem aprofitar el diagrama anterior ja que el funcionament dels dos casos d'ús és idèntic.

3.4.8. Cas d'ús Gestionar macro



Il·lustració 12 - Diagrama de seqüència Cas d'ús Gestionar Macro

3.4.9. Cas d'ús Gestionar programa



Il·lustració 13 - Diagrama de seqüència Cas d'ús Gestionar Programa

4. Disseny

4.1. Disseny arquitectònic

4.1.1. Patró de disseny

Per al present projecte s'ha escollit el patró de disseny MVC (model-vista-controlador) generat per Smalltalk. Aquest patró és molt utilitzat en aplicacions web i és considerat de gran solidesa. En la seva estructura separa les dades de l'aplicació, la interfície de l'usuari i la lògica de negoci en tres components:

- El model correspon al sistema de gestió de base de dades i la lògica de negoci.
- La vista és la pàgina HTML i el codi que proveeix de dades dinàmiques a la pàgina.
- El controlador és el responsable de rebre els esdeveniments de entrada des de la vista. A partir d'aquestes accions de l'usuari el controlador invoca crides al model i a la vista.

4.1.2. Tecnologies utilitzades

Com a llenguatge i plataforma de programació s'ha escollit J2EE (JSP, Servlets, EJB, JQuery).

Per a implementar el patró MVC es farà servir el bastiment Struts2 que el tenim disponible per a la plataforma J2EE.

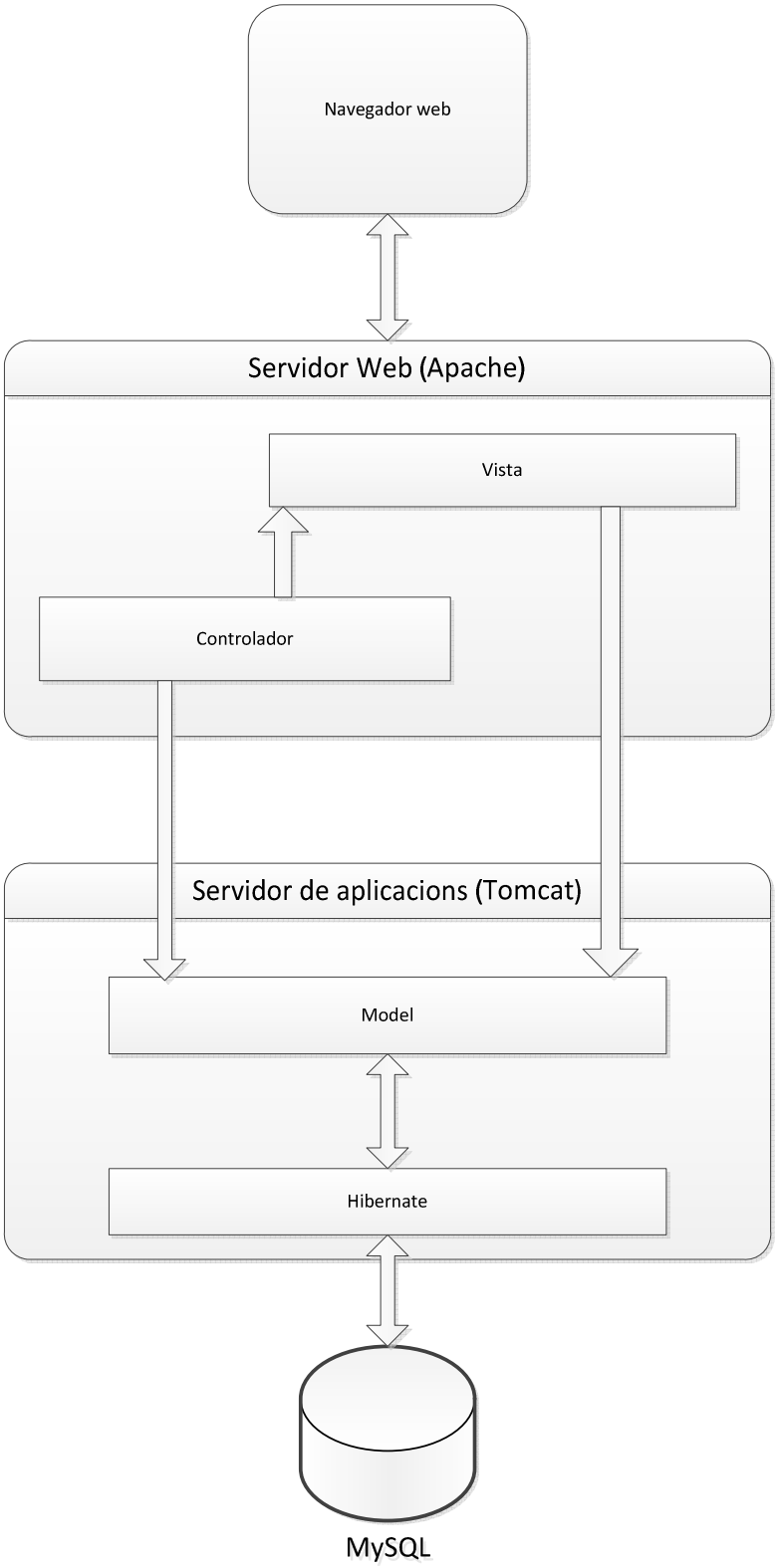
Per a la capa de persistència farem servir un altre bastiment anomenat Hibernate. Amb aquest bastiment podrem realitzar el mapatge objecte-relacional (ORM) entre els objectes del model i les taules de la base de dades.

Com a sistema de gestió de bases de dades relacional utilitzarem MySQL.

Per últim com a servidor d'aplicacions Apache tomcat.

4.1.3. Diagrama de l'arquitectura

El diagrama següent mostra l'arquitectura de l'aplicació on es pot veure la triangulació del model MVC.



Il·lustració 14 - Diagrama de l'arquitectura

4.2. Disseny de la persistència

Una vegada obtingudes les entitats necessàries per l'aplicació hem de dissenyar la persistència d'aquelles classes que ho requereixin. Aquest disseny de persistència s'ha fet en base a que l'aplicació farà servir el sistema gestor de base de dades relacional MySQL. I com a gestor de disc es farà servir el bastiment Hibernate.

4.2.1. Disseny de la base de dades

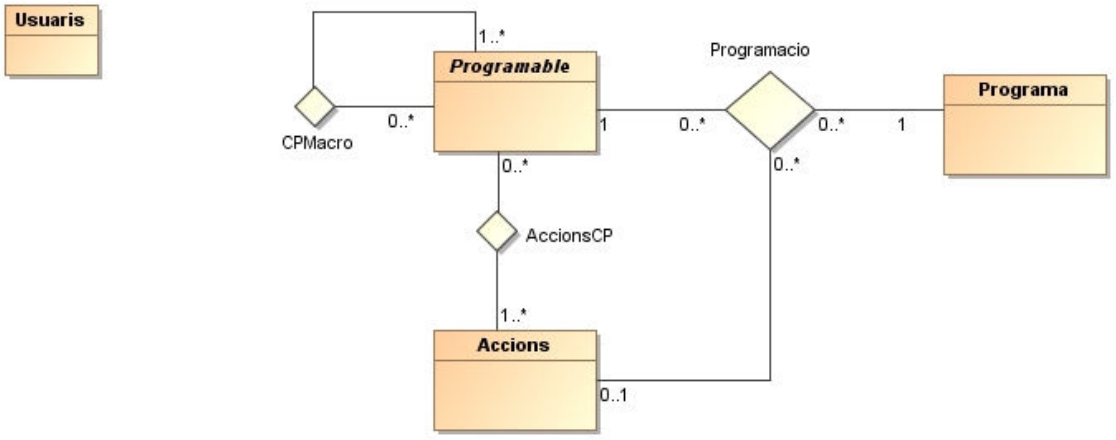
El disseny de la base de dades ho assolirem en tres etapes. Primer eliminarem l'herència, després elaborarem el diagrama ER equivalent al diagrama de les classes persistents i per últim especificarem les taules de la base de dades relacional.

Supressió de l'herència

Al diagrama de classes trobem una jerarquia d'herència encapçalada per la classe *Programable*. De les opcions que existeixen per a la supressió de l'herència, la més adient en aquest cas és la de implementar una taula única per a tota la jerarquia d'herència. Aquesta opció consisteix en crear una sola taula amb tots els camps, tant els de la superclasse com el de totes les subclasses, i afegint un camp que ens indiqui a quina subclasse pertany. Aquesta solució és menys eficient en ocupació de disc, però per les característiques de l'aplicació no es preveu que aquesta taula sigui molt gran en nombre de registres.

Diagrama entitat-relació

El diagrama resultant és el següent:



Il·lustració 15 - Diagrama entitat-relació

Base de dades relacional

En la transformació del diagrama ER anterior al model relacional obtenim quatre taules corresponents a les entitats resultants i tres taules més corresponents a les associacions entre aquestes taules. La especificació de les taules és la següent:

USUARIS (nom, tipus, pass, descripció)

PROGRAMABLE(id, descripcio, tipus, ip, descripcio2)

ACCIONS(id, descripcio)

PROGRAMA(id, estat, descripcio)

ACCIONSCP(accio, cp)
{accio} es clau forana cap a ACCIONS {id}
{cp} es clau forana cap a PROGRAMABLE {id}

CPMACRO(macro, accio, cp)
{macro} es clau forana cap a PROGRAMABLE {id}
{accio} es clau forana cap a ACCIONS {id}
{cp} es clau forana cap a PROGRAMABLE {id}

PROGRAMACIO (programa, accio, programable, diaHora)
{programa} es clau forana cap a PROGRAMA {id}
{accio} es clau forana cap a ACCIONS {id}
{programable} es clau forana cap a PROGRAMABLE {id}

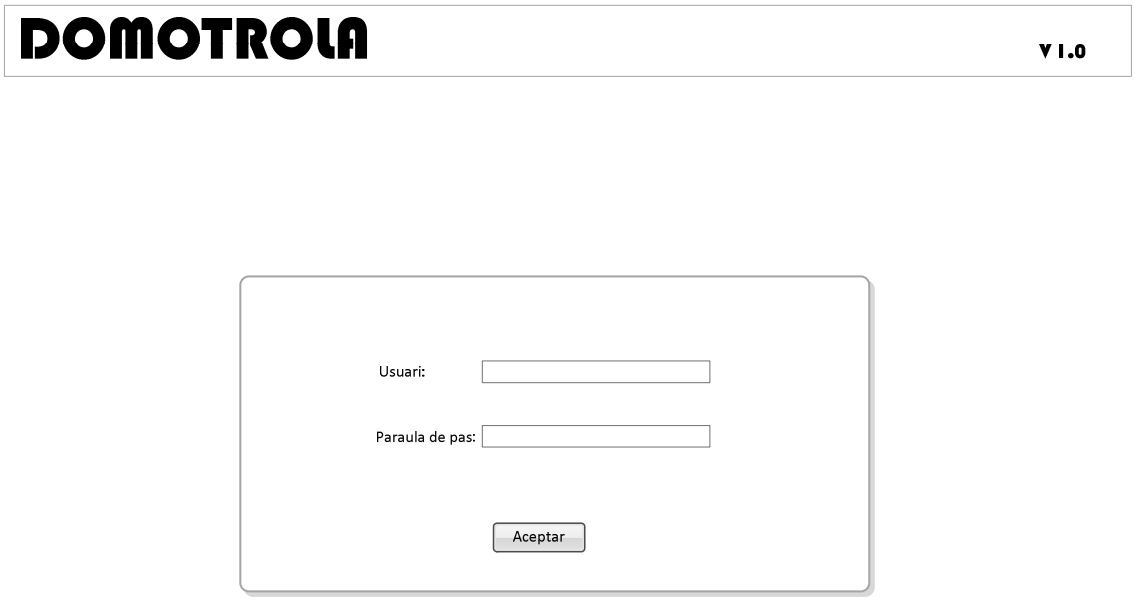
NOTA: En el disseny de la base de dades s'ha estimat oportú mantenir la taula PROGRAMA per al cas d'una futura modificació en la que es volgués disposar de més d'un programa. En aquest cas el camp *estat* indicaria quin es el programa actiu. Un exemple seria si es vol disposar d'una programació diferent per a l'estiu i per al hivern.

4.3. Disseny de la interfície d'usuari

A l'hora de realitzar el disseny de les interfícies s'ha optat per fer un disseny el més senzill possible. Ates que la aplicació no ha de atraure l'atenció de cap client no té la necessitat d'un disseny colorit, com tampoc de imatges o animacions. Per tant ens hem de centrar en un disseny funcional. L'objectiu és aconseguir una interfície d'usuari coherent.

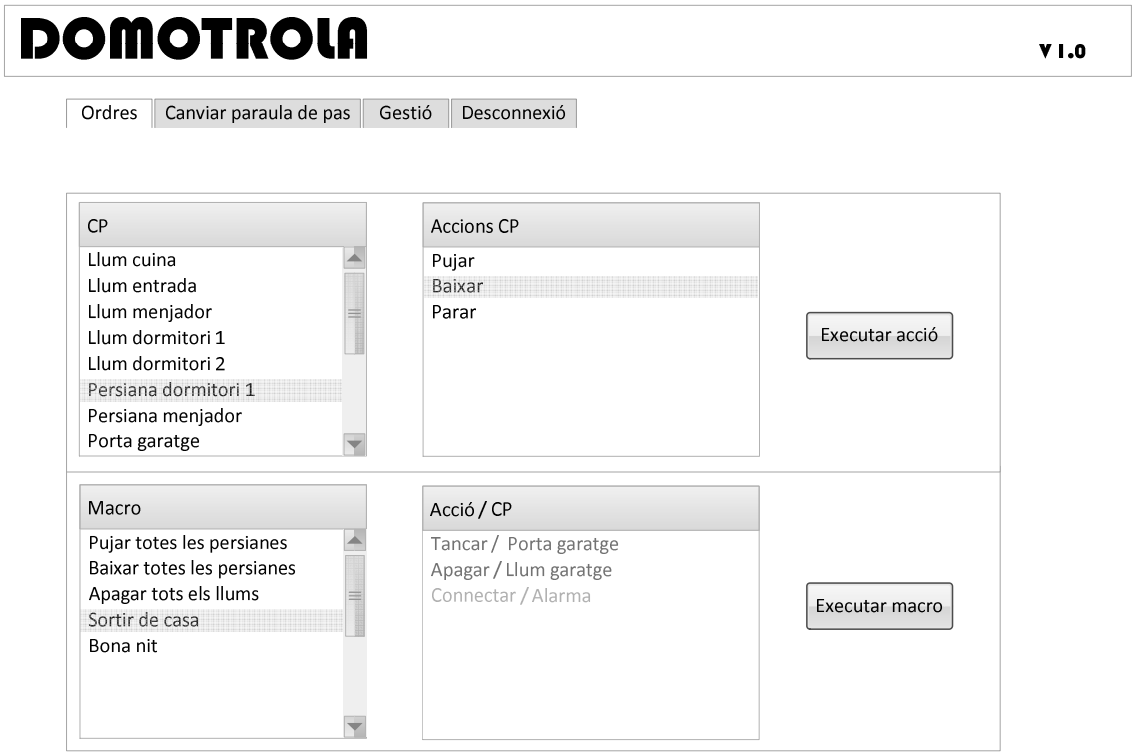
A continuació es presenten alguns prototips de pantalles.

Pantalla de connexió:



Il·lustració 16 - Interfície de connexió

Pantalla d'ordres

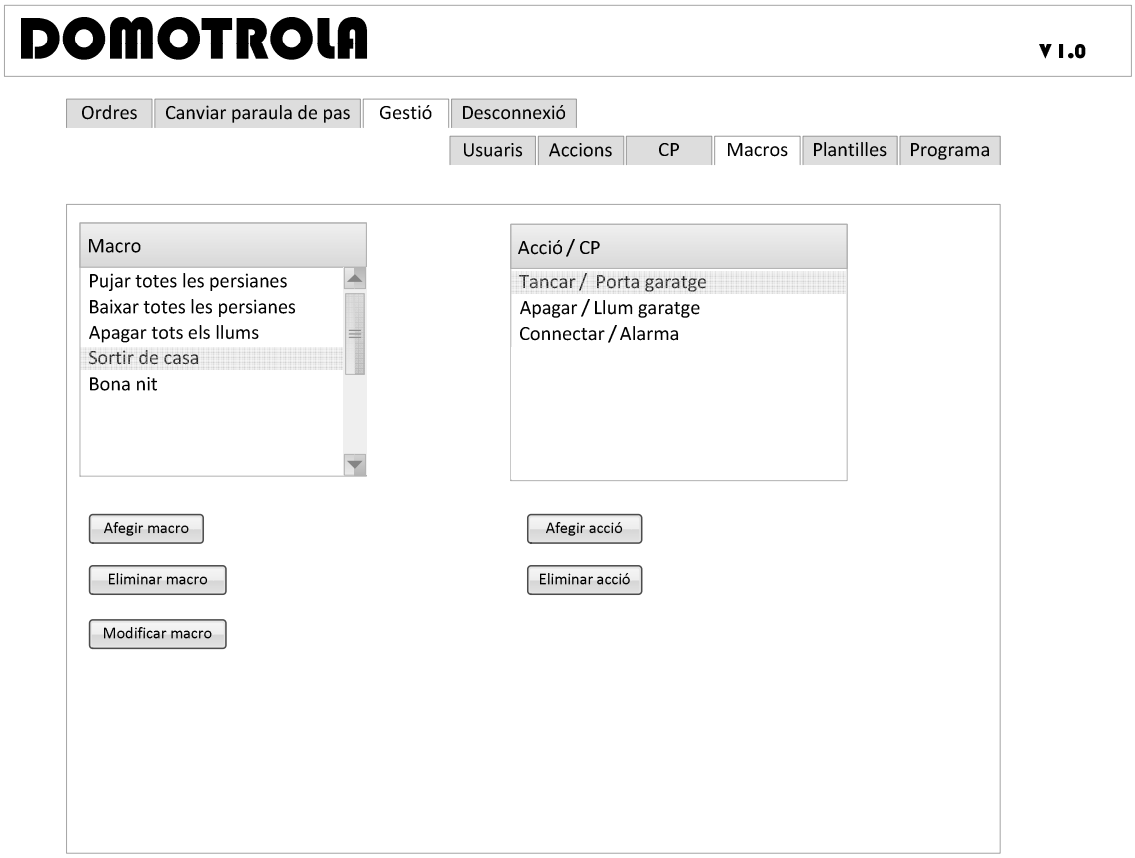


Il·lustració 17 - Interfície d'Ordres

Pantalla gestió d'usuaris

Il·lustració 18 - Interfície Gestió d'usuaris

Pantalla gestió de Macros



Il·lustració 19 - Interfície Gestió de Macros

5. Implementació

5.1. Eines i tècniques utilitzades

Per a executar l'aplicació es necessari el següent programari:

- Navegador web que permeti l'execució de codi JavaScript.
- Servidor web i d'aplicacions Tomcat versió 7.0.30.
- Servidor de base de dades MySQL 5.5.
- Plataforma Java J2SE 7 update 9 (JDK)

Pel desenvolupament e implementació he escollit les següents eines:

- Eclipse Java EE Juno (4.2) SR1. IDE opensource.
- Bastiment Struts2 (2.3.4.1).
- Bastiment Hibernate (4.1.8).
- Llibreria JQuery (1.8.3)

5.2. Estructura de l'aplicació

Les aplicacions web basades en J2EE tenen una estructura característica i aquest projecte està estructurat d'acord amb el model J2EE, que permet el seu desplegament a un fitxer web (WAR).

L'estructura té dues carpetes principals, la carpeta src on organitzem totes les classes java del projecte i la carpeta WebContent que és la base de l'estructura del fitxer war.

Dintre del WebContent trobem els elements públics de l'aplicació com les JSP o les imatges, i una carpeta WEB-INF on trobem els elements no públics de l'aplicació com la carpeta lib amb les llibreries necessàries o la carpeta classes amb les classes de Java compilades.

La estructura i contingut de la carpeta src és bastant lliure. En el nostre projecte hem organitzat les classes Java per funcionalitats seguint el patró MVC. Tenim una carpeta control on trobem totes les classes de control. Un altre carpeta model on trobem totes les classes del model. I una tercera carpeta interceptors on trobem les classes interceptors corresponents al bastiment Struts2. En src també trobem els arxius de configuració dels bastiments Struts2 i Hibernate. En el desplegament del projecte totes les classes Java són compilades i desades a la carpeta privada WEB-INF/classes seguint la mateixa estructura de directoris que a src.

6. Conclusió

Vaig començar aquest projecte amb els únics coneixements adquirits durant la carrera. En el moment de seleccionar l'àrea del TFC, pensava que amb aquests coneixements i una mica més d'informació en tenia prou, però tot just vaig començar em vaig adonar que només coneixia una minúscula part del mon Java.

Aquest projecte m'ha permès conèixer i assolir noves eines i tecnologies com Struts2, Hibernate, JSP, EJB, JQuery, DAO, Tomcat, MySQL i MVC. Aquestes tecnologies juntament amb els coneixements adquirits a la carrera es materialitzen en el present projecte.

La conclusió personal és bastant positiva. Encara que la fase de implementació no s'ha completat al cent per cent, es pot afirmar que els objectius principals del projecte s'han complert. Com per exemple la recerca de informació i aplicació de les noves eines i tecnologies esmentades. Penso que en la part de implementació realitzada, queda prou demostrat l'assoliment d'aquestes tecnologies. Si ben és cert que professionalment encara he d'aprofundir més en el seu ús. L'assoliment de coneixements adquirits durant la carrera, queda sobradament demostrat.

De cara a futurs projectes, en la planificació inicial se li haurà de donar més importància a l'ús de noves tecnologies i a la seva corba d'aprenentatge.

7. Glossari

- **Controlador de maquinari de dispositiu**. Al igual que en qualsevol perifèric el controlador és la part electrònica de control del dispositiu.
- **Controlador de programari**. Fa referència al objecte de software associat a un controlador de maquinari de dispositiu en concret.
- **CP**. Controlador de programari.
- **Dispositiu**. És la part mecànica del perifèric. En el nostre cas pot ser un llum, una persiana, una caldera, etc.
- **Habitatge domotitzat**. Habitatge que disposa d'un conjunt de sistemes o dispositius mecànics que l'automatitzen. Aportant serveis de comoditat i benestar.
- **Macro**. Conjunt d'accions de diferents CP que s'executen seqüencialment amb una única ordre.
- **Usuari**. Persona física que fa ús del sistema.
- **Usuari administrador**. Usuari que té drets afegits per administrar la base de dades del sistema.

8. Bibliografia

ANTONI PÉREZ. Treball de final de carrera. UOC.

BENET CAMPDERRICH. Enginyeria del programari. UOC.

JAUME SISTAC. Base de dades I i II. UOC.

CARLOS CASADO, JORDI COLELL, LOÏC MARTÍNEZ. Tecnologies i eines per al desenvolupament web. UOC.

JORDI SANCHEZ. Programació web avançada. UOC.

ERIC VAN DER VLIST, DANNY AYERS, ERIK BRUCHEZ, JOE FAWCETT, ALESSANDRO VERNET. Programación web 2.0. Anaya Multimedia 2008.

CESAR PÉREZ. MySQL para Windows y Linux. Ra-Ma 2004.

GAVIN KING, CHRISTIAN BAUER, MAX RYDAHL ANDERSEN, EMMANUEL BERNARD, STEVE EBERSOLE. Tutorial básico de Hibernate. http://www.davidmarco.es/tutoriales/hibernate-reference/

GAVIN KING, CHRISTIAN BAUER, MAX RYDAHL ANDERSEN, EMMANUEL BERNARD, STEVE EBERSOLE, HARDY FERENTSCHIK. Hibernate Core Reference Manual. http://www.hibernate.org/docs.

KOUSHIK KOTHAGAL. Hibernate. http://javabrains.koushik.org/p/hibernate.html

KOUSHIK KOTHAGAL. Struts 2. http://javabrains.koushik.org/p/struts-2.html

CHRIS HULBERT. Java & Struts2 & Spring & Hibernate & Eclipse Tutorial. http://es.scribd.com/doc/25244173/Java-Struts-Spring-Hibernate-Tutorial-github-com-chrishulbert-JavaTutorial.

APACHE SOFTWARE FOUNDATION. Apache Struts 2 documentation. http://struts.apache.org/2.3.8/docs/guides.html

JSEGOVIAD. Manual de Struts 2. http://code.google.com/p/prog2-60g-20121/downloads/detail?name=Manual-Struts2-Espanol.pdf

9. Annexos

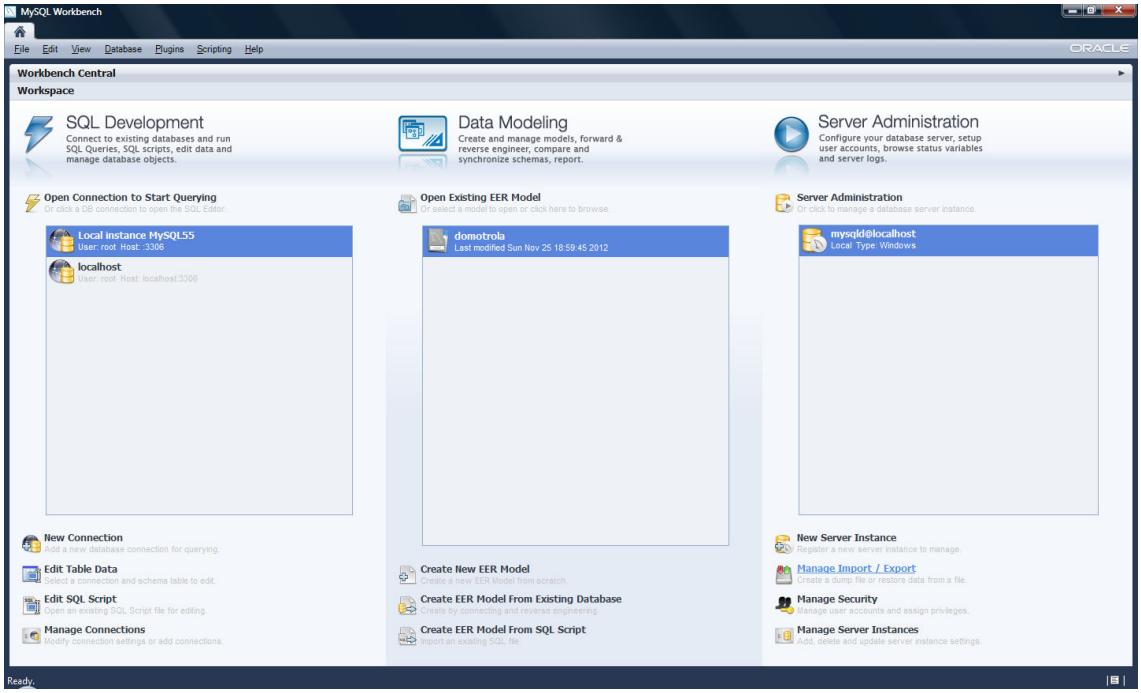
9.1. Manual de instal·lació

9.1.1. Requisits

Per a poder executar l'aplicació com a servidor, necessitem tenir instal·lats el servidor d'aplicacions Tomcat 7, el servidor de base de dades MySQL 5.5 i la plataforma Java J2SE 7 update 9 (JDK). La instal·lació d'aquestes aplicacions estan fora de l'abast d'aquest manual, però en el cas del projecte Domotrola funcionaran amb la configuració estàndard.

9.1.2. Configurar base de dades

La estructura de la base de dades necessària per al funcionament de l'aplicació la tenim al fitxer domotrola.sql. Aquesta estructura la podem carregar directament des de la consola de MySQL, les sentencies les tenim detallades al punt 9.2. Però és molt més fàcil amb alguna eina com per exemple "*MySQL Workbench*" proporcionada amb la instal·lació del servidor. Quan obrim l'aplicació ens trobem amb la pantalla següent:

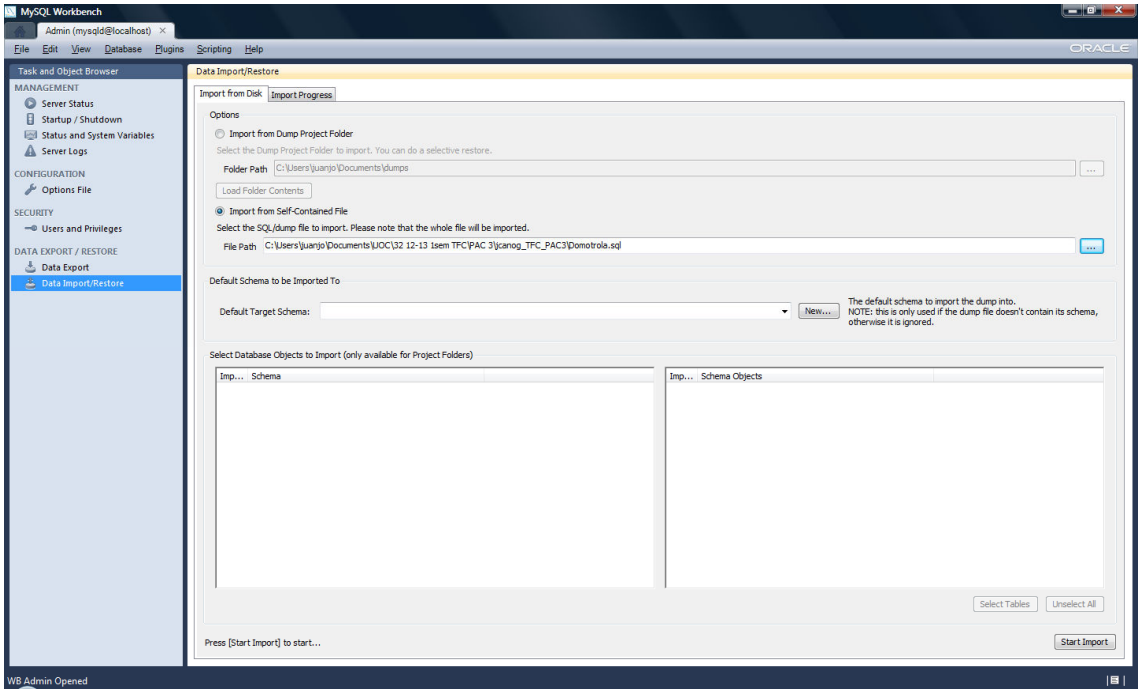


Il·lustració 20 - Configurar base de dades (1)

Aquí hem de seleccionar la opció de “Manage import / export” i accedirem a la següent pantalla. En aquesta segona pantalla hem de fer els següents passos:

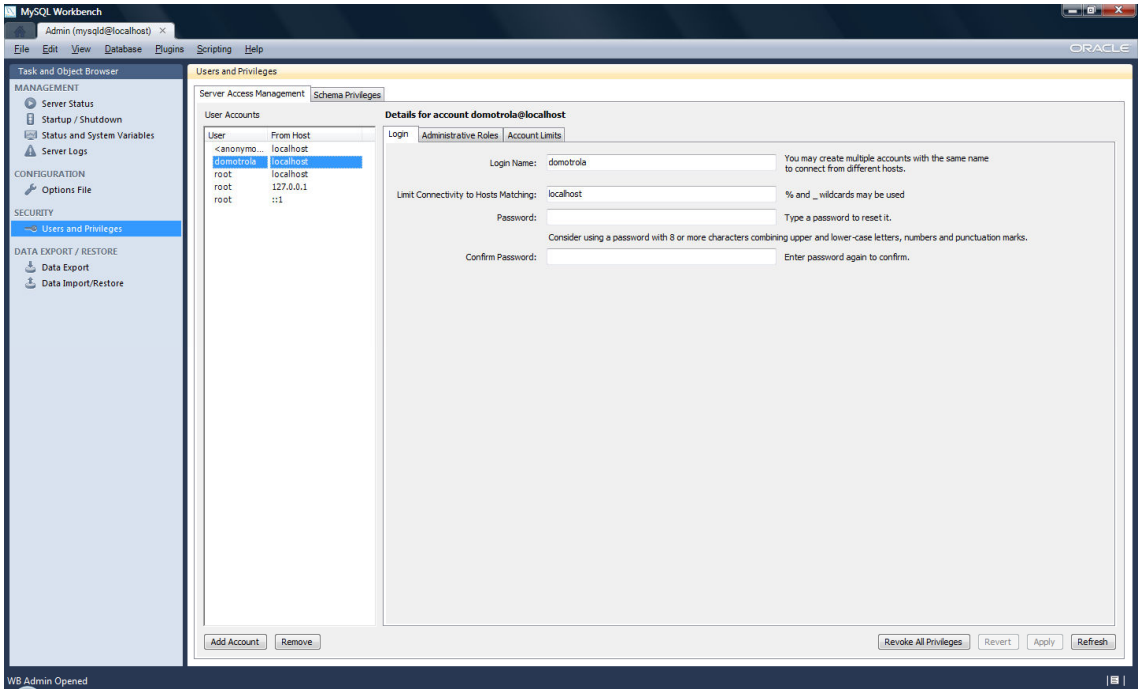
- Seleccionar “Data Import/Restore” en el menú de l’esquerra.
- Dintre de la pestanya “Import from disk” la opció “Import from self-Contained file”.
- En la casella “File Path” hem de posar la ruta i el nom del fitxer sql.

Amb tot això obtindrem una pantalla com la següent on farem clic al botó “Start Import”:



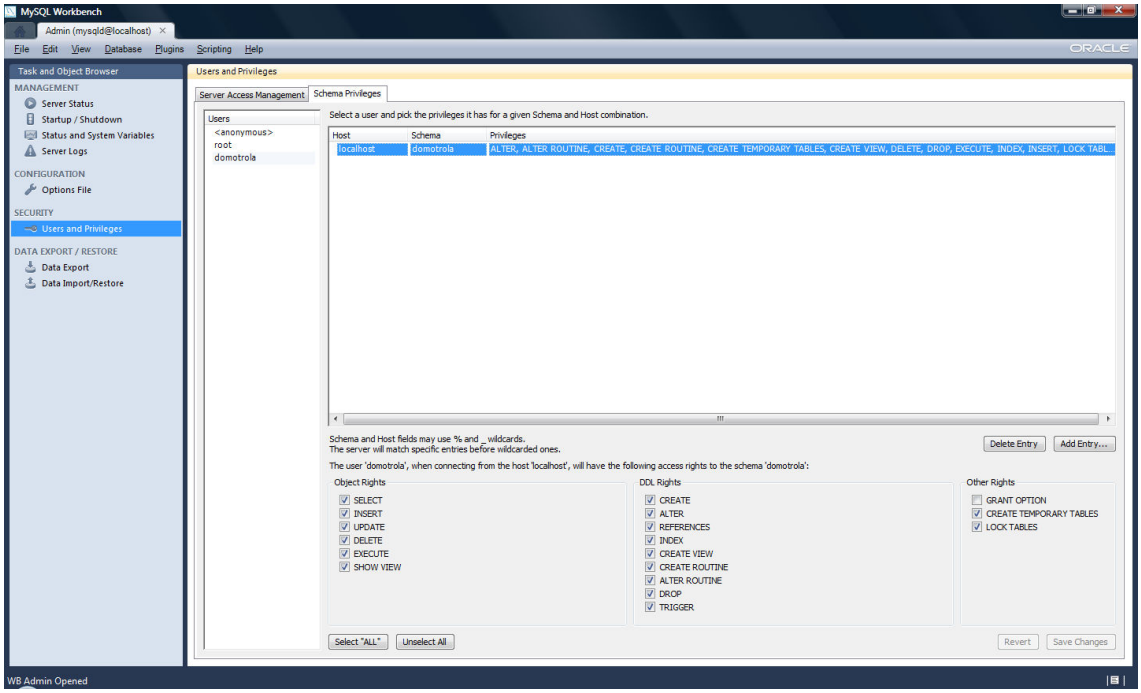
Il·lustració 21 - Configurar base de dades (2).

El següent pas és crear l'usuari "domotrola" amb password "domotrola" i donar-li tots els drets necessaris per manipular la nostra la base de dades. Per fer això tenim en el menú de l'esquerra la opció "Users and Privileges" on podem crear una compta:



Il·lustració 22 - Configurar base de dades (3)

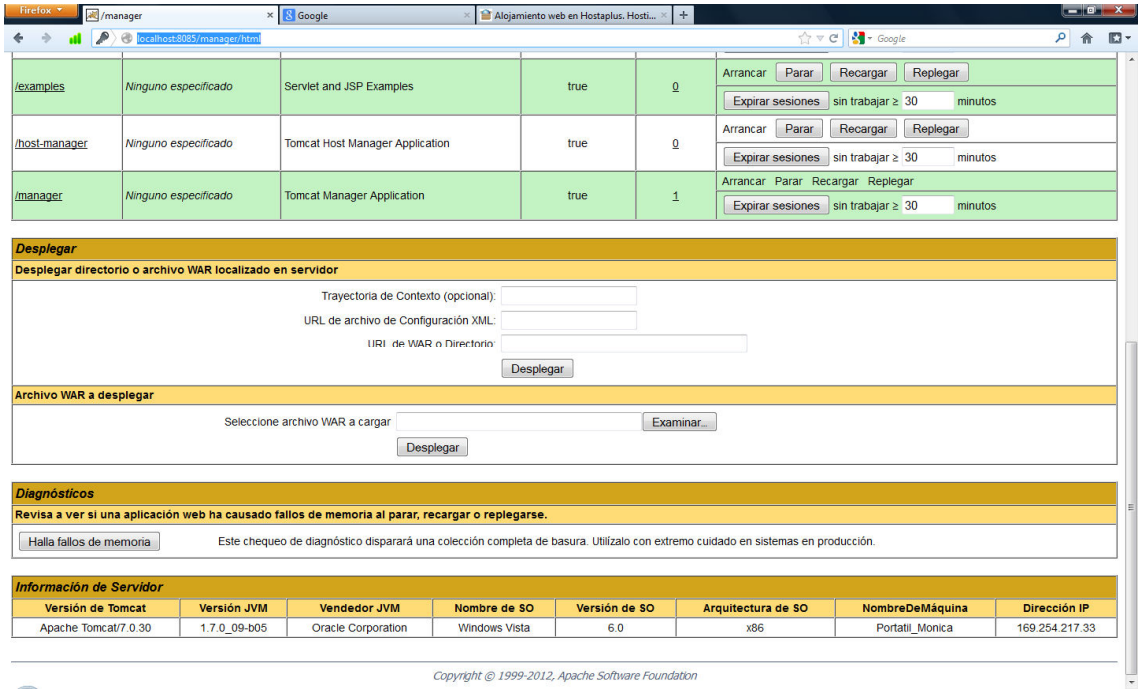
I donar-li els drets necessaris tal i com es mostra a continuació:



Il·lustració 23 - Configurar base de dades (4)

9.1.3. Desplegament de l'aplicació en el servidor d'aplicacions

La versió 7 del Tomcat ens proporciona una eina molt senzilla i útil per fer els desplegament d'una aplicació web des de un fitxer tipus war. Per desplegar la nostra aplicació accedirem a l'enllaç http://localhost:8080/manager/html on trobarem la pàgina següent:



Il·lustració 24 - Desplegament de l'aplicació

Dintre de l'apartat "*Archivo WAR a desplegar*" cliquem a examinar per cercar i seleccionar el nostre arxiu domotrola.war. Després cliquem a desplegar i llestos. A partir d'aquest moment podem accedir a la nostra aplicació des de l'enllaç http://localhost:8080/domotrola/. La instal·lació crea un usuari administrador per defecte amb nom d'usuari "admin" i clau "admin".

9.2. Sentències SQL

```
CREATE DATABASE IF NOT EXISTS domotrola;
USE domotrola;

CREATE TABLE accions (
        id int(11) NOT NULL AUTO_INCREMENT,
        description varchar(20) NOT NULL,
        PRIMARY KEY (id)
);

CREATE TABLE accionscp (
        cp int(11) NOT NULL,
        accio int(11) NOT NULL,
        PRIMARY KEY (cp,accio),
        KEY cp_idx (cp),
        KEY accio_idx (accio),
        CONSTRAINT FK_accionscp_accio FOREIGN KEY (accio)
        REFERENCES accions (id) ON DELETE NO ACTION ON UPDATE NO ACTION,
        CONSTRAINT FK_accionscp_cp FOREIGN KEY (cp) REFERENCES programable (id) ON DELETE
        NO ACTION ON UPDATE NO ACTION
```

```
);

CREATE TABLE accionsplantilla (
        plantilla int(11) NOT NULL,
        accio int(11) NOT NULL,
        PRIMARY KEY (plantilla,accio),
        KEY plantilla_idx (plantilla),
        KEY accioPlantilla_idx (accio),
        CONSTRAINT FK_accionsPlantilla_accio FOREIGN KEY (accio) REFERENCES accions (id)
        ON DELETE NO ACTION ON UPDATE NO ACTION,
        CONSTRAINT FK_accionsPlantilla_plantilla FOREIGN KEY (plantilla) REFERENCES
        plantilles (id) ON DELETE NO ACTION ON UPDATE NO ACTION
);

CREATE TABLE cpmacro (
        macro int(11) NOT NULL,
        cp int(11) NOT NULL,
        accio int(11) NOT NULL,
        PRIMARY KEY (macro,cp,accio),
        KEY FK_cpMacro_macro_idx (macro),
        KEY `FK_cpMacro_cp_idx` (`cp`,`accio`),
        CONSTRAINT FK_cpMacro_accio_cp FOREIGN KEY (cp, accio) REFERENCES accionscp (cp,
        accio) ON DELETE NO ACTION ON UPDATE NO ACTION,
        CONSTRAINT FK_cpMacro_macro FOREIGN KEY (macro) REFERENCES programable (id) ON
        DELETE NO ACTION ON UPDATE NO ACTION
);

CREATE TABLE plantilles (
        id int(11) NOT NULL,
        description varchar(45) NOT NULL,
        description2 text,
        PRIMARY KEY (id)
);

CREATE TABLE programa (
        id int(11) NOT NULL,
        estat int(11) NOT NULL DEFAULT '0',
        descripcio varchar(45) NOT NULL DEFAULT 'Descripcio del programa',
        PRIMARY KEY (id)
);

CREATE TABLE programable (
        id int(11) NOT NULL AUTO_INCREMENT,
        description varchar(45) NOT NULL,
        type varchar(5) NOT NULL,
        ip varchar(16) NOT NULL,
        description2 text,
        PRIMARY KEY (id),
        KEY tipus (type)
);

CREATE TABLE programacio (
        programa int(11) NOT NULL,
        programable int(11) NOT NULL,
        accio int(11) NOT NULL,
        diaHora datetime NOT NULL,
        PRIMARY KEY (programa,programable,accio),
        KEY FK_programacio_programa_idx (programa),
        KEY FK_programacio_programable_idx (programable),
        KEY FK_programacio_accio_idx (accio),
        CONSTRAINT FK_programacio_accio FOREIGN KEY (accio) REFERENCES accions (id) ON
        DELETE NO ACTION ON UPDATE NO ACTION,
```

```
        CONSTRAINT FK_programacio_programa FOREIGN KEY (programa) REFERENCES programa
        (id) ON DELETE NO ACTION ON UPDATE NO ACTION,
        CONSTRAINT FK_programacio_programable FOREIGN KEY (programable) REFERENCES
        programable (id) ON DELETE NO ACTION ON UPDATE NO ACTION
);

CREATE TABLE usuaris (
        userName varchar(20) NOT NULL,
        type varchar(5) NOT NULL DEFAULT 'basic',
        password varchar(20) NOT NULL,
        description text,
        PRIMARY KEY (userName)
);

LOCK TABLES usuaris WRITE;
INSERT INTO usuaris VALUES ('admin','admin','admin','Administrador principal');
UNLOCK TABLES;
```