**DADES DEL PROJECTE**

*Títol del Projecte: GWAL – Gestor Web d'Activitats d'interès cultural, Lúdic i esportiu*

*Nom de l'estudiant:Iván Gómez i Calero*
*Titulació: Enginyeria en Informàtica*
*Crèdits:37.5*
*Director/Ponent: Lluís Vila i Grabulosa*
*Departament:LSI*

**MEMBRES DEL TRIBUNAL** *(nom i signatura)*

*President: Ricard Gavaldà i Mestre*

*Vocal: Carme Martin i Escofet*

*Secretari: Lluís Vila i Grabulosa*

**QUALIFICACIÓ**

*Qualificació numèrica:*
*Qualificació descriptiva:*

*Data:*

***Títol: GWAL – Gestor Web d'Activitats d'interès cultural, Lúdic i esportiu***

***Volum: 1***

***Alumne: Iván Gómez i Calero***

***Director/Ponent: Lluís Vila i Grabulosa***

***Departament: LSI***

# Índex

# 1 Introducció

## 1.1 Organització de la memòria

En aquesta memòria es presenten totes les etapes per les quals ha passat el desenvolupament del projecte en l'ordre en què aquestes han estat dutes a terme. A continuació s'indica una breu descripció de cadascun dels apartats d'aquesta documentació.

- Capítol 1: **Introducció**. En aquest capítol s'expliquen els processos i plantejaments anteriors al desenvolupament.
- Capítol 2: **Anàlisi funcional**. Aquest apartat inclou l'anàlisi de requeriments que determina quin sistema cal construir, és a dir, quines funcionalitats i quines característiques generals ha de tenir l'aplicació, i l'especificació, documentació que determina què farà el sistema.
- Capítol 3: **Disseny**. En aquest capítol es defineix la manera en què el sistema farà el que s'ha especificat en l'apartat anterior.
- Capítol 4: **Implementació**. En aquesta part es presenta la fase de implementació on es descriu com s'ha dut a terme la codificació del sistema i quins són els aspectes importants en aquesta etapa del procés. Addicionalment, s'inclouen les probes sobre el sistema que es realitza en paral·lel.
- Capítol 5: **Manual d'usuari**. En aquest apartat es mostra el manual d'usuari, orientat als diferents perfils que utilitzaran l'aplicació.
- Capítol 6: **Planificació**. El sisè capítol inclou la descripció dels recursos assignats, l'estudi de la planificació i l'estudi econòmic del projecte.
- Capítol 7: **Conclusions**. En aquest apartat s'expliquen les conclusions obtingudes i les possibles línies de futur del projecte.
- Capítol 8: **Bibliografia**. Por concloure, s'inclou la bibliografia consultada.

## 1.2 Descripció del projecte

Actualment molts centres que ofereixen activitats d'interès cultural, lúdic i esportiu utilitzen un gran ventall de programari d'ofimàtica per dur a terme la seva gestió. Això comporta molts cops la disgregació de les dades i dificulta la seva gestió i cerca de forma eficaç. Per tal de contribuir a pal·liar aquesta mancança, neix el projecte GWAL.

El seu objectiu principal és dissenyar i desenvolupar una eina que permeti la gestió d'un centre lúdic que doni suport a l'administració d'activitats, recursos i usuaris considerant les darreres tecnologies de desenvolupament d'aplicacions web. Per altra banda, en el funcionament del dia a dia d'aquests tipus de centres hi intervenen moltes persones (gestors del centre, professors, usuaris,etc.). Aquest projecte vol fer que aquestes persones puguin interactuar amb el sistema cadascun d'ells obtenint unes funcionalitats pròpies a les seves necessitats. Per últim cal dir que el sistema pretén ser *genèric* per qualsevol entitat que dugui a terme activitats de caràcter lúdic, cultural i/o esportiu com és el cas de centres cívics, entitats culturals i acadèmies.

## 1.3 Objectius

Els objectius concrets del projecte es divideixen en objectius funcionals i objectius de disseny:

**Objectius funcionals del projecte:**

- Oferir una eina que tingui les funcionalitats de programació d'activitats, gestió del personal del centre, gestió de recursos, gestió d'usuaris del centre i seguiment de les activitats als centres.
- Actualment, per falta d'eines de generació automàtica de calendaris d'activitats, el personal d'administració dels centres distribueix les sessions de les diferents activitats a mà. El projecte oferirà una eina que permeti automatitzar aquest procés.
- Es pretén la unificació de gran part de les gestions dels centres a través d'un únic punt d'entrada. D'aquesta manera el centre es veurà beneficiat en treballar amb una sola aplicació, reduint el nombre de programes que actualment es fan servir.
- Es vol oferir funcionalitats diferents a cada tipus d'usuari. Per exemple: els professors podran introduir l'assistència a diferents sessions que imparteixin i així contribuir al seguiment de les activitats; i els usuaris per la seva banda podran fer preinscripcions en els cursos o mostrar les seves preferències.
- Emmagatzemar informació històrica del centre respecte les activitats i inscripcions dels usuaris.

**Objectius de disseny i desenvolupament de l'aplicació:**

- Flexibilitat: que davant de canvis de la realitat de les escoles l'esforç per adaptar el projecte sigui mínim. Amb petites modificacions sobre l'aplicació es podrien contemplar noves formes d'actuar.
- Integritat: que es garanteixi un control d'accessibilitat de les dades. Hi

haurà identificació de l'usuari per assignar el rols que cadascú tindrà i es mostraran opcions diferents segons el rol. Per a tot usuari es tindrà la seguretat de que aquest només veurà les dades que el seu rol li permeti accedir.

- Correctesa: quan l'eina estigui completa s'ha d'avaluar fins a quin punt el programa satisfà les especificacions inicials. L'aplicació està dissenyada com a eina genèrica, però per assegurar la seva correctesa hem pres com a prova pilot l'escola de ball "Balliball" i l'aplicació final s'adaptarà a les característiques concretes d'aquesta escola.
- Seguretat: l'aplicació s'adaptarà i seguirà la llei de protecció de dades. Amb l'existència de rols i traces del sistema es contemplaran possibles intents d'intrusió i manipulació.
- Portabilitat: es pretén dissenyar l'aplicació per tal que pugui funcionar sota qualsevol plataforma gràcies a les tecnologies amb les que es vol programar.
- Usabilitat: que l'eina resulti fàcil d'utilitzar de cara a l'usuari final i que els resultats es mostrin de manera que sigui senzill interpretar-los.

## 1.4 Agraïments

Vull agrair a la Marta Cardus la seva atenció personalitzada per a l'etapa d'anàlisi de requisits i a les persones dels centres cívics on vaig treballar durant 6 anys.

Vull agrair els consells del meu director de projecte i l'orientació que m'ha ofert al llarg del procés.

Vull agrair a tots els familiars i amics que han estat al meu costat mentre feia aquest projecte i han tingut la paciència d'aguantar els nervis que he portat aquests mesos.

I per últim vull donar les gràcies a la meva parella pel seu optimisme i ànims constants

# 2 Anàlisi funcional

## 2.1 Anàlisi de requeriments

Els objectius principals de l'anàlisi de requeriments són els següents:

- Definir el conjunt de requeriments que haurà de tenir el sistema per tal de satisfer les necessitats de l'usuari final.
- Detallar al màxim les funcionalitats del sistema amb l'objectiu de facilitar una especificació fiable i un disseny adequat.
- Servir de document d'acord amb el client.

A continuació veurem la llista de requeriments que ha de satisfer el projecte que ens ocupa. L'obtenció d'aquests requeriments s'ha dut a terme a través de vàries entrevistes amb el nostre interlocutor, la propietària de l'acadèmia *Ball i Ball* Sra. Marta Cardus. Per tal que l'eina fos genèrica s'ha aprofitat l'experiència docent de l'estudiant com a formador tallerista del centre cívic Can Castelló i del casal d'avis Can Fàbregues, ambdós situats al districte de Sarrià-Sant Gervasi a Barcelona, així com del centre lúdic Puig Coca situat a Esplugues de Llobregat.

S'ha fet una divisió en funció dels tipus de requeriments. Primer parlarem dels **requeriments funcionals** i més endavant dels **requeriments no funcionals**.

### 2.1.1 Requeriments funcionals

En aquest apartat, enumerarem les funcionalitats del sistema. Les hem dividit en mòduls amb funcionalitats de característiques que es detallen a continuació:

- Gestió d'activitats, temes i especialitats
- Gestió d'usuaris
- Gestió de fulls d'interès
- Gestió d'inscripcions
- Gestió de l'assistència
- Gestió de recursos
- Configuració del sistema

## Gestió d'activitats, temes i especialitats

El sistema ha de permetre donar d'alta activitats, temes i especialitats així com operacions associades a modificacions d'aquestes entitats o la seva eliminació. Després de realitzar acuradament l'anàlisi de requisits s'ha decidit que per cobrir les necessitats de la major part dels centres lúdics cal que el sistema diferenciï entre tres tipus diferents d'activitat:

- **Tallers**: són totes aquelles activitats que impliquin docència de forma continuada en el temps. Exemples de tallers poden ser un curs de fotografia o un curs de dansa del ventre. Dels tallers es vol conèixer el nombre de sessions que tindran i el seu horari, el nivell de dificultat que comporten, la periodicitat de les sessions, la data d'inici i finalització de les seves inscripcions i òbviament la data d'inici del taller. El taller es un tipus de entitat que ha de permetre fer un seguiment de l'assistència dels seus usuaris si el centre ho precisa.

- **Event:** són les activitats puntuals o de caràcter esporàdic amb durada variable que vulgui realitzar el centre. Un exemple d'event seria una visita cultural programada o un concert. En un principi no es considera necessari que un event tingui inscripcions ni seguiment però el sistema es dissenyarà de manera escalable per a què en un futur es pogués afegir fàcilment aquesta funcionalitat en cas de que sorgís la necessitat.

- **Concursos:** són activitats en les quals es duu a terme una competició que pot tenir o no un premi associat. No es poden considerar ni un event ni un taller degut a que té una sèrie de característiques pròpies. Un exemple de concurs seria un concurs de fotografia o dibuix.

Tota activitat que es dugui a terme en un centre ha de tenir un tema associat que conté la informació docent sobre el que s'imparteix en el curs. Així, en el cas de l'exemple del curs de fotografia un centre lúdic podria oferir dos temes diferents: "Fotografia Digital" i "Fotografia Analògica" en cas de contenir temaris molt amplis o "Fotografia Digital Blanc i negre" i "Revelat fotogràfic" en cas de contenir temaris més concrets. Tot tema té un temari que s'imparteix durant el curs en una sèrie de sessions.

A més, un tema pertany a una especialitat, que alhora és d'una comunitat. En el cas del curs de fotografia la especialitat podria ser "Fotografia Artística" i la

"Comunitat de Fotògrafs del Vallès" ser la comunitat experta en aquesta especialitat.

Amb la informació subministrada pel personal del centre en donar d'alta un taller el sistema haurà de planificar les sessions en que aquest es durà a terme en funció de la periodicitat i la data d'inici. A més, per establir aquesta planificació caldrà saber quin és el calendari de dies festius. Una sessió és la representació única d'un dia de docència de taller i per tant té una data concreta, es realitza en un aula concreta i correspon a una franja horària entre una hora d'inici i una hora de finalització.

La planificació inicial del taller generarà sessions previstes. Això permetrà que diferents tallers es programin en el temps i les seves sessions previstes es puguin encavallar provocant col·lisions. Els tallers que tinguin sessions previstes estaran en l'estat previst i per tant no permetran inscripcions. Per tal de permetre inscripcions en el taller caldrà que l'administratiu del sistema passi a confirmades les sessions previstes i a més que es superi la data d'inici d'inscripcions i el taller passi a l'estat '*obert'*. D'igual manera, no es permetran inscripcions quan es superi la data de finalització d'inscripcions del taller. El sistema no permet l'encavallament de sessions confirmades responent a la realitat de l'espai i el temps. Aquest funcionament ofereix als administradors del centre la possibilitat de distribuir els tallers en el temps i veure'n els resultats, i un cop decidit quins tallers es faran finalment confirmar-los.

El sistema haurà de ser capaç de diferenciar la informació que ha de mostrar depenent del rol de l'usuari. El rol més baix és el del lector de la web que és considerat com usuari anònim que visita la web. Aquest tipus d'usuari podrà veure els llistats en un format diferent al que veuen el personal del centre. Tan sols aquells usuaris que s'identifiquin en el sistema a través de l'apartat d'entrada d'usuari i gaudeixin de suficients privilegis podran inscriure's a les activitats (veure l'apartat de gestió d'usuaris en aquest mateix apartat per saber quins tipus d'usuari tenen permís a aquestes funcionalitats). Per poder obtenir aquests privilegis i inscriure's a un taller obert caldrà que ompli un registre donant certa informació personal i escollint un nom d'usuari únic i una contrasenya.

Per tant, les funcionalitats que gestionarà aquest mòdul seran les següents:

- Alta, baixa i modificació d'activitats
- Alta, baixa i modificació de temes

- Alta, baixa i modificació d'especialitats
- Alta, baixa i modificació de comunitats
- Gestió de la planificació de tallers

## Gestió d'usuaris

La gestió d'usuaris inclou l'alta i baixa d'usuaris del sistema, així com la modificació de les seves dades personals i la consulta de la informació disponible en diferents formats.

Com ja s'ha comentat breument anteriorment s'implementaran una sèrie de rols que correspondran als tipus de persones que faran servir el sistema i atorgaran diferents permisos d'accés a les dades. La part d'autenticació de l'aplicació i protecció de les dades es veurà en més detall en l'apartat 4.6 d'aquesta memòria. Els rols que es dissenyaran per l' aplicació són:

- lector
- usuari
- formador
- administratiu
- administrador

El rol de **lector** representa qualsevol usuari que no estigui autenticat al sistema. Bàsicament, podrà veure tota la informació pública del lloc web:

- Consultar les activitats obertes actualment
- Consultar informació sobre els temes, les especialitats i les comunitats
- Fer un registre al sistema per obtenir el rol d'usuari

El rol d'**usuari** podrà fer les mateixes funcions que el lector i a més se n'afegiran les següents funcions pròpies:

- Pot modificar les seves dades personals
- Pot sol·licitar una contrasenya nova
- Pot introduir fulls d'interès per temes, amb la seva disponibilitat

- Pot introduir fulls d'interès per tallers confirmats i dels quals tingui un nivell d'expertesa suficient
- Pot inscriure's a un taller que estigui en estat *'obert'* o bé *'en seguiment obert'*

El rol de **formador** és una mica diferent de l'anterior. A més de les operacions que pot realitzar un lector el formador serà capaç de:

- Emplenar l'assistència de les sessions que imparteix
- Fer canvis en els temes, especialitats i comunitats. Per exemple, el centre pot demanar-li que sigui l'encarregat d'omplir el temari dels temes i sessionsTemari
- Crear i veure documents dels temes dels que és expert
- Fer reserves de material

L'usuari amb rol **administratiu**, igual que en els casos anteriors , tindrà les funcions de lector i a més haurà de poder:

- Llistar i donar d'alta activitats,Temes, Especialitats
- Planificar sessions, confirmar sessions previstes
- Gestionar inscripcions
- Veure els fulls d'interès que han enviat els usuaris
- Donar d'alta festes en el calendari
- Gestionar recursos

Finalment, el rol d'administrador podrà fer totes les funcions dels rols anteriors i a més:

- És l'encarregat de donar d'alta Formadors i administratius al sistema
- Pot modificar la configuració del sistema

## Gestió de fulls d'interès

Per tal que els centre pugui tenir una idea d'aquells temes i tallers que interessen més als seus clients, que són usuaris de l'aplicació, el sistema GWAL disposa d'una funcionalitat per gestionar fulls d'interès. Conceptualment el full d'interès es el mitjà pel qual l'usuari pot mostrar les seves preferències i la seva disponibilitat horària. Per al centre aquesta informació és molt útil a l'hora de dissenyar i/o escollir els tallers que s'ofertaran en el període docent. Hi haurà dues vies diferents per mostrar l'iterés:

- En els **fulls d'interès de temes** l'usuari indicarà d'entre un conjunt de temes disponibles quins li interessen més a l'hora de fer algun taller, així com la seva disponibilitat horària per aquell període docent
- Un cop el centre hagi gestionat aquesta informació, l'usuari podrà omplir un **full d'interès de tallers** en el qual se li mostraran els tallers previstos pel centre (i que ja tenen assignat horari) per aquell període docent i d'entre els quals l'usuari dirà a quins li interessaria apuntar-s'hi

Aquesta funcionalitat permetrà millorar l'organització del centre en quant a la previsió de demanda que tindran, ja que a través dels fulls d'interès el centre disposarà d'informació sobre:

- Quines temàtiques de tallers interessen més
- Quins horaris resulten més convenients als clients
- Quins tallers tindran més demanda, cosa que podria originar en la decisió de fer el taller en més d'un horari fins i tot assignant recursos externs que es puguin contractar

En conclusió, aquesta eina pot proveir al centre d'informació molt valuosa per ajustar-se a les necessitats del mercat i adequar l'oferta a la demanda dels clients. A més, en el cas concret d'entitats com una escola de ball, on els tallers es solen fer en grup (majoritàriament en parella), els fulls d'interès permeten fer una estimació del nombre de grups que podrien haver.

Per tant, les funcionalitats que gestionarà aquest mòdul seran les següents:

- Alta, baixa i modificació de fulls d'interès Tema

- Alta, baixa i modificació de fulls d'interès Taller
- Consulta de fulls d'interès Tema
- Consulta de fulls d'interès Taller

## Gestió d'inscripcions

El sistema GWAL serà capaç de gestionar inscripcions d'usuaris a activitats que tinguin el període d'inscripció obert. Per poder inscriure's l'usuari haurà d'estar autenticat en el sistema. D'aquesta manera, cada usuari tramitarà la seva inscripció a través del lloc web. Un cop inscrit l'usuari podrà anar al centre a deixar una paga i senyal que l'administratiu registrarà en la inscripció. En el mateix moment o posteriorment l'usuari podrà pagar la resta de la inscripció i l'administratiu modificarà la seva inscripció indicant el mètode de pagament, el lloc de pagament i la data en què s'ha dut a terme.

Un cop inscrit en un taller, l'usuari tindrà accés a tots els documents associats al taller.

Per tant, les funcionalitats que gestionarà aquest mòdul seran les següents:

- Alta i baixa de la inscripció
- Consulta i modificació de la inscripció
- Consulta de documents del Taller

## Gestió de l'assistència

Després de dur a terme una sessió d'un taller, el formador haurà d'introduir al sistema la llista d'assistència dels usuaris a la sessió. Per tant, el formador caldrà que s'autentiqui a l'eina amb el seu rol de formador, cerqui el taller que ha impartit i la sessió d'aquell dia i introduexi l'assistència. En donar d'alta les assistències el formador haurà d'indicar quins assistents són usuaris inscrits i quins són convidats

si n'hi ha. En el que respecta a la sessió el formador ha de poder introduir el temari que s'ha realitzat en la sessió (que pot correspondre o no amb el que s'havia de donar). En cas de que la sessió no s'hagi pogut dur a terme es podrà indicar el motiu pel qual no s'ha pogut realitzar. El sistema calcularà llavors el número d'assistents i ho emmagatzemarà al sistema. Més endavant, si s'escau, es podran fer modificacions de tota la informació introduïda.

Cal remarcar que el sistema ha de ser prou flexible per tenir en compte que el formador que imparteixi aquella sessió pot no ser el formador que té assignat el taller al qual pertany la sessió.

Per tant, en aquest cas les funcionalitats que gestionarà aquest mòdul seran les d' alta, modificació i baixa de l'assistència, que de forma implícita tindran en compte les substitucions de formadors.

## Gestió de recursos

En els centres hi ha diferents tipus de recursos i per això el sistema GWAL emmagatzemarà informació relativa a tots ells. Així, els tipus de recursos de què es disposarà seran:

- **Consumibles**, dels quals ens interessarà conèixer el preu i la quantitat
- **Aules**, de les quals voldrem saber si són pròpies del centre o externes, i en aquest últim cas caldrà saber la seva adreça, el telèfon i el fax
- **Materials**, dels quals ens interessa conèixer la quantitat

Pel que fa a les aules, caldrà saber exactament a quin **local** del centre s'ubiquen i per a cada local caldrà conèixer el nom, l'adreça, el telèfon, el fax i saber si és la seu del centre. En cas de que l'organigrama dels locals de l'empresa que adquireixi l'aplicació sigui més complex podrà atendre les seves necessitats incloent la informació en el propi nom del local.

És molt possible que un formador o administratiu necessiti fer una reserva d'un material del centre per a una de les sessions. Un exemple d'aquest cas seria reservar l'únic projector de què es disposa. Per això, el sistema haurà de poder gestionar **reserves de materials** per a una sessió concreta de tal manera que es sap en tot moment qui ha realitzat la reserva.

Per tant, les funcionalitats que gestionarà aquest mòdul seran les següents:

- Alta, modificació i baixa de recursos
- Alta i baixa de reserves de material
- Alta, modificació i baixa de locals

## Configuració del sistema

Aquest mòdul engloba totes les funcionalitats que permeten ajustar l'aplicació el millor possible a les característiques del centre. L'accés a aquestes

funcionalitats que es detallen a continuació està reservada a l'administrador del sistema:

- **Donar d'alta usuaris** amb permisos de tipus formador o administratiu al sistema per a què aquests puguin realitzar les seves tasques de gestió específiques a l'aplicació. Inicialment l'aplicació quan s'instal·li al servidor d'aplicacions vindrà amb **dos** comptes d'usuari ja **creats per defecte**, un de tipus administrador per permetre administrar l'aplicació i crear nous rols, i un altre de usuari testejador per tal de poder veure a la web pública els canvis en la configuració que l'administrador introdueix en les preferències.

- El sistema disposa d'una **configuració per defecte** i l'administrador serà capaç de modificar-la. Ens referim a paràmetres per defecte en donar d'alta les diferents entitats objecte que tindrà l'aplicació. Això és pot il·lustrar amb el següent exemple: una entitat que imparteix cursos d'ofimàtica per a adults en estat d'atur sap que els seus tallers sempre tindran 15 alumnes perquè tenen sempre 15 ordinadors a cada aula. L'administratiu cada cop que doni d'alta un taller haurà d'introduir el mateix nombre en el camp de text "capacitat". Aquest procés se'l pot estalviar si l'administrador indica prèviament que el camp capacitat corresponent a l'entitat taller en la configuració per defecte de l'aplicació és 15. A partir de llavors, cada cop que es vulgui donar d'alta un taller, aquesta capacitat estarà ja fixada en la fitxa de creació del taller. Dissenyant l'aplicació per que permeti realitzar canvis en la configuració l'eina resultarà més usable pel personal del centre.

  Per tant, les funcionalitats que gestionarà aquest mòdul seran les següents:

  - Alta, modificació i baixa d'usuaris
  - Consulta i modificació de la configuració del sistema

### 2.1.2 Requeriments no funcionals

Els requeriments no funcionals defineixen les qualitats generals que ha de tenir el sistema per tal de realitzar la seva funció. Es poden classificar d'acord al motiu del requisit. A continuació es mostra una relació dels requisits no funcionals classificats per tipus.

#### Requeriments d'accés

- El sistema GWAL serà una aplicació web, és a dir, l'usuari executarà l'aplicació i accedirà a les seves funcionalitats a través d'un navegador.
- El sistema ha de permetre la gestió d'usuaris, ja que la informació accessible per a un formador o per l'administrador és diferent a la que pot consultar qualsevol usuari de l'aplicació. Aquesta gestió, que es durà a terme per l'administrador del sistema, ha de permetre la creació i l'esborrat d'usuaris i la gestió de contrasenyes dels usuaris donats d'alta a l'aplicació.

Com a conseqüència de la necessitat de satisfer els requisits no funcionals d'accés a l'aplicació per part de diferents tipus d'usuaris, apareixen un conjunt de noves funcionalitats que es descriuen a continuació. Aquestes funcionalitats formaran part de la gestió d'usuaris que durà a terme l'administrador de l'aplicació. Les noves funcionalitats que es contemplen ara són les següents:

-

#### Requeriments de la interfície

- El sistema ha d'oferir una interfície amigable, intuïtiva, fàcil d'usar i ben estructurada, ja que el sistema està orientat a usuaris que poden no estar habituats al camp de la informàtica.

#### Requeriments de rendiment

- El temps de resposta del sistema a les peticions de l'usuari ha de ser imperceptible. Cal optimitzar al màxim la interacció amb la base de dades.

### Requeriments legals

- El sistema ha d'assegurar la privacitat de les dades introduïdes pels usuaris, proporcionant un accés segur a l'aplicació.

### Requeriments de cost

- El sistema ha de contemplar tots i cadascun dels requeriments amb un cost mínim en els recursos emprats. És a dir, totes les eines utilitzades per implementar el sistema han de tenir un cost mínim.

### Requeriments de qualitat del software

- EFICIENT, que utilitzi el mínim nombre de recursos.
- FLEXIBLE, que sigui fàcil de modificar.
- ÍNTEGRE, que el software sigui segur, que respecti la privacitat i la integritat de les dades.
- MANTENIBLE, que sigui fàcil localitzar errors i solucionar-los.
- FIABLE, que realitzi totes les funcions correctament.
- ACTUAL, que es desenvolupi amb tecnologia actualitzada.
- REUSABLE, que es pugui fer servir des d'altres aplicacions
- USABILITAT, que el seu ús sigui senzill, intuïtiu i agradable.

## 2.2 Model conceptual

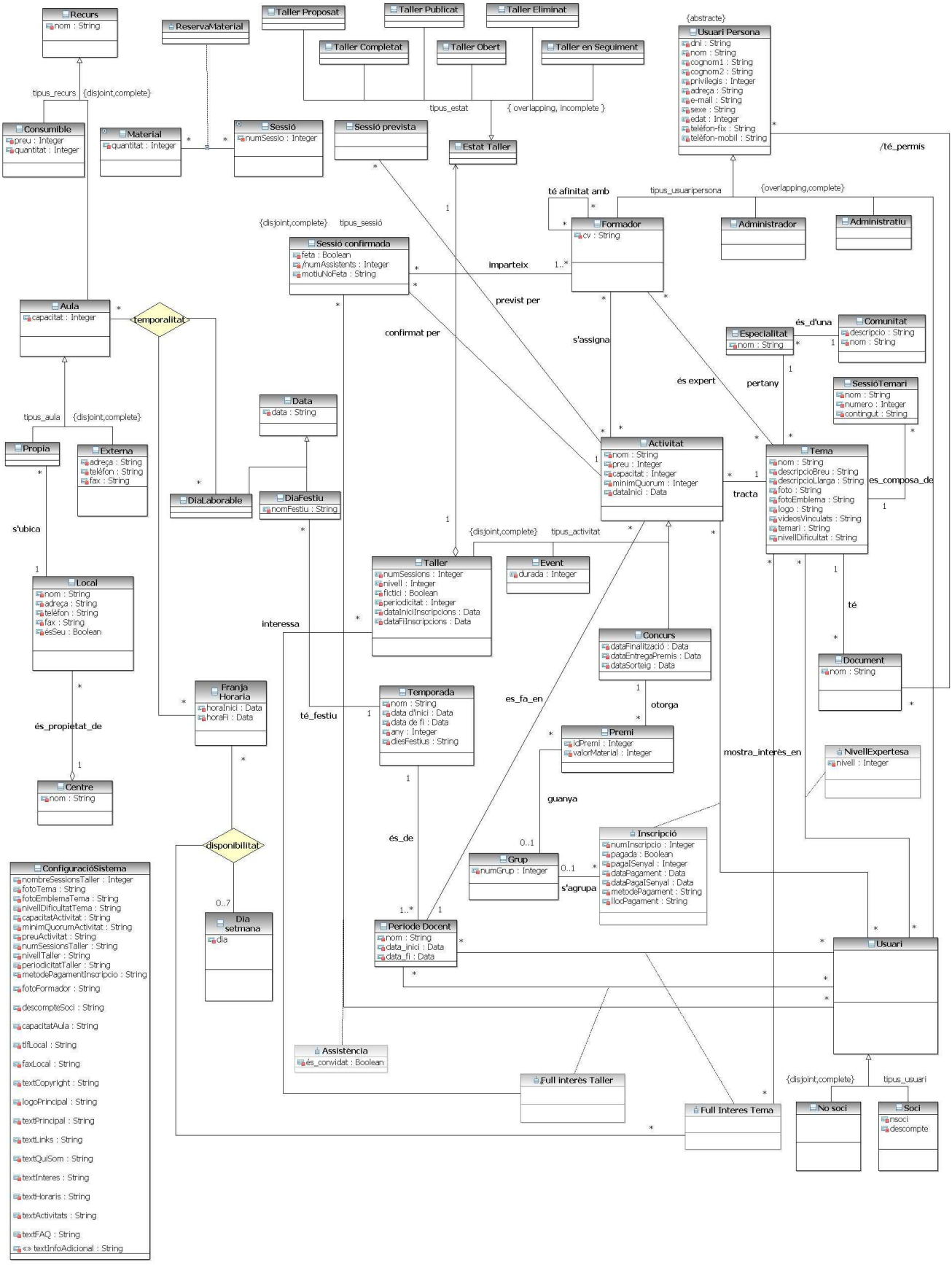
El model conceptual representa els conceptes (objectes) significatius en el domini del nostre problema. Principalment mostra:

- Classes d'objectes
- Associacions entre classes d'objectes
- Atributs de les classes d'objectes

El model conceptual també permet expressar restriccions aplicades als objectes del model, de manera que aquest es correspongui completament amb el domini del problema.

A continuació es presenta un diagrama on es mostren les classes que simbolitzen els conceptes.

## 2.2.1 Diagrama de classes



## Restriccions d'integritat

CLAUS EXTERNAS

| Clase | Clave |
|---|---|
| Recurs | nom |
| Local | nom |
| Centre | nom |
| Data | data |
| Activitat | nom |
| Temporada | (data_inici, data_fi) |
| PeriodeDocent | nom |
| Premi | id_premi |
| UsuariPersona | dni |
| FranjaHoraria | (hora_inici, hora_fi) |
| DiaSetmana | dia |
| Comunitat | nom |
| Tema | nom |
| Document | nom |
| Premi | id_premi |
| SessioTemari | nom |

Tal i com queda indicat en UML, les classes associatives (Inscripció, ReservaMaterial, Sessio, FullInterèsTema, FullInterèsTaller, NivellExpertesa, Temari, Assistència) són identificades mitjançant una clau composta per les claus de les classes que formen l'associació.

Les restriccions d'integritat textuals són aquelles que ha de cumplir el sistema però no apareixen gràficament en el model conceptual. A continuació es mostren:

1. Un alumne amb un nivell no pot inscriure's en un curs d'un nivell superior al seu.
2. Per tot taller, el nombre d'inscripcions ha de ser major o igual que el mínimquorum del taller.
3. Tots els períodes docents als que pertany un taller han de ser de la mateixa temporada.

4. Un formador assignat a un taller ha de ser expert en els temes que tracta el taller.
5. Un formador que imparteix una sessió d'un taller ha de ser expert en els temes que tracta el taller.
6. Tot formador assignat a un taller que te més formadors assignats ha d'estar en el conjunt d'afinitat de tots els formadors del taller.
7. Un grup només pot guanyar un premi d'un concurs en el que hi participa.
8. El dia de la setmana en que cau una data d'una sessió és el mateix dia de la setmana de l'espai-temps de la sessió.
9. Un usuari no pot fer una inscripció a una activitat que tingui les sessions previstes.

### 2.2.2 Atributs derivats

Els atributs derivats que apareixen en el nostrei sistema són els següents:

Class: SessióConfirmada
Atribut: **/numAssistents →** emmagatzema el número d'usuaris assistents a una sessió confirmada.

Entre les classes UsuariPersona i Document
Relació: **/te_permis →** emmagatzema si un usuari té permís de lectura sobre aquell document en funció de si està inscrit o no.

## 2.3 Model de casos d'ús

### 2.3.1 Definició d'actors

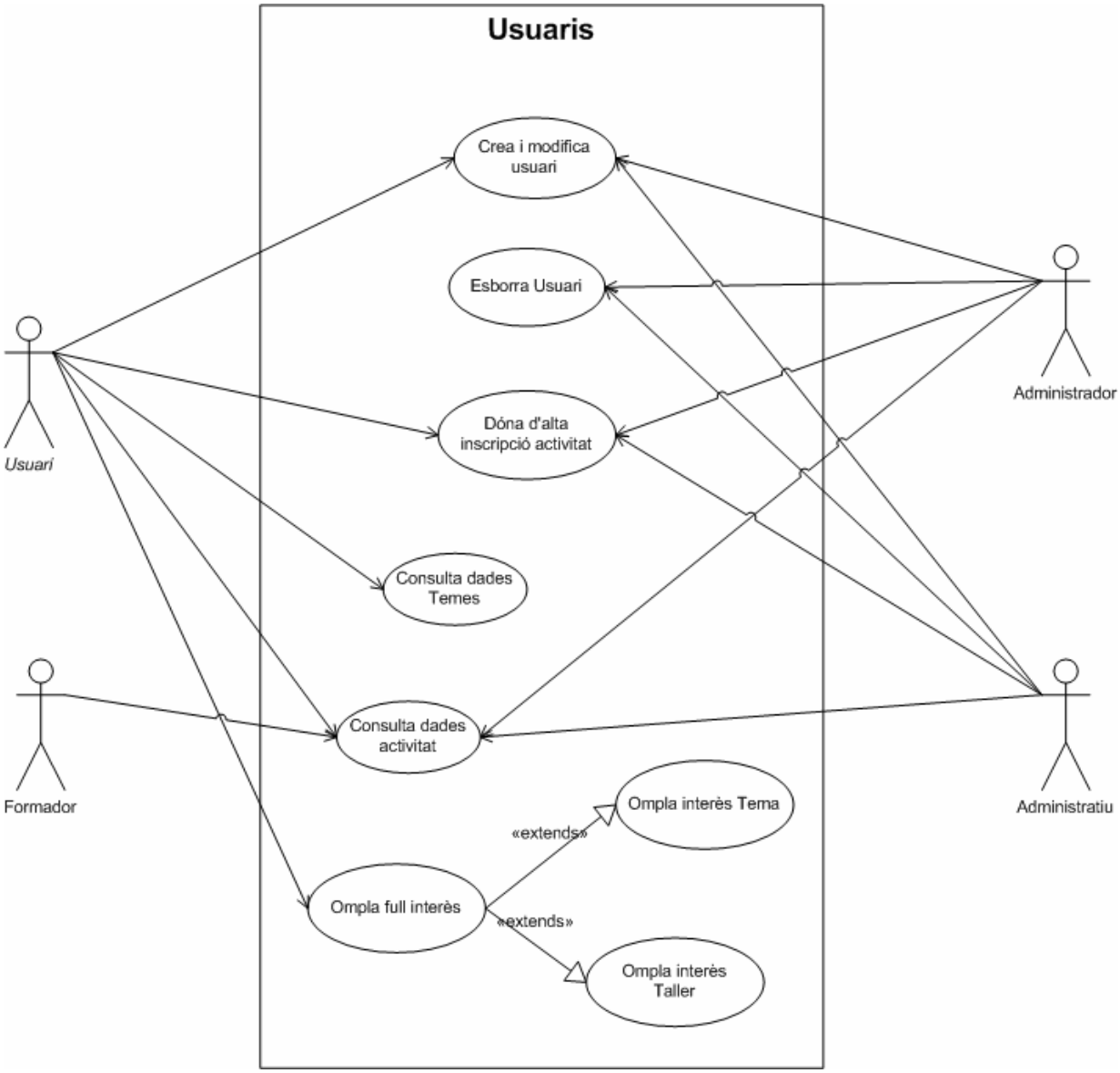
En la definició del model de casos d'ús del sistema GWAL trobem els següents actors els quals hereden de l'**UsuariPersona**:

- **Administrador**: Persona encarregada de l'administració del sistema.
- **Usuari**: Usuari del sistema que cerca informació sobre tallers per omplir fulls d'interès o inscriure-s'hi.
- **Administratiu:** Tipus d'usuari que representa a una empresa, i realitza tasques d'alta d'inscripcions, programació de cursos i gestió de formadors i usuaris.
- **Formador:** Persona que dóna classes a tallers del centre. Després de cada sessió confirmada que ha tingut lloc, ha de passar llista dels alumnes que han assistit.
- **Sistema**

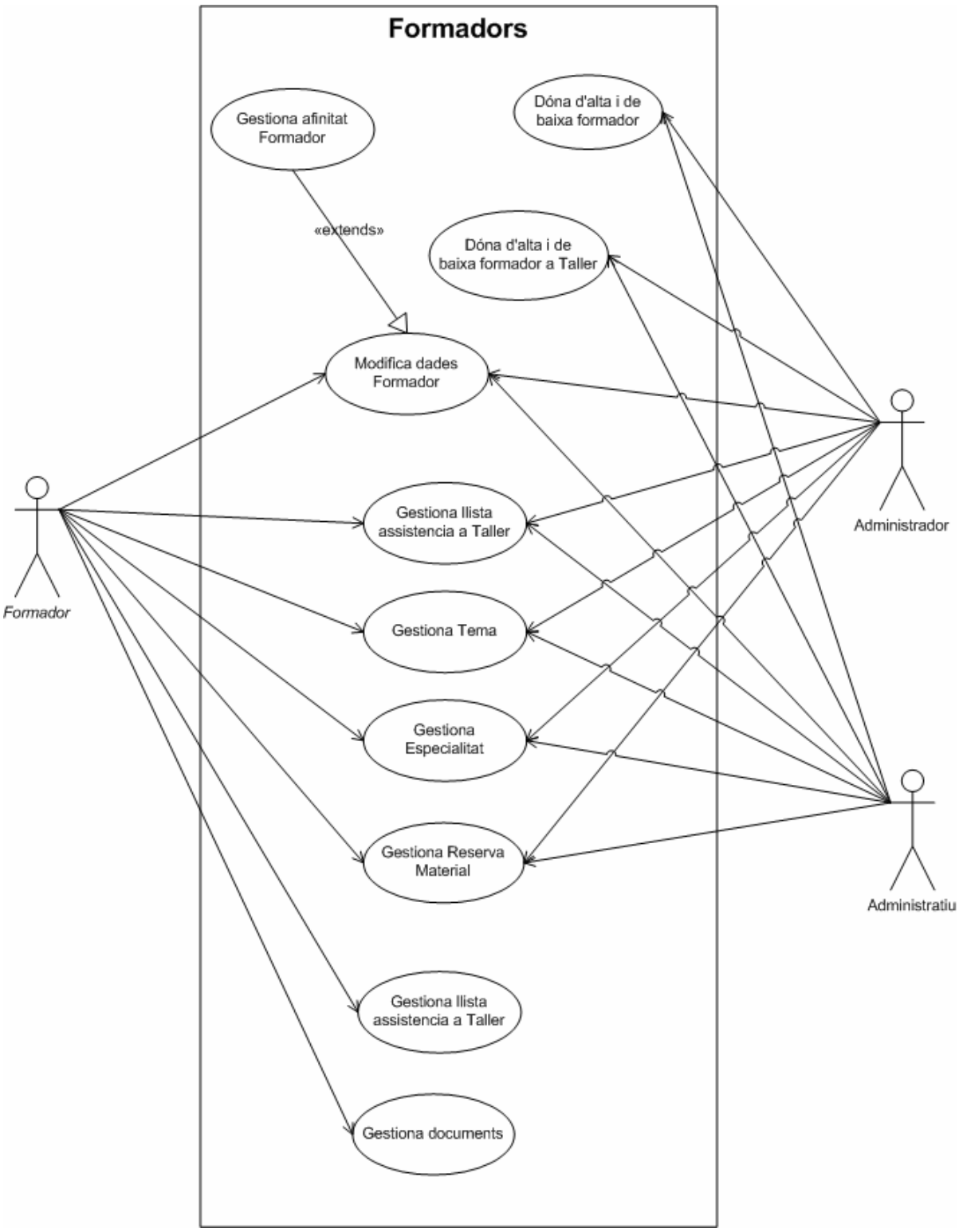
### 2.3.2 Diagrames de casos d'ús

Per tal de facilitar la comprenssió dels diagrames de casos d'ús, aquests s'han agrupat pels actors que hi intervenen, i per cada actor s'han agrupat els casos d'ús que tenen característiques comunes.

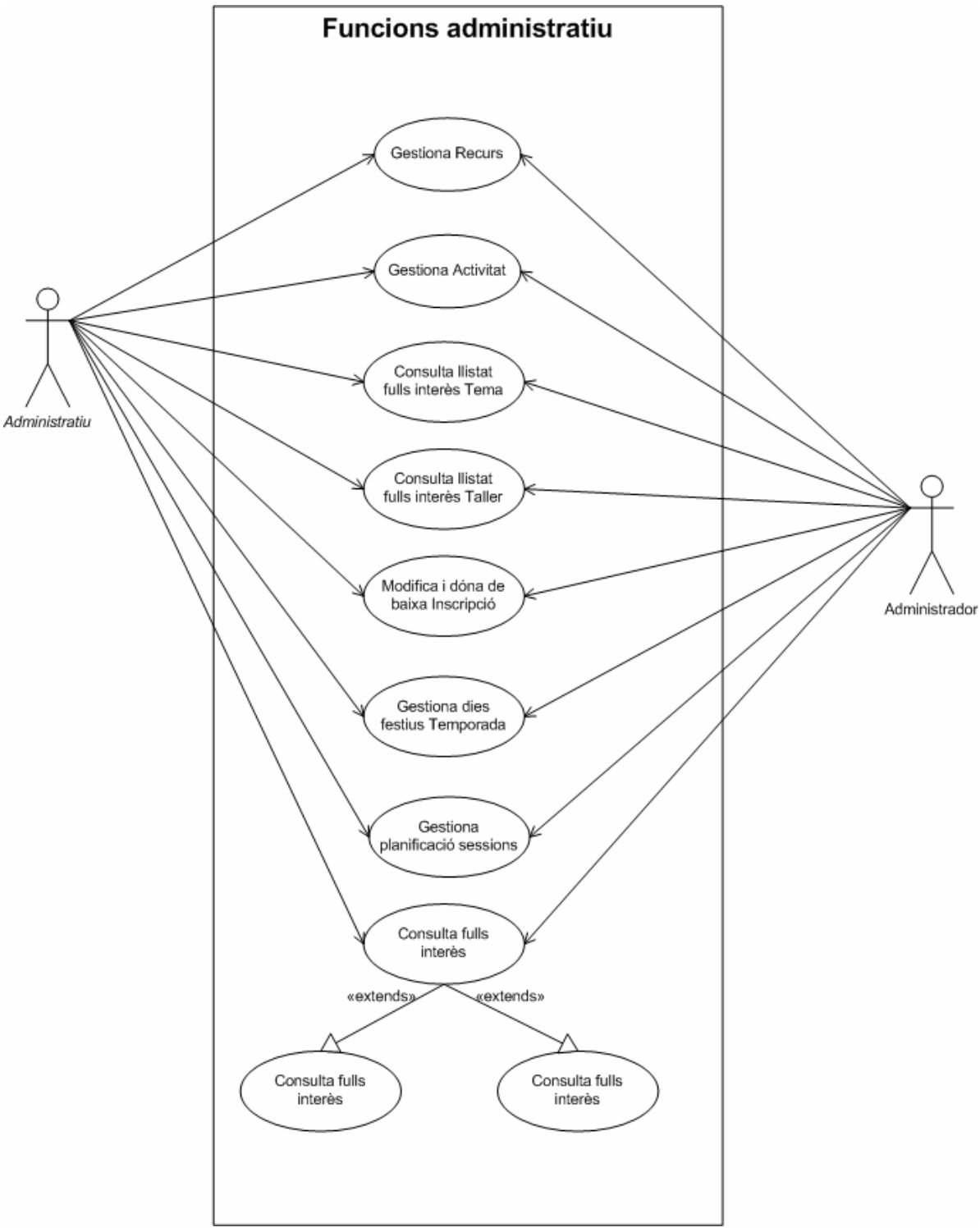
#### Casos d'ús d'Usuari



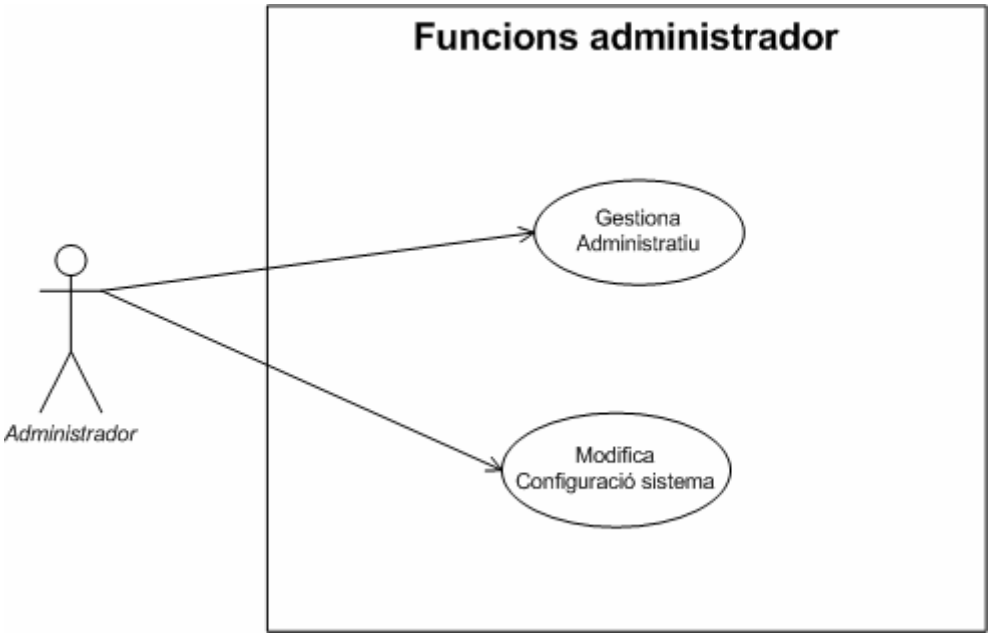
## Casos d’ús de Formador



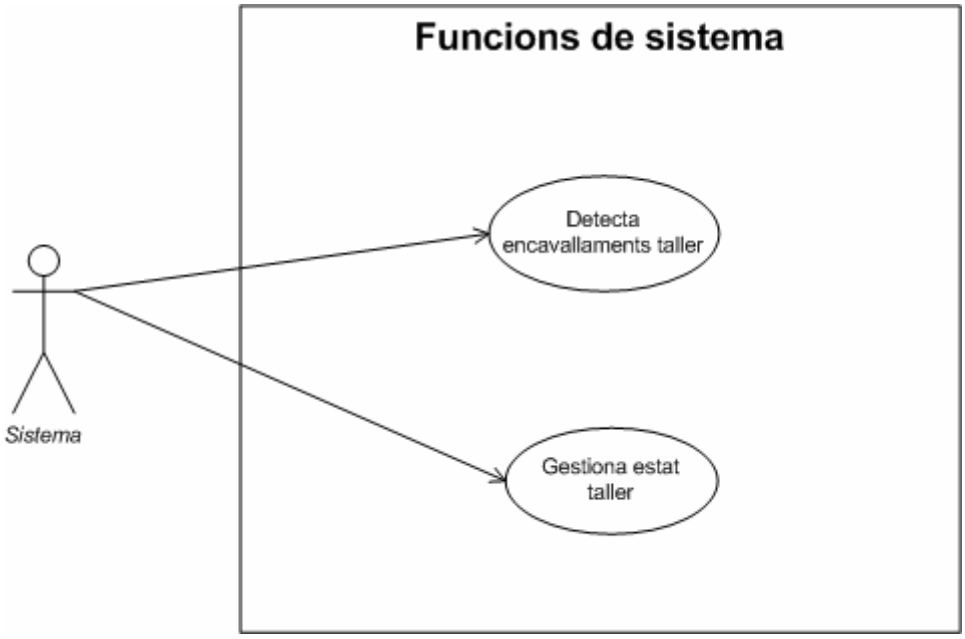
## Casos d'ús d'Administratiu



## Casos d'ús d'Administrador




## Casos d'ús de Sistema



### 2.3.3 Llista de casos d'ús

A continuació es mostra un llistat dels casos d'ús que defineixen les funcionalitats necessàries per a implementar el projecte.

1. Crea i modifica usuari
2. Esborra Usuari
3. Dóna d'alta inscripció Activitat
4. Consulta dades activitats
5. Consulta dades Temes
6. Ompla full interès
7. Ompla full interès Tema
8. Ompla full interès Taller
9. Dóna d'alta i baixa formador
10. Modifica dades formador
11. Gestiona afinitat formador
12. Dóna d'alta i baixa formador a Taller
13. Gestiona llista assistència a Taller
14. Gestiona documents
15. Gestiona Reserva Material
16. Gestiona Tema
17. Gestiona Especialitat
18. Gestiona Recurs
19. Gestiona Activitat
20. Consulta llistat fulls d'interès Temes
21. Consulta llistat fulls d'interès Tallers
22. Modifica i dóna de baixa Inscripció
23. Gestiona dies festius Temporada
24. Gestiona planificació sessions
25. Gestiona Administratiu
26. Modifica Configuracio Sistema
27. Detecta encavallaments taller
28. Gestiona estat taller

### 2.3.4 Especificació de casos d'ús

L'especificación dels casos d'ús es pot fer de dues formes:

- **D'alt nivell**: Descripció breu de les accions del cas d'ús.

  **Cas d'Ús**: Nom del Cas d'Ús.

  **Actors**: Llista d'actors, iniciador.

  **Tipus**: Primari, Secundari. Real, Essencial.

  **Descripció**: Descripció breu de les activitats que s'han de dur a terme.

- **Expandida**: Descripción detallada de las acciones y los requisitos. Añade a la especificación de alto nivel:

  **Curs típic d'esdeveniments:** Descripció detallada de la interacció (conversa) entre actors i el sistema.

  **Curs alternatiu d'esdeveniments**: Descriu excepcions al curs típic.

Els casos d'ús s'especificaran en alt nivell amb l'objetiu de facilitar la tasca. A continuació es presenta l'especificació dels casos d'ús en el mateix ordre en què apareixen en el llistat de casos d'ús, és a dir, agrupats per funcionalitats relacionades.

### 1. Crea i modifica usuari

| | |
|---|---|
| **Cas d'ús :** | Crea i modifica usuari |
| **Actors :** | Usuari, Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Crea un nou usuari registrat al sistema i permet modificar les seves dades personals |

### 2. Esborra Usuari

| | |
|---|---|
| **Cas d'ús :** | Esborra usuari |
| **Actors :** | Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Dóna de baixa un usuari del sistema |

### 3. Dóna d'alta inscripció Activitat

| | |
|---|---|
| **Cas d'ús :** | Dóna d'alta inscripció Activitat |
| **Actors :** | Usuari,Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet fer una inscripció d'un usuari a una activitat |

### 4. Consulta dades activitats

| | |
|---|---|
| **Cas d'ús :** | Consulta dades activitats |
| **Actors :** | Usuari, Formador, Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet consultar les activitats que hi ha obertes actualment i les seves dades |

### 5. Consulta dades Temes

| | |
|---|---|
| **Cas d'ús :** | Consulta dades Temes |
| **Actors :** | Usuari, Formador, Administrador, Administratiu (iniciadors) |

| | |
|---|---|
| **Tipus :** | Secundari |
| **Descripció :** | Permet consultar els temes d'una activitat i les seves dades |

### 6. Ompla full Interès

| | |
|---|---|
| **Cas d'ús :** | Ompla full Interès |
| **Actors :** | Usuari, Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Gestiona els fulls d'interes en temes i tallers d'un usuari, permet donar d'alta un full d'interes, modificar les dades i esborrar-lo |

### 7. Ompla full Interes Tema

| | |
|---|---|
| **Cas d'ús :** | Ompla full Interès Tema |
| **Actors :** | Usuari, Administrador, Administratiu (iniciadors) |
| **Tipus :** | Secundari |
| **Descripció :** | Gestiona els fulls d'interes en temes d'un usuari, permet donar d'alta un full d'interes, modificar les dades i esborrar-lo |

### 8. Ompla full Interes Taller

| | |
|---|---|
| **Cas d'ús :** | Ompla full Interès Taller |
| **Actors :** | Usuari, Administrador, Administratiu (iniciadors) |
| **Tipus :** | Secundari |
| **Descripció :** | Gestiona els fulls d'interes en tallers d'un usuari, permet donar d'alta un full d'interes, modificar les dades i esborrar-lo |

### 9. Dóna d'alta i baixa formador

| | |
|---|---|
| **Cas d'ús :** | Dóna d'alta i baixa formador |
| **Actors :** | Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet donar d'alta i de baixa un formador al sistema, amb el rol |

| | |
|---|---|
| | de formador associat |

### 10.Modifica dades formador

| | |
|---|---|
| **Cas d'ús :** | Modifica dades formador |
| **Actors :** | Formador, Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet modificar les dades personals del Formador |

### 11.Gestiona afinitat formador

| | |
|---|---|
| **Cas d'ús :** | Gestiona afinitat formador |
| **Actors :** | Formador, Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Gestiona les afinitats d'un formador amb d'altres formadors. Aquesta informació és rellavant a l'hora d'assignar formadors a una activitat. |

### 12. Dóna d'alta i baixa formador a Taller

| | |
|---|---|
| **Cas d'ús :** | Dóna d'alta i baixa formador a Taller |
| **Actors :** | Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet assignar o desassignar un formador donat a un taller existent |

### 13.Gestiona llista assistència a Taller

| | |
|---|---|
| **Cas d'ús :** | Gestiona llista assistència a Taller |
| **Actors :** | Formador (iniciador) |
| **Tipus :** | Primari |
| **Descripció :** | Permet crear, modificar i esborrar la llista d'assistència d'usuaris a una sessió confirmada d'un taller |

### 14.Gestiona documents

| | |
|---|---|
| **Cas d'ús :** | Gestiona documents |
| **Actors :** | Formador (iniciador) |
| **Tipus :** | Primari |
| **Descripció :** | Permet afegir o eliminar documents relacionats amb el contingut docent d'una activitat a aquesta |

### 15.Gestiona Reserva Material

| | |
|---|---|
| **Cas d'ús :** | Gestiona Reserva Material |
| **Actors :** | Formador, Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet realitzar, modificar o donar de baixa reserves de material |

### 16.Gestiona Tema

| | |
|---|---|
| **Cas d'ús :** | Gestiona Tema |
| **Actors :** | Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet gestionar totes les operacions sobre un tema: alta, modificació del contingut i baixa |

### 17.Gestiona Especialitat

| | |
|---|---|
| **Cas d'ús :** | Gestiona Especialitat |
| **Actors :** | Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet gestionar totes les operacions sobre una especialitat: alta, modificació del contingut i baixa |

### 18.Gestiona Recurs

| | |
|---|---|
| **Cas d'ús :** | Gestiona Recurs |
| **Actors :** | Usuari, Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet donar d'alta, modificar-ne l'informació o donar de baixa un recurs en el sistema |

### 19.Gestiona Activitat

| | |
|---|---|
| **Cas d'ús :** | Gestiona Activitat |
| **Actors :** | Usuari, Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Gestiona les accions referents a l'activitat: donar d'alta una activitat nova, modificar-ne el contingut, donar-la de baixa i especificar el tema i l'especialitat que tractarà. |

### 20.Consulta llistat fulls d'interès Temes

| | |
|---|---|
| **Cas d'ús :** | Consulta llistat fulls d'interès Temes |
| **Actors :** | Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet consultar la llista de tots els fulls d'interès que han omplert els usuaris sobre els temes que s'han proposat en un període docent |

### 21.Consulta llistat fulls d'interès Tallers

| | |
|---|---|
| **Cas d'ús :** | Consulta llistat fulls d'interès Tallers |
| **Actors :** | Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet consultar la llista de tots els fulls d'interès que han omplert els usuaris sobre els tallers que s'han proposat en un període docent |

### 22.Modifica i dóna de baixa Inscripció

| | |
|---|---|
| **Cas d'ús :** | Modifica i dóna de baixa Inscripcio |
| **Actors :** | Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet modificar les dades d'una inscripció d'un usuari a una activitat així com donar-la de baixa |

### 23.Gestiona dies festius Temporada

| | |
|---|---|
| **Cas d'ús :** | Gestiona dies festius Temporada |
| **Actors :** | Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet donar d'alta i de baixa dies festius en una temporada. En fer-ho, el sistema haurà de comprovar si hi ha sessions previstes o confirmades d'una activitat en aquella data. |

### 24.Gestiona planificació sessions

| | |
|---|---|
| **Cas d'ús :** | Gestiona planificació sessions |
| **Actors :** | Administrador, Administratiu (iniciadors) |
| **Tipus :** | Primari |
| **Descripció :** | Permet gestionar la planificació de sessions d'una activitat. Bàsicament, seran les operacions de planificar les sessions per una activitat, la modificació de la planificació inicial i l'esborrat de la mateixa. |

### 25.Gestiona Administratiu

| | |
|---|---|
| **Cas d'ús :** | Gestiona Administratiu |
| **Actors :** | Administrador (iniciador) |
| **Tipus :** | Primari |
| **Descripció :** | Permet donar d'alta usuaris de tipus administratiu al sistema, amb el seu rol associat, així com modificar-ne les seves dades o |

| | |
|---|---|
| | donar-los de baixa |

### 26.Modifica Configuració Sistema

| | |
|---|---|
| **Cas d'ús :** | Modifica Configuració Sistema |
| **Actors :** | Administrador (iniciador) |
| **Tipus :** | Primari |
| **Descripció :** | Permet modificar els paràmetres de la configuració del sistema que ajuden a què l'eina sigui més usable |

### 27.Detecta encavallaments taller

| | |
|---|---|
| **Cas d'ús :** | Detecta encavallaments taller |
| **Actors :** | Sistema |
| **Tipus :** | Primari |
| **Descripció :** | Quan s'ha donat d'alta un taller i abans que aquest s'obri, el sistema haurà de detectar els encavallaments de les sessions previstes que s'hagin planificat amb la resta de sessions d'altres tallers i avisar a l'usuari si en troba. |

### 28.Gestiona estat taller

| | |
|---|---|
| **Cas d'ús :** | Gestiona estat taller |
| **Actors :** | Sistema |
| **Tipus :** | Primari |
| **Descripció :** | Funció que s'encarrega de supervisar l'estat d'un taller i actualitzar-ne el seu estat en cas necessari. Així, un taller confirmat passarà a estar obert quan arribi la data d'inici d'inscripcions. Un cop obert passarà a en estat "en seguiment obert" quan s'inciï el curs. Quan finalitzi el període d'inscripcions, el sistema haurà d'actualitzar l'estat del taller a l'estat de "en seguiment". Finalment, quan el curs hagi finalitzat, el sistema marcarà el taller com a "completat" |

## 2.4 Model del comportament

El model del comportamento mostra l'**aspecte dinàmic** d'un sistema software, és a dir, mostra les interaccions entre els diversos objectes del sistema.

La realització del model del comportament es divideix en tres fases:

- Diagrames de seqüència
- Contractes de les operacions
- Diagrames d'activitat d'objectes

### 2.4.1 Diagrames de seqüència i contractes de les operacions

Per a fer més fàcil la lectura, s'han disposat de forma seguida els diagrames de seqüència i els contractes de les operacions de cada cas d'ús. Els diagrames de seqüència mostren la seqüència de successos entre els actors i el sistema i permeten identificar les operacions del sistema. En canvi, els contractos de les operacions defineixen el comportament del sistema definint els canvis d'estat de la base d'informació i les sortides que el sistema proporciona. Es defineix un contracte per cada operació que apareix en els diagrames de seqüència.

Els contractes de les operacions inclouen principalment:

**Nom:** Nom de l'operació.
**Cas d'ús:** Nom del cas d'ús al que fa referència
**Responsabilitats:** Descripción informal del propòsit de l'operació.
**Excepcions:** Descripció de la reacció del sistema a situacions excepcionals.
**Precondicions:** Condicions sobre l'estat del sistema abans de la invocació de l'operació.
**Postcondicions:** Canvis d'estat en el domini que s'han produit després de la invocació de l'operació.
**Sortida:** Descripció de la sortida que proporciona l'operació en OCL.

La nomenclatura que s'ha utilitzado per definir la informació composta en els fluxes de dades és la següent:

Inclusió: a = (a1, a2)

Repetició: a = {a1, a2}

Selecció: a = [a1 | a2]

Els casos d'ús més representatius que es mostren en aquesta memòria són els següents:

1. Dóna d'alta inscripció activitat
2. Ompla full interès Tema
3. Dóna d'alta afinitat formador
4. Dóna de baixa formador a taller
5. Dóna d'alta llista assistència a taller
6. Dóna de baixa recurs
7. Dóna de baixa Tema
8. Consulta llistat fulls interès Temes
9. Planifica sessions Taller
10. Detecta encavallaments Taller
11. Confirma Taller

## 1. Dóna d'alta inscripció activitat


dadesInscripcio=(pagada, pagaISenyal, dataPagament, dataPagaISenyal, metodePagament, llocPagament)

**Nom:** AltaInscripcioActivitat(dadesInscripcio, usuari, activitat)

**Cas d'ús:** Dóna d'alta inscripció activitat

**Responsabilitats:** Dóna d'alta una inscripció d'un usuari a una activitat

**Excepcions:**

- No existeix un **Usuari** amb dni=usuari
- No existeix una **Activitat** amb nom=activitat

**Precondicions:**

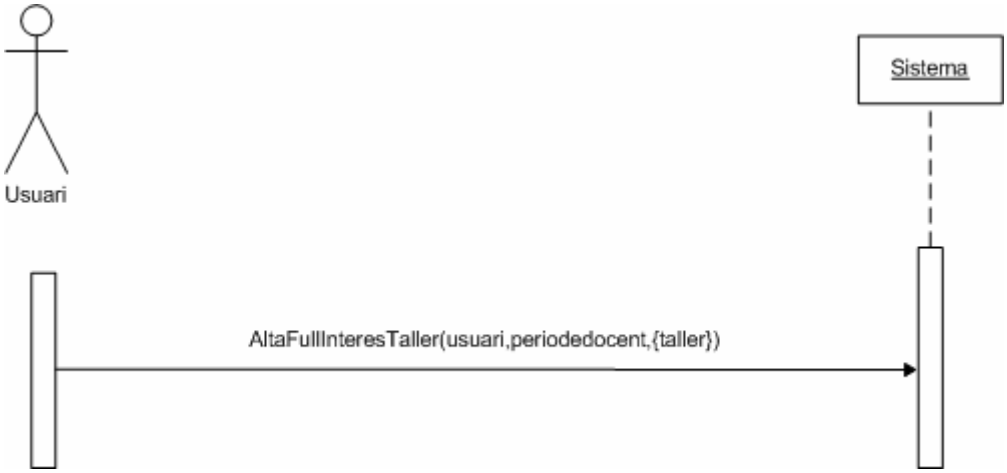
- Existeix un **Usuari** amb dni=usuari
- Existeix una **Activitat** amb nom=activitat

**Postcondicions:**

1. Es crea una nova instància de la classe associativa **Inscripció** i s'inicialitza amb els paràmetres de l'operació

**Sortida:** -

**2. Ompla full interès Taller**




**Nom:** AltaFullInteresTaller(usuari,periodedocent, {taller})

**Cas d'ús:** Ompla full interès Taller

**Responsabilitats:** Dóna d'alta un full d'interès d'un usuari en un període docent amb els tallers que li interessen

**Excepcions:**

- No existeix un **Usuari** amb dni=usuari
- No existeix un **Periode Docent** amb nom=periodedocent
- Per a cada Taller a {taller} no existeix el **Taller** amb nom=taller
- Existeix un **Full Interes Taller** de l'Usuari amb dni=usuari i el Periode Docent amb nom=periodedocent

**Precondicions:**

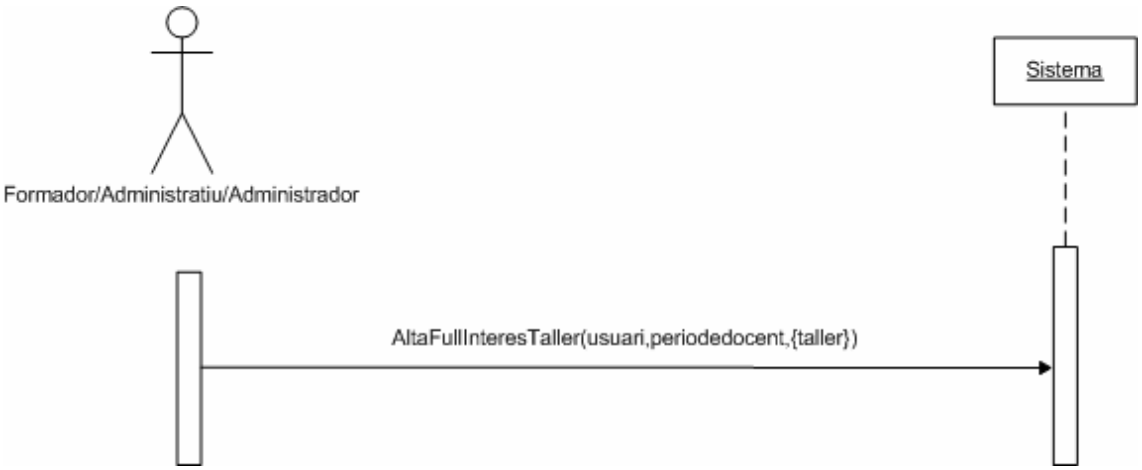
- Existeix un **Usuari** U amb dni=usuari
- Existeix un **Periode Docent** PD amb nom=periodedocent
- Per a cada Taller a {taller} existeix el **Taller** amb nom=taller
- No existeix un **Full Interes Taller** de l'Usuari amb dni=usuari i el Periode Docent amb nom=periodedocent

**Postcondicions:**

1. Dóna d'alta una nova instància FIT de la classe associativa **Full Interes Taller** entre l'Usuari U i el Periode Docent PD
2. Per a cada Taller de {taller}
    a. Dóna d'alta una instància de l'associació '**interessa'** entre el Full Interes Taller FIT i el Taller amb id=taller

**Sortida:** -

### 3. Dóna d'alta afinitat formador




**Nom:** AltaAfinitatFormador(formador_actual,{formador_afi})

**Cas d'ús:** Dóna alta afinitat formador

**Responsabilitats:** Dóna d'alta una llista de formadors afins amb un formador donat.

**Excepcions:**

- No existeix un **Formador** amb dni=formador_actual
- Per a cada **Formador fa** amb dni =formador_afi, no existeix fa
- Per algun **Formador fa** amb dni =formador_afi, existeix una instància de l'associació 'te_afinitat_amb' entre fa i el Formador amb dni=formador_actual

**Precondicions:**

- Existeix un **Formador** amb dni=formador_actual
- Per a cada **Formador fa** amb dni =formador_afi, existeix fa
- Per a cada **Formador fa** amb dni =formador_afi, no existeix una instància de l'associació 'te_afinitat_amb' entre fa i el Formador amb dni=formador_actual

**Postcondicions:**

1. Per a cada **Formador fa** de {formador_afi}
    b. Dóna d'alta una nova instància de l'associació 'te_afinitat_amb' entre fa i el **Formador** amb dni=formador_actual

**Sortida:** -

## 4. Dóna de baixa formador a taller


**Nom:** BaixaFormadorTaller(formador,taller)

**Cas d'ús:** Dóna de baixa formador a Taller

**Responsabilitats:** Dóna de baixa un formador que estaba assignat a un taller.

**Excepcions:**

- No existeix un **Formador** amb dni=formador
- No existeix un **Taller** amb nom=taller
- No existeix una instància de l'associació 's'assigna' entre el Formador amb dni=formador i el Taller amb nom=taller

**Precondicions:**

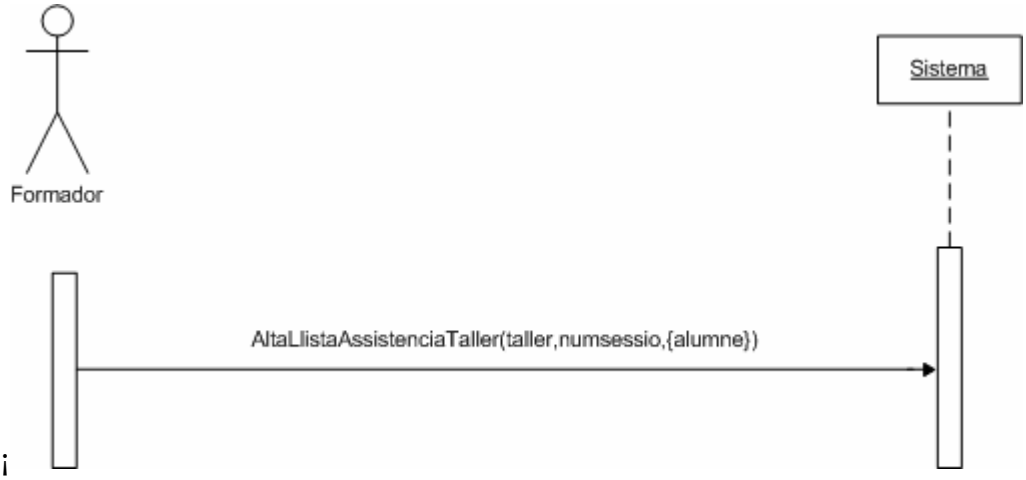
- Existeix un **Formador** amb dni=formador
- Existeix un **Taller** amb nom=taller
- Existeix una instància de l'associació 's'assigna' entre el Formador amb dni=formador i el Taller amb nom=taller

**Postcondicions:**

1. Dóna de baixa la instància S de l'associació entre el Formador amb dni=formador i el Taller amb nom=taller

**Sortida:** -

**5. Dóna d'alta llista assistència a taller**




¡

**Nom:** AltaLlistaAssistenciaTaller(taller,numsessio,{alumne, convidat})

**Cas d'ús:** Dóna d'alta llista assistència a taller

**Responsabilitats:** Dóna d'alta la llista d'assistència d'un conjunt d'alumnes a una sessió confirmada d'un taller.

**Excepcions:**

- No existeix un **Taller** amb nom=taller
- No existeix una **Sessio Confirmada** amb numSessio=numsessio
- Per algun **Usuari** a {alumne}, no existeix l'Usuari amb dni=alumne

**Precondicions:**

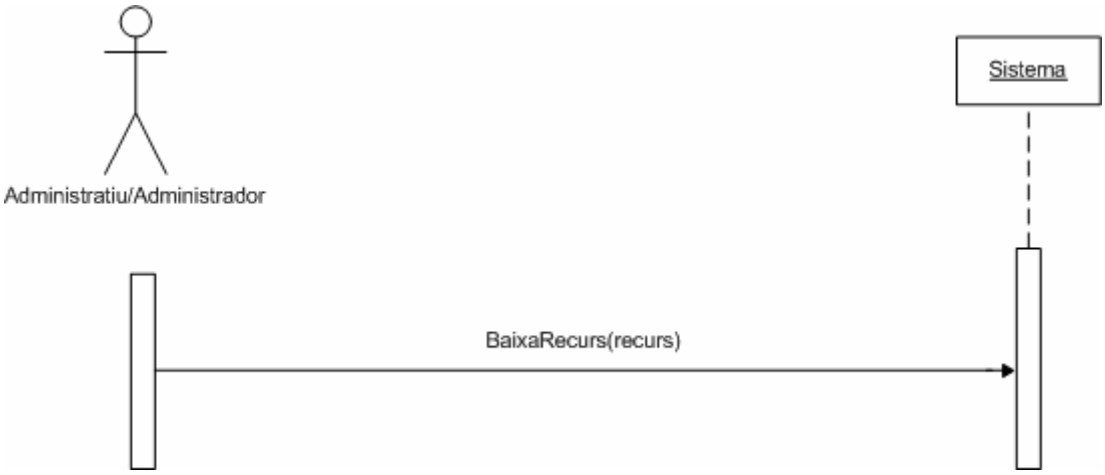
- Existeix un **Taller** amb nom=taller
- Existeix una **Sessio Confirmada** amb numSessio=numsessio
- Per a cada **Usuari** a {alumne}, no existeix una instància d'Usuari amb dni=alumne

**Postcondicions:**

1. Per a cada Usuari a {alumne, convidat}
    a. Dóna d'alta una instància de la classe associativa **Assistència A** entre l'Usuari amb dni=alumne i el Taller amb nom=Taller. A actualitza el seu atribut es_convidat=convidat.
2. Un cop finalitzada l'entrada d'assistencies, es recalcula l'atribut **numAssistents** de la Sessio Confirmada amb numSessio=numsessio amb el nombre d'instàncies Assistència creades.
3. Dóna d'alta una instància de la classe 'imparteix' entre el Formador i l'assistència A.

**Sortida:** -

### 6. Dóna de baixa recurs




**Nom:** BaixaRecurs(recurs)

**Cas d'ús:** Dóna de baixa Recurs

**Responsabilitats:** Dóna de baixa un recurs.

**Excepcions:**

- No existeix un **Recurs** amb nom=recurs

**Precondicions:**

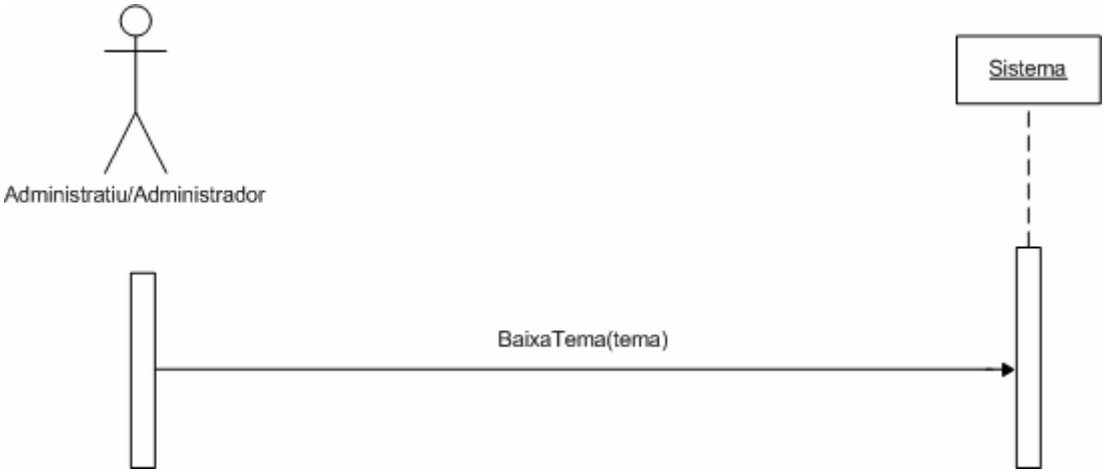
- Existeix un **Recurs** amb nom=recurs

**Postcondicions:**

1. En cas que el Recurs R amb nom=recurs sigui de tipus Material, caldrà fer per cada instància de la classe associativa Recurs Material RM amb Material R (si n'hi ha):
    a. Esborrar-ne totes les instàncies 'fa_reserva' entre RM i instàncies de la classe Usuari.
    b. Esborrar la instància RM de Recurs Material.
2. Dóna de baixa la instància de Recurs R.

**Sortida:** -

## 7. Dóna de baixa Tema




**Nom:** BaixaTema(tema)

**Cas d'ús:** Dóna de baixa Tema

**Responsabilitats:** Dóna de baixa un tema.

**Excepcions:**

- No existeix un **Tema** amb nom=tema

**Precondicions:**

- Existeix un **Tema** amb nom=tema

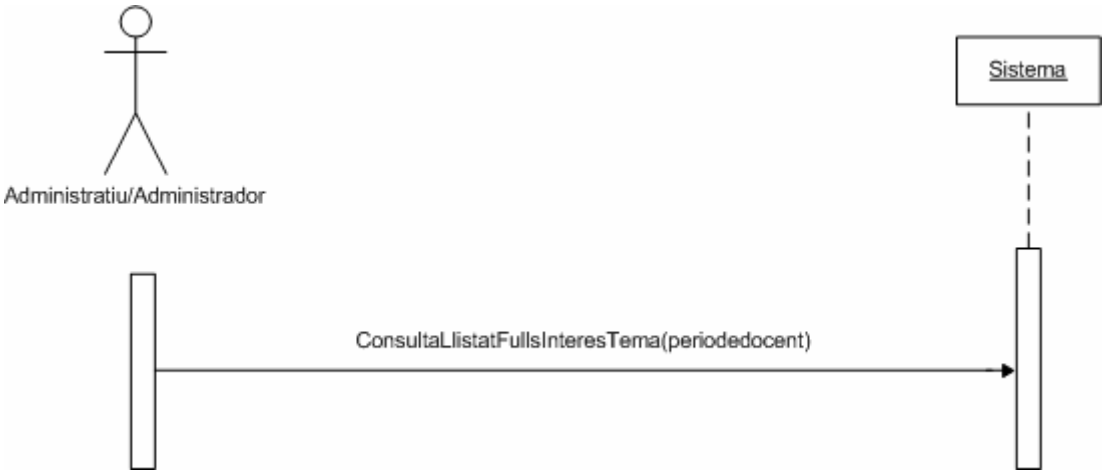
**Postcondicions:**

1. Dóna de baixa la instància de la associació 'pertany' amb l'**Especialitat** E amb qui té la relació.
2. Per cada instància de la associació 'es_composa_de':
    a. Dóna de baixa la **SessioTemari** ST accessible a través de la instància de l'associació 'es_composa_de'.
    b. Dóna de baixa la instància de l'associació 'es_composa_de'.
3. Per cada instància de la associació 'té':
    a. Dóna de baixa el **Document** D accessible a través de la instància de l'associació 'té'.
    b. Dóna de baixa la instància de l'associació 'té'.
4. Per cada instància de classe associativa **Nivell Expertesa** NE entre el Tema T i instàncies de la classe Usuari:
    a. Dóna de baixa la instància NE de la classe Nivell Expertesa
5. Per cada instància de la associació 'mostra_interes_en':
    a. Dóna de baixa la instància de l'associació 'mostra_interes_en' amb les instàncies de la classe **Full Interes Tema**.

6. Per cada instància de la associació 'tracta', dóna de baixa la instància de l'associació 'tracta'' entre la instància del Tema T i la instància de la classe Activitat.

**Sortida:** -

## 8. Consulta llistat fulls interès Temes




**Nom:** ConsultaLlistatFullsInteresTema(periodedocent)

**Cas d'ús:** Consulta llistat fulls interes Temes

**Responsabilitats:** Obté la llista de fulls d'interès en un periode docent.

**Excepcions:**

- No existeix un **Periode Docent** amb nom=periodedocent

**Precondicions:**

- Existeix un **Periode Docent** PD amb nom=periodedocent

**Postcondicions:** -

**Sortida:** Per a cada **Full Interes Tema** FIT del Periode Docent PD, amb **Usuari** U es mostrarà:

- U.dni, U.nom, U.cognom, U.email, U.nsoci

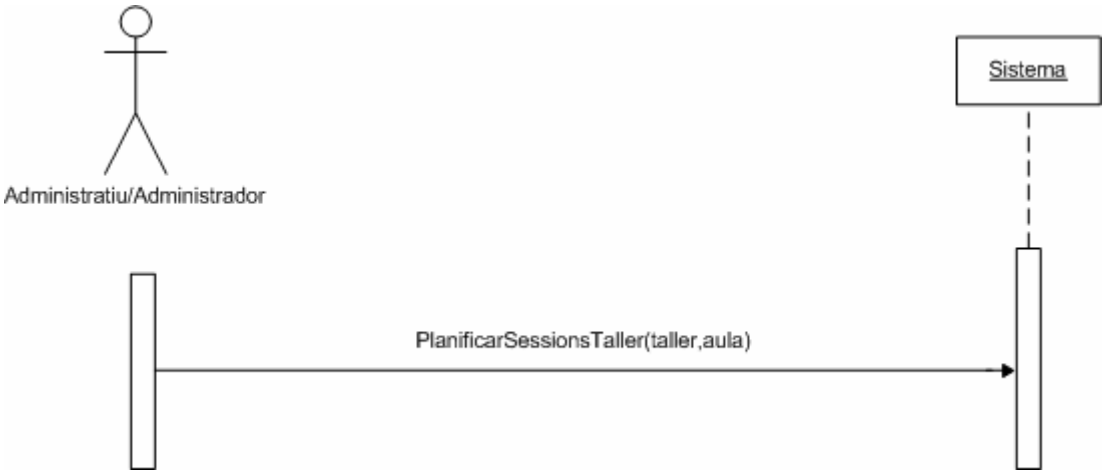
Per a cada instancia T de la classe **Tema** accesible a través de l'associació 'mostra_interes_en':

- T.nom

Per cada **Disponibilitat** D asociada amb FIT:

- D.dia, D.horaInici, D.horaFi.

## 9. Planifica sessions Taller




**Nom:** PlanificarSessionsTaller(taller,aula)

**Cas d'ús:** Planifica sessions Taller

**Responsabilitats:** Planifica les sessions d'una taller en una aula tenint en compte la seva data d'inici, els dies que es fa el taller i la seva periodicitat, el nombre de sessions del taller i el calendari de dies festius.

**Excepcions:**

- No existeix un **Taller** amb nom=taller
- No existeix una **Aula** amb nom=aula

**Precondicions:**

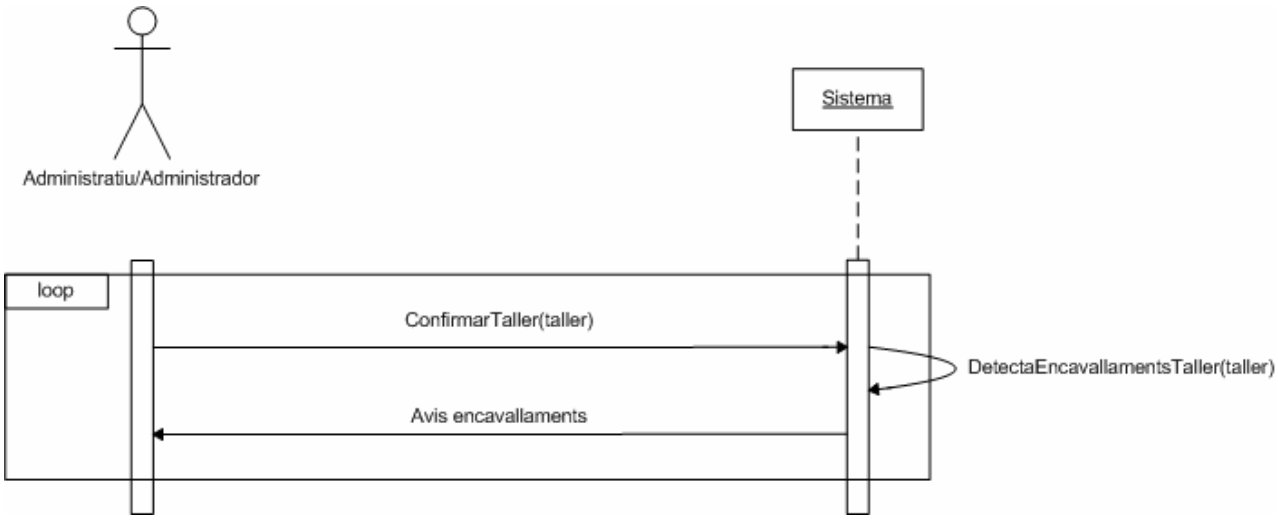
- Existeix un **Taller** T amb nom=taller
- Existeix una **Aula** A amb nom=aula

**Postcondicions:**

1. Crea una instància SP de la classe **Sessió Prevista** entre l'Aula A i la Data amb data igual a T.dataInici.
2. Mentre no s'hagin creat un nombre de sessions previstes igual a T.numSessions
    c. Crea una instància SP de la classe **Sessió Prevista** entre l'Aula A i la Data amb data igual a T.dataInici + T.periodicitat sempre que no sigui festiu.

**Sortida:** -

**10. Confirma Taller**




**Nom:** DetectaEncavallamentsTaller(taller)

**Cas d'ús:** Confirma Taller

**Responsabilitats:** Detecta encavallaments d'un taller mentre aquest encara no és obert.

**Excepcions:**

- No existeix un **Taller** amb nom=taller

**Precondicions:**

- Existeix un **Taller** amb nom=taller

**Postcondicions:**

1. Per a cada instància de la classe Sessio Prevista accessibles a través de l'associació 'previst_per':
    a. Mirar que si existeix una sessió confirmada o prevista en la mateixa data, aula i franges que s'encavallin. En cas positiu aturar-se i donar un avís a l'usuari informant de l'encavallament i instant-lo a que arregli la situació.

**Sortida:** Si s'han trobat encavallaments, retornar l'avís per l'usuari. Altrament no retornar null.

**Nom:** ConfirmaTaller(taller)

**Cas d'ús:** Confirma Taller

**Responsabilitats:** Passa un taller a estat confirmat

**Excepcions:**

- No existeix un **Taller** amb nom=taller

**Precondicions:**

- Existeix un **Taller** T amb nom=taller

**Postcondicions:**

1. Crida a la funció DetectaEncavallamentsTaller(taller)
2. Si retorna null
    a. Per a cada instància de l'associació 'prevista_per' del Taller T
        i. Crea una instància de la classe Sessió Confirmada SC amb paràmetre sessioFeta=false.
        ii. Crea una instància de l'associació 'confirmada_per' entre el Taller T i la Sessió Confirmada SC.
        iii. Esborra la instància de l'associació 'prevista per' entre el Taller T i la Sessió Prevista SP.
        iv. Esborra la instància de la classe Sessió Prevista SP.
    b. Canvia l'estat del Taller T a '*confirmat'*
3. En cas contrari, retorna l'avís amb els encavallaments trobats

**Sortida:** -

### 2.4.2 Diagrames d'activitat d'objectes

A continuació es mostra el diagrama d'activitat per la classe Taller, que s'implementa amb un patró estat.

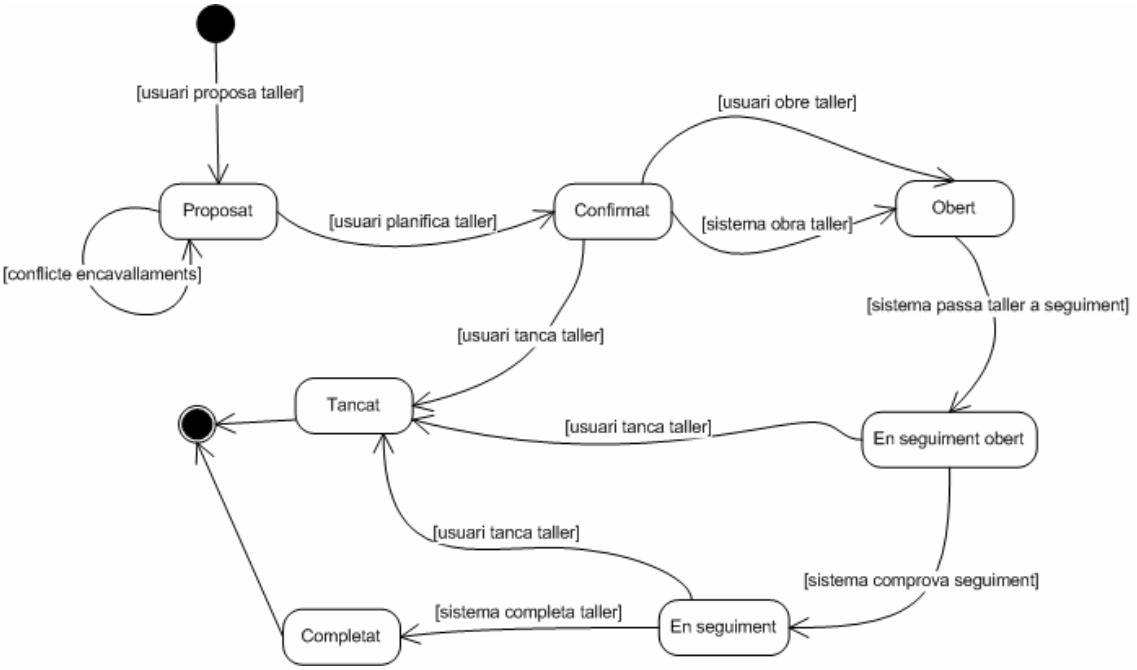



*Diagrama d'activitat per a Taller*

Un taller pot passar per 7 estats diferents en el seu cicle de vida. El gràfic de la imatge mostra els estats del taller i els esdeveniments que fan que passi d'un estat a l'altre.

Així quan un administrador o administratiu crea un Taller aquest neix amb l'estat *Proposat*. Quan es demana planificar el taller es generen totes les sessions per al taller i aquestes poden entrar en conflicte amb algunes sessions d'altres activitats.

Un cop es prenen decisions per a resoldre els encavallaments, és a dir es canvien les sessions de lloc o es tanquen els tallers en conflicte, el taller esdevé *Confirmat*. En aquest estat es pot cancelar el taller per la raó que sigui i passar-la a l'estat *Tancat*.

Ja sigui perquè s'obre manualment un taller o bé perquè la data actual és més gran o igual que la data d'inici d'inscripcions, el taller passa a l'estat d'*Obert.*

Una vegada arribi l'inici del curs, el taller passa automàticament a l'estat *En seguiment obert* permetent-ne encara durant uns dies les inscripcions. Quan es

sobrepassa el periode d'inscripcions aquest passarà automàticament a l'estat *En seguiment* i ja no es permetran més incripcions.

Finalment, quan arriba la data de finalització del curs s'executa una funció batch que canvia l'estat del taller a *Completat*.

# 3 Disseny

## 3.1 Decisions de disseny

En la fase de disseny es parteix de l'especificació i s'apliquen diferents tècniques per definir el sistema de forma prou detallada per a poder-lo implementar. És en aquesta etapa on, a més de tenir en compte els requisists que ha de cumplir el sistema, es considera també la tecnologia que s'utilitzarà.

Aquest apartat pretèn explicar les decisions de disseny que s'han pres: la plataforma física i l'arquitectura de l'aplicació.

Addicionalment, s'ha considerat interessant incloure un apartat sobre la usabilitat i com aquesta afecta al disseny de la capa de presentació.

### 3.1.1 Arquitectura de l'aplicació

L'arquitectura de l'aplicació és una descripció dels subsistemes i components d'un sistema i les relacions entre ells. L'arquitectura és un conjunt de decisions sobre:

- L'organització del sistema software
- La selecció dels elements estructurals i les interfícies que composen el sistema
- El comportament dels elements estructurals
- La composició d'elements en subsistemes més grans

Per dissenyar l'arquitectura cal decidir quina serà l'estructgura general utilitzada. De totes les propietats desitjables d'una arquitectura, es considera que és important aconseguir que l'aplicació sigui eficient i fiable. A partir d'aquestes propietats cal escollir els patrons arquitectònics que millor s'adaptin a les necessitats del sistema.

Un patró arquitectònic expresa un esquema d'organització estructural fonamental pels sistemes de software. Proporciona un conjunt de subsistemes, especifica les seves responsabilitats i inclou regles i guies per organitzar les relacions entre ells.

Els patrons arquitectònics donen l'esquema general del sistema. Després, és necessari adaptar el patró arquitectònic escollit al sistema concret que s'està

desenvolupant. Aquesta adaptació es realitza a través de la utilització de patrons de disseny.

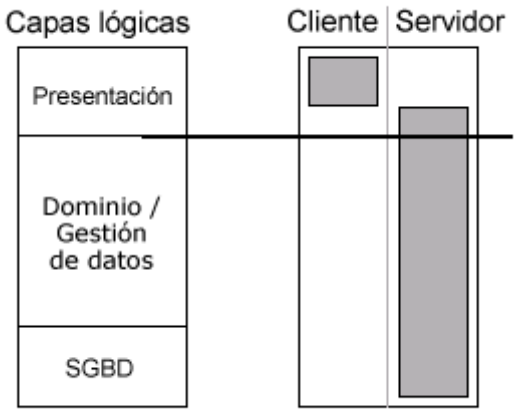
Per definir l'arquitectura del nostre sistema, els patrons arquitectònics que han estat escollit han estat:

- Arquitectura client – servidor
- Arquitectura en tres capes

## Arquitectura client – servidor

El patró arquitectònic client-servidor proposa solucions al problema de com partir un sistema en un conjunt de components clients i servidors, de manera que satisfacin els requisits funcionals i no funcionals.

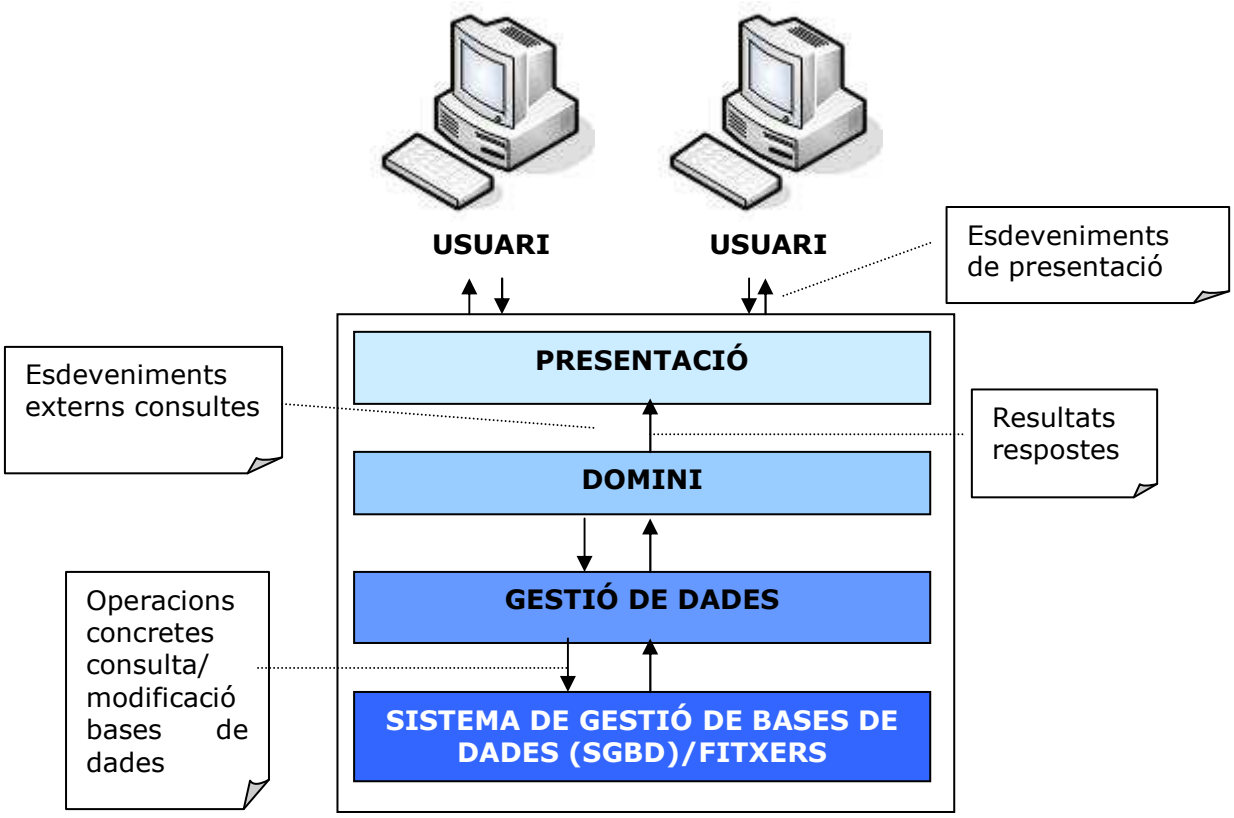
Existeixen vàries maneres de realitzar aquesta divisió, en el cas d'una aplicació web el patró que s'utilitza és de presentació distribuïda. Aquest patró té com a característica que el client té una part de la capa de presentació i el servidor conté tots els demés components del sistema: part de la capa de presentació, la lògica de l'aplicació i l'administració de les dades. Aquesta solució es caracteritza per ser molt simple (fàcil d'implementar), permet una millor presentació i des del punt de vista de l'ús dels recursos es comporta de forma similar a un sistema centralitzat.




*Patró arquitectònic client-servidor*

## Arquitectura en tres capes

Aquest patró arquitectònic està format per tres capes: la capa de Presentació, la capa de Domini i la capa de Gestió de Dades. Les funcions de cadascuna d'elles es pot veure en el gràfic següent:




Bàsicament, la **capa de presentació** és la responsable de la interacció amb l'usuari i per tant s'encarrega de rebre les peticions, ordenar l'execució de les accions, el tractament de diàleg i entorn gràfic amb el client (menús, finestres,etc.) i de comunicar els resultats de les accions a l'usuari final. Per tant, aquesta capa ignora les transformacions que cal fer per donar desposta a les peticions.

La **capa de domini** és la responsable de la implementació de les funcionalitats del sistema. Les seves funcions principals són les de realitza les tasques de rebre les peticions i comprovar-les, canviar l'estat del domini, executar les accions ordenades, encarregar-se de saber quan hi ha consultes, obtenir els resultats i comunica les respostes pertinents. Aquesta capa desconeix totalment on es troben les dades i com li arriben a l'usuari.

I finalment, la **capa de gestió de dades** permetrà que el domini pugui ignorar on estan les dades i adaptar-se al sistema de gestió de bases de dades que s'utilitzi per dur a terme les funcions concretes sobre les dades. Per tant, aquesta capa coneix on es troben les dades i com estan emmagatzemades però desconeix com tractar-les.

El **Sistema de Gestió de Base de Dades** (SGBD) s'encarrega de mantenir una representació persistent i concreta de l'estat del domini.

### 3.1.2 Plataforma física

A l'hora de dissenyar cal tenir en compte la plataforma física que s'emprarà per implementar l'aplicació.

Com s'ha pogut veure en la fase d'especificació, s'ha utilitzat la metodologia orientada a objectes (UML). És per això que s'ha cregut convenient proporcionar un entorn de desenvolupament que permeti utilitzar les característiques de l'orientació a objectes. A continuació s'expliquen les decisions preses respecte a la plataforma escollida.

#### Tecnologia web

A l'hora de dissenyar cal tenir en compte la plataforma física que s'emprarà per implementar l'aplicació. En aquest cas, s'ha decidit utilitzar un framework anomenat **Grails**. Un framework és una estructura de suport definida en la qual un altre projecte de software pot ser organitzat i desnevolupat. Està compost per components personalitzables i intercanviables pel desenvolupament d'una aplicació. És a dir, és una aplicació genèrica incomplerta i configurable a la que podem afagir-li les últimes peces per construir una aplicació concreta. El seu ús és cada vegada més extés avui en dia ja que les avantatges d'utilitzar un framework per al desenvolupament d'aplicacions Web són nombroses. Entre dellases destaquem:

- Desenvolupament més ràpid
- Desenvolupament estructurat

- Reutilització de codi
- Disminueix l'esforç en el desenvolupament
- Permet aprofitar les funcionalitats ja implementades, de manera que "no hem de reinventar la roda"
- Permet concentrar-se directament en la solució del problema
- Permet tenir com a aliat metodologies de desenvolupament àgil (XP, AD)

Grails és doncs un framework pel desenvolupament d'aplicacions web. En l'apartat 4.1.2 es descriurà amb més profonditat.

### Llenguatge de programació

Donat que s'utilitza el framework de Grails, el llenguatge utilitzat per l'aplicació serà **Groovy**. Groovy és un llenguatge de programació dinàmic per a la JVM (*Java Virtual Machine*) que ens permet construir aplicacions web utilitzant la potència de Java i afegint-hi característiques inspirades en d'altres llenguatges com Python, Ruby, Smalltalk i Perl. Groovy afegeix característiques de programació modernes pels desenvolupadors Java amb una corba d'aprenentatge baixa. Millora la productivitat de la programació quan es tracta d'apliacions web, GUIs (*Graphical User Interfaces*), bases de dades o aplicacions de consola. A més, simplifa força l'etapa de probes. Es parlarà de Groovy en més profonditat a l'apartat 4.3.2 d'aquesta memòria.

### Desenvolupament web

Donat que s'utilitza el framework de Grails, el llenguatge de presentació per l'aplicació web serà **Groovy Server Pages (GSP)**. Es tracta d'un llenguatge molt similar al Java™Server Pages (JSP). GSP permet barrejar contingut estàtic i dinàmic en el mateix document. El resultat és un codi generat dinàmicament en HTML, XML o un altre tipus de document en resposta a la requesta del client web.

Un exemple de GSP seria el següent:

```
      <html>

      <head>
            <meta name="layout" content="main" />
            <title>Entrada per usuaris registrats</title>

        </head>
  <body>
    <div class="body">
      <h1>Login</h1>
      <g:if test="${flash.message}">
        <div class="message">${flash.message}</div>
      </g:if>
      <g:form action="autenticar" method="post" >
        <div class="dialog">
          <table>
            <tbody>
              <tr class="prop">
                <td class="name">
                  <label for="login">Usuari:</label>
                </td>
                <td>
                  <input type="text" id="login" name="login"/>
                </td>
              </tr>
              <tr class="prop">
                <td class="name">
                  <label for="password">Contrasenya:</label>
                </td>
                <td>
                  <input type="password" id="password" name="password"/>
                </td>
              </tr>
            </tbody>
          </table>
        </div>
        <div class="buttons">
          <span class="button">
            <input class="save" type="submit" value="Login" />
          </span>
        </div>
      </g:form>
    </div>
  </body>
</html>
```

## Gestor de bases de dades

Tal i com apareixia en el gràfic de l'apartat 3.1.1, s'utilitzarà un sistema gestor de base de dades (SGBD). Els SGBDs serveixen com a interfície de gestió de les dades d'una aplicació i la pròpia aplicació que les utilitza. La base de dades proporciona el suport permanent que emmagatzema aquestes dades i el SGBD és el medi per accedir a ella.

La informació d'una aplicació es pot emmagatzemar també en fitxers i les operacions necessàries per utilitzar-los generalment vénen donades pel sistema operatiu. No obstant, la gestió de fitxers no resulta tan eficient. Aquesta mancança la recullen els sistemes gestors de base de dades proporcionant aquesta interfície

entre les aplicacions i el sistema operatiu. D'aquesta manera, s'aconsegueix accedir a les dades de manera eficaç, de forma segura i senzilla en la implementació.

A l'hora de seleccionar el SGBD es va escollir com a condició que tingués cost zero, que fos capaç de respondre a múltiples peticions que el servidor pugui realitzar de forma persistent i que existís un suport per poder accedir i fer les peticions amb la tecnologia de Java. A més, s'han considerat característiques com la facilitat d'ús en les eines d'administració i utilització per part de l'usuari, que en aquest cas serà l'administrador del sistema. Altre aspecte que es va tenir en compte que fos un SGBD amb una gran comunitat que el fes servir ja que això asseguraria en cert grau que hi hauran bones eines de suport i serà fàcil obtenir informació.

En termes més estrictes podem dir que busquem un SGBD de reconegut **prestigi**, **fiabilitat**, **velocitat**, **rendiment**, **facilitat d'administració** i **connexió amb altres productes**, **ben documentats** i **amb bona evolució** i **suport**.

De tots els Sistemes Gestors de Bases de Dades (SGBD) lliures els que responen millor a aquestes pautes de selecció són *PostgreSQL* i *MySQL*, ambdós candidats pel nostre sistema. **MySQL** és un dels SGBD més populars desenvolupat sota la filosofia de codi obert. El desenvolupa i manté l'empresa MySQL AB, des de gener de 2008 una subsidiària de Sun Microsystems, la qual pertany a Oracle Corporation des de abril del 2009. Tot i ser actualment d'Oracle continua sota la llicencia GPL i per tant pot utilitzar-se gratuïtament i el seu codi font està disponible. **PostgreSQL** també és lliure, però sota llicència BSD, superior a la de MySQL. El seu desenvolupament no és en mans d'un sola empresa sinó que és dirigit per una comunitat de desenvolupadors i organitzadors comercials anomenada PGDG *(PostgreSQL Global Development Group)*.

Ambdós sistemes són solucions que han anat evolucionant molt amb el pas del temps, millorant els seus punts febles aconseguint competir fortament amb les bases de dades de software propietari. Existeix la creença que MySQL és el SGBD més ràpid i que compta amb menys funcions que PostgreSQL. Així mateix, es sol pensar que PostgreSQL és un SGBD més dens i moltes vegades s'associa a la versió en codi obert d'Oracle. No obstant això, aquestes creences són avui en dia obsoletes i incorrectes. MySQL ha recorregut un llarg camí en l'edició de funcionalitats avançades, mentre que PostgreSQL ha millorat enormement la seva velocitat en les darreres versions.

A l'hora de triar entre un SGBD o un altre convé conèixer com es comporten ambdós eines davant d'alguns paràmetres respecte al rendiment, les característiques i el tipus de dades. Escollir quin dels dos sistemes utilitzar és una decisió difícil per a qualsevol analista perquè planteja grans controvèrsies i requereix un anàlisi en profunditat de l'aplicació en la que es vol integrar l'SGBD. Tot seguit analitzem com es comporten ambdós SGBD per a cadascun del paràmetres.

#### 3.1.2.1 Rendiment

És molt difícil donar una comparació precisa en el rendiment dels SGBDs sense parar atenció a la configuració i l'entorn, ja que aquests es poden optimitzar d'acord amb l'entorn en el qual corren. Tant PostgreSQL com MySQL utilitzen diverses tècniques per millorar el seu rendiment.

- **Raw Speed**

El motor MyISAM de MySQL funciona més ràpid que PostgreSQL quan es fan queries simples i quan la concurrència és baixa o segueix certs patrons (per exemple, els inserts es realitzen en taules optimitzades i sense bloquejos, el count (*) és molt ràpid).

El cost de la velocitat del motor MyISAM ve de no donar suport a les transaccions ni claus foranes**,** i no ofereix integritat garantida en les dades, cal controlar l'integritat referencial des de la capa de domini.

PostgreSQL, en canvi, proporciona algunes característiques significatives de rendiment com són:

- L'eficient executor de sentències SQL tan dinàmiques com estàtiques
- Un avançat optimitzador, amb multitud d'opcions
- Indexació parcial, funcional i múltiple combinada
- Millora de la gestió de la cache
- Alta escalabilitat
- I commit asíncron

- **Compressió de dades**

PostgreSQL pot comprimir i descomprimir les seves dades sobre la marxa amb un ràpid sistema de compressió per encaixar més dades en un espai de disc

assignat. L'avantatge de les dades comprimides, a més d'estalviar espai en disc, és que la lectura de dades triga menys.

En canvi, fins a la versió 5.1 de MySQL els motors d'emmagatzematge d'alt rendiment no suporten compressió sobre la marxa. MySQL té un tipus de "query cache" que evita treball a la base de dades per conèixer si una Query ha estat processada recentment. Aquesta característica és valuosa per aquells sistemes que realitzin moltes lectures. No obstant, si hi ha canvis en la taula consultada aquesta utilitat disminueix.

MySQL també suporta compressió a nivell de protocol de xarxa, una opció que pot ser accionada pel client si el servidor ho permet. Això fa que tot el que provingui o viatgi al servidor sigui comprimit .

El motor MyISAM de MySQL suporta compressió de l'índex el qual utilitza per defecte en certa mesura. Una millor compressió pot ser adquirida mitjançant l'ús de l'opció PACK_KEYS. Altres motors d'emmagatzematge estables MySQL per ara no tenen aquesta característica, pel que els seus índexs utilitzen més espai.

- **Multi-processador**

En general, l'avantatge en velocitat de PostgreSQL sobre MySQL es pot veure en un ambient de grans processadors multi-core. No obstant, MySQL ha fet grans avanços en aquest sentit i les versions posteriors a la 5.0 ja solucionen molts problemes de colls d'ampolla que tenien les versions anteriors.

- **Concurrència**

El servidor PostgreSQL funciona amb un únic motor d'enmagatzematge que es comporta molt eficient tant en termes de la utilització d'un maquinari d'alt rendiment com en fer front a la concurrència. MySQL d'altra banda, té una arquitectura de dues capes, la capa principal de SQL i un conjunt de motors d'enregistrament. Sovint quan es comparen els dos SGBD es necessari especificar quin motor està fent servir MySQL per que l'elecció afecta l'adaptabilitat, rendiment i les funcionalitats del SGBD. Els motors d'enregistrament més comuns que es fan servir amb MySQL son: InnoDB per a un complet suport al compliment d' ACID (detallat en el següent punt) i alt rendiment en sistemes sobrecarregats amb grans fluxes de dades accedides concurrentment; i MyISAM per fluxes de dades menys concurrents o altament concurrents però majoritàriament amb accessos de lectura

que no precisen de les funcionalitats ACID. Tanmateix és pot configurar el sistema gestor perquè es faci servir una combinació dels motors disponibles en cas de que l'aplicació ho requereixi i així aprofitar-se de les avantatges de cadascun d'ells.

- **Cumpliment d'ACID**

ACID (*Atomicity*, *Consistency*, *Isolation* and *Durability*) és un model utilitzat per avaluar la intreguitat de les dades en diferents SGBDs. La majoria de SGBDs aconsegueixen les característiques ACID implementant transaccions. Tant PostgreSQL com MySQL compleixen el model ACID, tot i que PostgreSQL és conegut per un enfoc més rigorós en la integritat de les dades. Per establir el motor compatible amb ACID en MySQL per defecte, cal establir la variable *default-storage-engine=innodb* en el seu arxiu de configuració.

- **E/S asíncrona**

PostgreSQL suporta una complerta API asíncrona per ser utilitzada per les aplicacions clients. Aquesta augmenta el rendiment fins a un 40% en alguns casos. Pel contrari, MySQL no té suport a E/S asíncrona tot i que s'han fet creat alguns drivers per millorar-ne el rendiment. A partir de la versió MySQL 5.0 el motor InnoDB fa servir E/S asíncrona simulada: crea un nombre de processos per fer-se càrrec de les operacions d'E/S, tals com la lectura amb avançament (read-ahead).

InnoDB fa servir una novedosa tècnica de descarrega de fitxers anomenada *doublewrite.* Aquesta incrementa la seguretat en la recuperació que succeeix a una caiguda del sistema operatiu o una interrupció de l'energia elèctrica, i millora el rendiment en moltes varietas de Unix al reduir la necessitat d'utilitzar operacions fsync().

- **COUNT(*)**

Les bases tradicionalment transaccionals que implementen el sistema de concurrència MVCC com PostGreSQL són molt més lentes si es comparen amb les no-transaccionals com MyISAM que és utilitzat per MySQL. Aquesta lentitut és deguda a què en lloc de dir les files utilitzant un índex d'exploració ha de recórrer tota la taula seqüencialment. El motor MyISAM utilitza un índex per COUNT (*) i

emmagatzema el resultat del count en la cache del compte, cosa que el fa molt més ràpid.

#### 3.1.2.2 Característiques

MySQL i PostgreSQL tenen un gran conjunt de característiques que augmenten la integritat de les dades, funcionalitat i rendiment. Les característiques incloses en una base de dades poden ajudar a millorar el rendiment, la facilitat d'ús, la funcionalitat o l'estabilitat.

- **Facilitat d'ús**

S'afirma que MySQL té més facilitat d'ús que PostgreSQL però no hi ha cap raó consistent i el raonament es basa en que MySQL està més extés i per tant hi ha més gent que sap operar amb ell.

- **Insert Ignore/Replace**

MySQL suporta declaracions INSERT IGNORE i REPLACE les quals fan un insert si una fila no existeix i reemplaça la fila actual (i res si no és així). PostgreSQL no suporta cap d'aquestes declaracions i suggereix l'ús de procediments emmagatzemats per pal·liar aquesta mancança. No obstant, fent servir procediments emmagatzemats existeixen grans deficiències en cas de necessitar-ho:

Només es pot insertar un valor únic a la vegada. Aquesta és una limitació important de rendiment i també comporta problemes de concurrència. INSERT IGNORE i REPLACE pot manegar diversos valors i insertar molt millor.

- **Constraints**

Tant MySQL i PostgreSQL suporten Not-Null, Unique, Clau primària i claus foranes. No obstant això, MySQL no suporta la comprovació de limitació (Check constraint) mentre que PostgreSQL sí que ho ofereix.

La clàusula `DEFAULT value` en l'especificació del tipus de dada en una taula indica el valor per defecte per una columna. En MySQL hi ha una excepció: el valor per defecte ha de ser una constant, no pot ser una funció o una expressió.

PostgreSQL permet a qualsevol funció marcada com IMMUTABLE o STABLE retornar un valor que es col·locarà com *default value* per una columna d'una taula. A MySQL actualment, now() es l'única funció que es pot fer servir com a valor per defecte en una taula MySQL i només es pot aplicar en una columna per taula, en files de tipus TIMESTAMP.

L'aplicació GWAL, degut a què té molts atributs en les entitats de la capa de domini, farà servir moltes d'aquestes constraints. Per exemple, no permetrem que quan es doni d'alta un usuari al sistema per mitjà del registre de la web quedin a null els camps del login i la contrassenya, i a més es comprovarà que el login sigui únic. Per tant, en aquest cas podem emprar les constraints not null i unique als camps pertinents de la taula usuari. Hi ha també un gran nombre d'entitats que precisen checks a la base de dades, com seria el cas de que impedeix que la periodicitat d'un taller sigui menor que 1. Com ja s'ha comentat MySQL no ens permet afegir la comprovació en la base de dades. No obstant, amb l'elecció del framework Grails, decisió que es pren en l'apartat 4.1.2 això es pot fer a nivell de domini gràcies a la declaració de constraints amb GORM.

- **Procediments emmagatzemats**

Tant MySQL com PostgreSQL suporten procediments emmagatzemats. El primer llenguatge de consulta per PostgreSQL, PL/pgsql, és similar al d'Oracle PL/SQL. PostgreSQL també suporta procediments emmagatzemats en molts altres llenguatges, entre ells Python, Perl, TCL, Java i C.

MySQL segueix el SQL: 2003 per a la sintaxi de rutines emmagatzemades, que també és utilitzat per IBM DB2.

- **Triggers**

Tant MySQL com PostgreSQL suporten triggers. Un disparador PostgreSQL pot executar qualsevol funció definida per l'usuari des de qualsevol dels seus llenguatges de procediment, no només PL / pgsql.

No obstant això, la sintaxi per a la definició dels disparadors en PostgreSQL no és tan directa com en MySQL, on els triggers són activats només per comandes

SQL. Amb PostgreSQL és necessària una definició d'una funció amb la devolució del tipus de dades específic.

#### 3.1.2.3 Replicació i Alta disponibilitat

La replicació és la capacitat d'un sistema de gestió de base de dades de duplicar les seves dades emmagatzemades per fer còpies de seguretat i una manera de prevenir-ne la inactivitat de la base de dades. Tots dos SGBDs, PostgreSQL i MySQL, suporten replicació.

#### 3.1.2.4 Tipus de dades

PostgreSQL no té un tipus de dada sencer sense signe, però té molt més suport de tipus de dades en diversos aspectes: compliment de normes, el tipus de dades booleà, mecanismes de tipus de dades definits per l'usuari i tipus nadius i contribuïts.

Tant MySQL i PostgreSQL suporten subconsultes, però en MySQL algunes formes poden tenir impacte notable en el seu rendiment. Això serà corregit en la propera versió 6.0.

#### 3.1.2.5 Indexació avançada

Els mètodes d'indexació avançada permeten als sistemes de bases de dades optimitzar les consultes per tal d'aconseguir un major rendiment. MySQL y PostgresSQL tenen característiques similars tot i que MySQL no és compatible amb índexs parcials (índexs construïts sobre un subconjunt d'una taula).

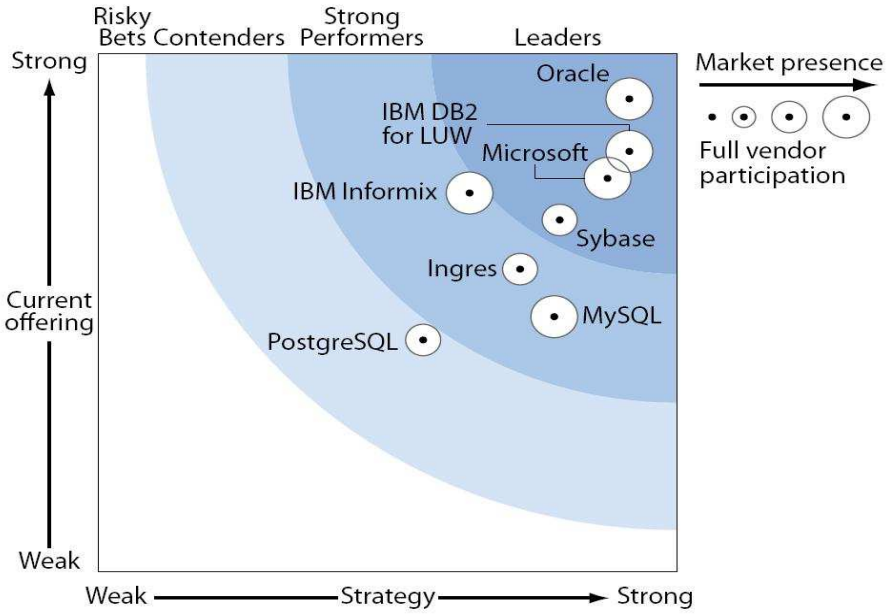
#### 3.1.2.6 Conclusió

Existeixen molts estudis comparatius respecte la velocitat d'ambdós gestors. No obstant, quan aquests estudis es duen a terme moltes vegades ja han aparegut versions posteriors de les eines i els estudis queden desfasats. Si bé és cert que alguns estudis conclouen que PostgreSQL escala millor amb un nombre elevat de processadors amb alta concurrència, hi ha altres escenaris on MySQL és superior.

Si preguntéssim a algunes de les empreses (Amazon,Wikipedia,...) que fan servir MySQL com a gestor de les seves aplicacions web ens dirien probablement que té moltes eines afegides al seu voltant i que es més lleuger i respon més ràpid pels accessos de lectura que són els predominants en les aplicacions web.

Una bona tendència per decidir-se és testejar la mateixa aplicació tant amb un gestor com l'altre. No obstant, això requereix un temps ampli en el que es prova l'aplicació i s'analitza el funcionament amb els dos SGBD. Encara que haguéssim pogut adoptar aquesta estratègia no ho hem fet ja que ens sortiríem del temps previst per la realització del projecte. A més els resultats només servirien per comparar-los en cas d'implantar l'aplicació GWAL en la gestió d'una entitat lúdica real i deixar-la funcionant un temps per que es succeïssin situacions i comportaments diferents. L'àmbit de l'aplicació del nostre projecte, la mitjana empresa i entorn web, ja ens permet estimar que el nombre d'usuaris del sistema i el nombre d'escriptures que hi haurà no serà molt alta i majoritàriament la web haurà de suportar moltes lectures.

Per acabar es mostra un gràfic resum realitzat per l'empresa Forrester Wave Database Systems al 2009.




Podem veure que MySQL està en millor posició respecte PostgreSQL. Les variables Current Offering, Strategy i Market Presence que es veuen en el gràfic es descriuen acuradament a continuació:

| | Forrester's Weighting | IBM DB2 for LUW | IBM Informix | Ingres | Microsoft | MySQL | Oracle | PostgreSQL | Sybase |
|---|---|---|---|---|---|---|---|---|---|
| CURRENT OFFERING | 50% | 4.04 | 3.64 | 2.90 | 3.77 | 2.44 | 4.54 | 2.22 | 3.37 |
| Data types and data integrity | 10% | 4.60 | 4.36 | 2.76 | 4.36 | 2.81 | 4.84 | 2.13 | 3.78 |
| Performance, scalability, and VLDB | 15% | 4.21 | 3.81 | 2.10 | 3.06 | 2.30 | 5.00 | 1.80 | 2.81 |
| Application development | 15% | 4.73 | 3.99 | 3.16 | 4.61 | 2.72 | 4.70 | 2.97 | 3.01 |
| Database availability | 20% | 3.94 | 4.22 | 3.46 | 4.40 | 1.46 | 5.00 | 1.81 | 4.22 |
| Database security | 15% | 3.36 | 1.95 | 2.34 | 3.43 | 1.52 | 3.56 | 1.89 | 2.86 |
| Platform support | 10% | 2.96 | 2.88 | 3.09 | 1.60 | 4.74 | 3.27 | 4.16 | 3.09 |
| Database administration | 15% | 4.32 | 4.05 | 3.24 | 4.17 | 2.77 | 4.94 | 1.52 | 3.55 |
| STRATEGY | 50% | 4.58 | 3.20 | 3.65 | 4.40 | 3.93 | 4.58 | 2.80 | 3.98 |
| Product strategy | 50% | 4.75 | 3.00 | 3.50 | 5.00 | 3.25 | 4.75 | 2.80 | 3.95 |
| Commitment | 50% | 4.40 | 3.40 | 3.80 | 3.80 | 4.60 | 4.40 | 2.80 | 4.00 |
| Pricing and licensing | 0% | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| MARKET PRESENCE | 0% | 4.19 | 4.11 | 3.01 | 4.60 | 4.06 | 4.60 | 3.44 | 3.83 |
| Company financials | 20% | 4.00 | 3.70 | 5.00 | 4.00 | 5.00 | 4.00 | 5.00 | 4.00 |
| Adoption | 20% | 4.30 | 2.90 | 2.60 | 5.00 | 5.00 | 5.00 | 3.60 | 3.60 |
| Training and certification | 20% | 3.70 | 3.95 | 1.40 | 5.00 | 3.30 | 5.00 | 3.00 | 2.90 |
| Systems integrators and partners | 20% | 3.95 | 5.00 | 3.05 | 5.00 | 4.00 | 5.00 | 2.60 | 4.65 |
| Global presence | 20% | 5.00 | 5.00 | 3.00 | 4.00 | 3.00 | 4.00 | 3.00 | 4.00 |

All scores are based on a scale of 0 (weak) to 5 (strong).

Source: Forrester Research, Inc.

Després d'aquest extens anàlisi donem per resolta l'elecció. En l'aplicació GWAL farem servir MySQL. Encara que alguns estudis diuen que PostgreSql és millor per entorns on la integritat de les dades és fonamental, a nivell de rendiment MySql és millor. Concretament quan utilitzem el tipus de motor MyIsam el rendiment de MySql és molt millor. Aquest serà el tipus d'Engine amb el que definirem totes les taules en l'aplicació GWAL. El motor InnoDb és més lent i està orientat a aplicacions on l'actualització e integritat és més important. Com ja hem comentat en un dels punts anteriors l'integritat referencial estarà ben controlada per la capa de domini mitjançant GORM (Grails Object Relational Mapping).

En qualsevol cas, degut a que més endavant farem servir el patró "Domain Model (amb consultes) i gestió de la persistència amb Hibernate", la base de dades generada podrà ser desplegada sobre qualsevol SGBD que creguem oportú, necessitant només introduir canvis en un parell de fitxers de configuració del servidor d'aplicacions web.

### 3.1.3 Disseny extern

El disseny extern de la capa de Presentació consisteix en el disseny dels elements que l'usuari veu, sent i toca a l'interactuar amb el sistema. Concretament, consisteix en la definició de:

- **Mecanismes d'interacció,** a través dels quals l'usuari pot fer peticions al sistema.
- **Presentació de la informació**, que és la manera de mostrar a l'usuari el resultat d'aquestes peticions.

Per l'èxit de qualsevol sistema d'informació que es desenvolupi resulta crucial que la capa de presentació sigui agradable i pràctica per l'usuari. Per això és necessari seguir unes normes comunes que s'han de tenir en compte per a què el sistema resultant sigui usable per a l'usuari. Tot això ho veurem amb més detall a l'apartat 3.2.2 dedicat a l'usabilitat. En el desenvolupament de l'aplicació i en el disseny de les pantalles s'ha intentat tenir en compte tots aquests criteris per millorar la interacció amb el sistema per part de l'usuari.

Per a què tota l'aplicació presenti un aspecte homogeni s'han definit plantilles de disseny (templates), que encapsulen tots els elements comuns en una pàgina: capçalera, menú de navegació, fons, peu de pàgina, etc. D'aquesta manera es facilita la tasca de crear pàgines, ja que a una pàgina buida se li pot aplicar una plantilla ja definida, i a la vegada es manté la consistència en l'aparença (*'look and feel'*) de l'aplicació. Les plantilles ofereixen a més una gran avantatge al modificar un element comú: tot i que el canvi afecti a totes les pàgines de l'aplicació, la modificació tan sols haurà de realitzar-se una vegada sobre la plantilla, i automàticament totes les pàgines que utilitzin la plantilla heretaran el canvi.

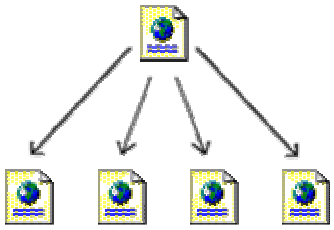
En els següents apartats es detalla l'estructura que tindrà l'aplicació, els components visuals i exemples de pantalles que formen part del disseny extern de l'aplicació, així com l'apartat de gestió d'usuaris, que s'ha cregut convenient afegir-la en aquest capítol.

## Estructura del lloc

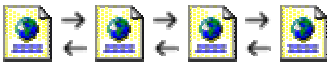
Definir l'estructura del conjunt de pàgines web és important, ja que una bona estructura permetrà a l'usuari visualitzar tots els continguts d'una manera fàcil i clara, mentre que un conjunt de pàgines web amb una estructura incorrecta produirà en l'usuari una sensació d'estar perdut, no trobarà ràpidament el que cerca i acabarà per abandonar l'aplicació.

Per planificar l'estructura de la web s'ha de tenir en compte que aquesta depèn del contingut. Bàsicament existeixen les següents estructures:

- **Jeràrquica.** Es tracta de la típica estructura d'arbre, en el que l'arrel és la pàgina principal, on s'exposen les diferents seccions que contindrà el nostre lloc. La selecció d'una secció ens condueix a una llista de subtemes que poden no dividir-se. Aquest tipus d'organització permet a l'usuari de conèixer en quin lloc de l'estructura es troba, a més de saber que, a mesura que s'endinsa en l'estructura, obté informació més específica i que la informació més general es troba en els nivells superiors.

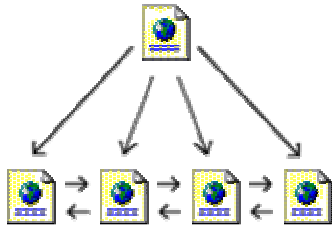


- **Lineal.** L'estructura lineal és la més simple de totes. La manera de recórrer-la és la mateixa que si estiguéssim llegint un llibre, de manera que estant en una pàgina, podem anar a la següent pàgina o a l'anterior. Aquesta estructura és molt útil quan volem que l'usuari segueixi un camí fixe i guiat. Aquest tipus d'estructura seria vàlida per tutorials d'aprenentatge o tours de visita guiada.

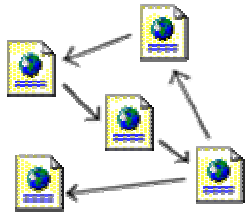


- **Lineal amb jerarquia.** Aquest tipus d'estructura és una barreja de les dues anteriors, els temes i subtemes estan organitzats d'una forma jeràrquica, però es pot llegir tot el contingut d'una forma lineal si es desitja. Aquest

permet tenir el contingut organitzat jeràrquicament i simultàniament poder acceder a tota la informació de forma linial.



- **Xarxa.** L'estructura de xarxa és una organització en la que aparentment no hi ha cap ordre establert, les pàgines poden vincular-se unes a unes altres sense cap ordre aparent. Aquest tipus d'organització és la més lliure, però pot resultar complicada per l'usuari perquè pot perdre's.



Per la nostra aplicació, s'ha decidit seguir una estructura en forma **jeràrquica**, per adaptar-se millor al contingut que ofereix l'aplicació.

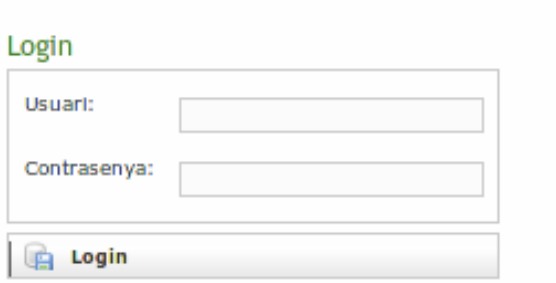
## Capçalera

La capçalera és un element molt important en el disseny d'una pàgina web ja que identifica l'aplicació i està present en totes les pàgines. Inclou el logotip i la barra de navegació amb les icones i vincles per tornar a la pàgina principal en qualsevol moment. També inclou a la dreta la part d'identificació de l'usuari i a l'esquerra s'han habilitat opcions per ampliar el contingut de la pàgina central ocultant el menú esquerra i la informació addicional dreta.

Com s'explica en el capítol 4, aquest element és configurable per adaptar-lo a les característiques del centre.


## Àrea d'identificació

L'àrea d'identificació és un element que apareix en clicar a l'opció "Login" de la part dreta de la capçalera, sempre que no estigui ja connectat amb un perfil concret. Aquesta àrea permet a l'usuari entrar a l'àrea de gestió del sistema amb un rol determinat. Aquest component, doncs, valida que l'usuari i la contrassenya introduïts siguin els correctes i en aquest cas redirecciona a la pàgina corresponent del seu perfil.




En el cas que l'usuari intenti accedir a una zona en la que no té permisos li sortirà el següent missatge d'error:

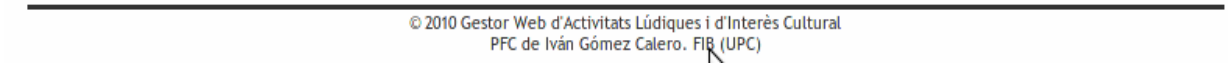
## Menú

El menú és un element dinàmic en l'aplicació, ja que es mostra diferent segons el perfil de la persona que estigui connectada. Existeixen tants menús com perfils diferents hi ha definits en l'aplicació.

Hi ha dos menús al sistema GWAL. El que té format horitzontal permet situar a l'usuari en els diferents blocs d'informació o de gestió de l'aplicació. El segon menú, situat al lateral esquerra permet seleccionar les diferents opcions d'aquell bloc, per accedir a informació més concreta.

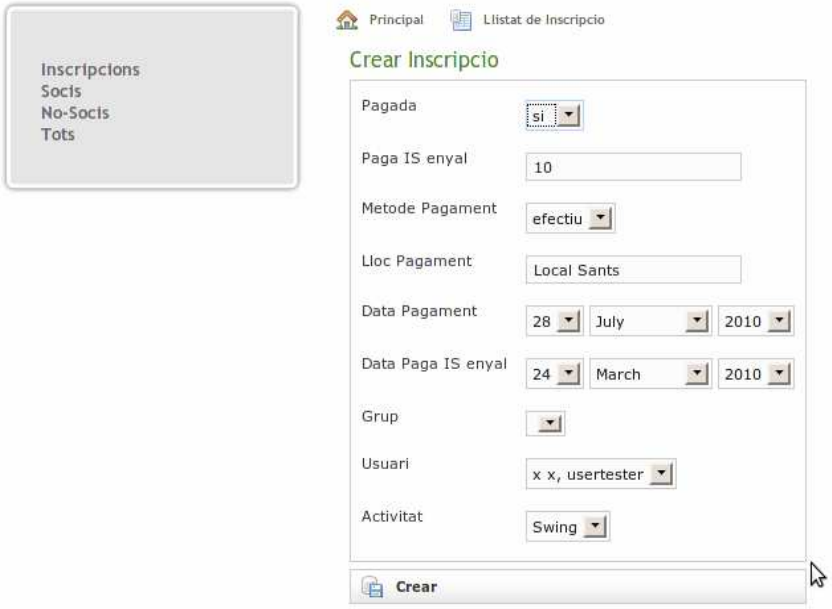

## Peu de pàgina

El peu de pàgina també és un element comú a totes les pàgines de l'aplicació i inclou en el nostre cas l'any actual, el nom de l'aplicació i el dissenyador de la mateixa. Com s'explica en el capítol 4, aquest element és configurable per adaptar-lo a les característiques del centre.



## Formularis

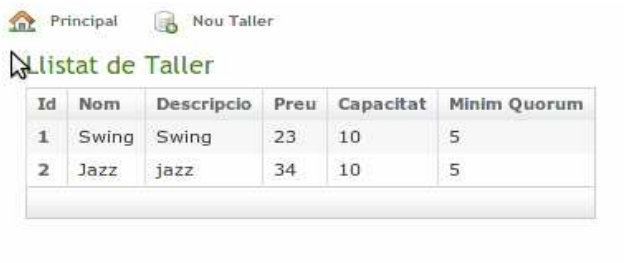
Els formularis són pàgines que permeten gestionar els diferents objectes de l'aplicació. Són la manera de permetre realitzar insercions, modificacions, consultes i eliminacions. A continuació es mostra un exemple de formulari que correspon a la creació d'una inscripció desde la vista d'un administratiu.




## Llistats

Els llistats a l'aplicació GWAL ens ofereixen una forma ordenada de veure la informació requerida. El format que es fa servir és el d'una taula que té camps ordenables i que conté 10 elements. La resta d'elements queden en pàgines succesives ja que els llistats són pàginats. Per facilitar la tasca de llegir els resultats les files de la taula es mostren en colors alternats.

El format general d'un llistat és el que es mostra a continuació:

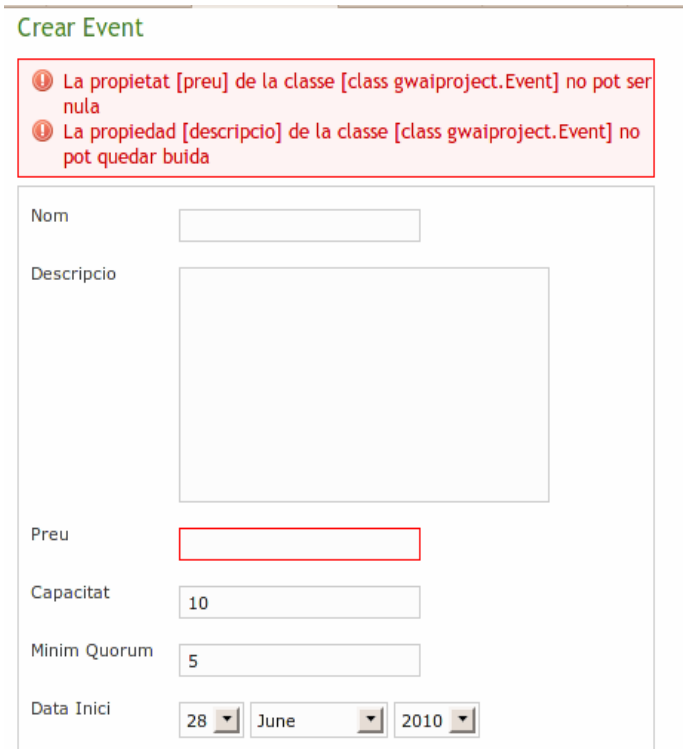



## Missatges informatius

Mostren un missatge informatiu a l'usuari quan s'ha realitzat una acció. Estan situats a la part superior de la pantalla per facilitar la seva visualització i es mostren en color blau per diferenciar-los dels missatges d'error.


## Missatges d'error

Els missatges d'error es mostren per indicar les excepcions del sistema. Es mostren en color vermell per donar énfasi al fet de que s'ha produit un error. Aparèixen en la part superior indicant quins camps estan en conflicte i a més es recuadra el camp en vermell.


Crear Event
La propietat [preu] de la classe [class gwaiproject.Event] no pot ser nula
La propiedad [descripcio] de la classe [class gwaiproject.Event] no pot quedar buida
Nom
Descripcio
Preu
Capacitat
10
Minim Quorum
5
Data Inici
28
June
2010

### 3.1.4 Usabilitat de la web

La usabilitat és un tema que està tenint cada vegada més importància en el desenvolupament d'aplicacions informàtiques i sobre el qual no es troba actualment massa documentació en les assignatures de la facultat. És per això que s'ha considerat interessant aprofundir una més en la temàtica, que degut a que els sistemes estan dirigits a un públic cada vegada més ampli, està destacant com una propietat fonamental per a l'èxit d'un producte software.

En aquest apartat es definirà el concepte d'usabilitat orientada a la web, s'analitzaran els atributs que defineixen la usabilitat, els criteris que es poden aplicar a nivell de pàgina, de continguts i de lloc web per tal d'aconseguir oferir als usuaris finals una aplicació web fàcil d'usar i d'aprendre.

Pel desenvolupament de la nostra aplicació s'ha mirat de tenir en compta la majoria de regles pràctiques i recomenacions que tot seguit veurem.

#### Definició d'usabilitat

La usabilitat d'un sistema (producte o servei) es defineix com la mesura en la qual pot ser usat per aconseguir objectius concrets amb efectivitat, eficiència i satisfacció. Més concretament, la norma ISO 9241:11 de 1993 defineix la usabilitat com '**la facilitat d'ús d'una aplicació informàtica**'.

En termes generals, la usabilitat és aplicable a qualsevol objecte que faci servir una persona, definint la facilitat amb la qual el mateix és utilitzat i amb el que és après el seu funcionament.

Així, es considera que la usabilitat està relacionada amb un conjunt de ciències dedicades a la adaptació dels objectes al ser humà, com l'ergonomia, el disseny industrial i, dins del camp de la informàtica, la interacció persona - ordinador i el disseny de interfícies d'usuari.

Pel que fa a l'entorn web, la usabilitat contempla un conjunt de tècniques que ajuden a les persones a realitzar tasques en el entorn gràfic de la interfície d'usuari d'una aplicació web. Parlem d'aplicació web ja que l'usabilitat no afecta

només a pàgines aïllades, sinó al conjunt de totes les pàgines que componen un lloc web, tenint en compte aspectes com la navegació entre elles, la facilitat de trobar un element donat dins el lloc, el nombre d'errors del sistema, la legibilitat dels texts, la correcta transmissió de la informació, l'accés a aquesta per tots els possibles usuaris, etc.

La usabilitat és la responsable que un lloc web compleixi de forma correcta les expectatives amb les que ha estat creat, sempre **des del punt de vista de l'usuari final**, dels visitants de les pàgines, ja que són ells els que faran que un lloc web resulti o no exitós.

Una aplicació considerada "usable" haurà de reunir una sèrie de característiques, entre les que podem destacar:

- Ha de ser **fàcilment entendible** per l'usuari final.
- Ha de ser **intuïtiva**, oferint a l'usuari cada cosa on aquest espera que sigui, de tal manera que la navegació i la realització de tasques es produeixi de forma lògica i senzilla.
- Ha de ser **ràpida**, permetent a l'usuari la ràpida visualització de les pàgines del lloc web, la fàcil localització de la informació cercada i la realització en poques passes de la tasca desitjada.
- Ha d'estar **lliure d'errors** o, si n'hi ha, informar a l'usuari quin tipus d'error s'ha produït i perquè, permetent-li recuperar-se de l'error amb comoditat i rapidesa.
- Ha de **proporcionar satisfacció a l'usuari**, fer-li sentir bé davant de l'ordinador, donar-li la impressió que en tot moment és ell qui controla els processos.
- Ha de **facilitar al màxim el seu aprenentatge**, de tal manera que l'usuari pugui reconèixer l'abans possible el sistema de navegació i el funcionament del lloc.
- Ha de ser **agradable**, estèticament parlant, dissenyada amb una paleta de colors adequada, que proporcioni un entorn de treball visualment relaxat.

Aquestes característiques són algunes de les que defineixen un lloc web usable. A continuació es definiran els atributs que defineixen la usabilitat.

## Atributs d'usabilitat

La usabilitat és una qualitat massa abstracta com per ser mesurada directament. Per poder estudiar-la habitualment es descomposa en els següents 5 atributs bàsics:

- **Facilitat d'aprenentatge**: Com de fàcil resulta aprendre la funcionalitat bàsica del sistema, per tal de ser capaç de realitzar correctament la tasca que desitja realitzar l'usuari. Es mesura generalment pel temps empleat amb el sistema fins a ser capaç de realitzar certes tasques en menys d'un temps donat.

- **Eficiència**: El nombre de transaccions per unitat de temps que l'usuari pot realitzar usant el sistema. El que es cerca és la màxima velocitat de realització de tasques de l'usuari. Quant més gran és l'usabilitat d'un sistema, més ràpid és l'usuari en utilitzar-lo, i la feina es realitza amb major rapidesa.

- **Record en el temps**: Per usuaris intermitents (que no utilitzen el sistema regularment) és vital ser capaços d'utilitzar el sistema sense haver d'aprendre com funciona partint de zero cada vegada. Aquest atribut reflexa el record sobre com funciona el sistema que manté l'usuari, quan torna a emplear-lo després d'un període de no utilització.

- **Taxa d'errors**: Aquest atribut contribueix de forma negativa a la usabilitat d'un sistema. Es refereix al nombre d'errors comesos per l'usuari mentre realitza una determinada tasca. Un bon nivell d'usabilitat implica una tassa d'errors baixa.

- **Satisfacció**: Aquest és el atribut més subjectiu. Mostra la impressió subjectiva que l'usuari obté del sistema.

Alguns d'aquests atributs no contribueixen a la usabilitat del sistema en la mateixa direcció, podent donar-se el cas que l'augment d'un d'ells tingui com efecte la disminució d'un altre. Per exemple, això pot passar amb la facilitat d'aprenentatge i l'eficiència.

## Criteris d'usabilitat en el disseny de pàgines

A continuació s'enumeren els principals criteris a seguir si es desitja desenvolupar una pàgina web usable:

### Criteris generals

- La navegació s'ha de minimitzar respecte al contingut de la pàgina. Es molt recomanable seguir la distribució següent: contingut -> 50 – 80%, navegació –> 20%.
- Utilitzar espais en blanc millor que línies com a element de separació.
- És recomanable utilitzar el logo com a vincle a la pàgina principal.
- No utilitzar una amplada fixa per les taules i els frames.
- La pàgina no ha de trigar més de 10 segons en carregar-se.

### Links (enllaços)

- Els links han de ser curts: només s'ha d'incloure la informació important (2-4 paraules).
- Evitar el link 'Faci clic aquí'. El recomenable és utilitzar descripcions dels links, és a dir, s'ha d'escriure com si no hi haguessin links, triant les paraules significatives.
- S'ha de comprovar el funcionament de tots els links.

### Fulls d'estil

- Las pàgines han ser llegibles sense la fulla d'estils.
- No s'han d'utilitzar més de 2 fonts (màxim 3 en el caso de text especial, com codi de programa)

### Tipografia

- No abusar de la tipografia negreta o cursiva.
- És recomanable no escriure títols en majúscules. La vista està acostumada a fixar-se en la part superior de les lletres pel que si posem totes les paraules en majúscules estarem obligant a realitzar un esforç superior que pot resultar molest a l'usuari.

### Frames

- S'ha d'evitar en la mesura del possible l'ús de frames.

### Consistència / Coherència

- L'usuari adquireix habilitat a través de la repetició de tasques. Per això, els botons o links genèrics han de romandre sempre a la mateixa ubicació en totes les pàgines.
- L'estructura i el contingut de la pàgina han de ser coherents. Les pàgines han d'estar ordenades i s'ha de mantenir el màxim espai en blanc.

### Color

- No s'han d'emprar més de 5 ó 6 colors per pàgina.
- No s'han de codificar objectes petits en blau perquè aquest color no es processa en el centre de la retina.

## Criteris d'usabilitat en el disseny del contingut

Tot seguit es mostren les recomanacions d'usabilitat per dissenyar el contingut de les pàgines web de forma satisfactòria:

### Escriure er la web

- Cal simplificar el contingut. S'ha d'escriure el 50% del text que s'utilitzaria per redactar el mateix en una publicació, ja que llegir sobre pantalla és un 25% més lent que sobre paper.
- L'usuari ha de trobar la informació amb un 'cop de vista': no s'han d'utilitzar grans blocs de text, sinó emprar paràgrafs curts, llistats i subcapçalera.
- Usar links per dividir la informació en múltiples pàgines.
- Controlar l'ortografia i la gramàtica.
- Destacar les paraules importants.

**Llegibilitat**

- Utilitzar colors amb fort contrast entre el text i el fons.
- Utilitzar colors clars pel fons i si tenen algun motiu, que sigui molt discret.
- Utilitzar text estàtic i suficientment gran per a ser llegit.
- S'han d'evitar els paràgrafs en majúscula, ja que es llegeixen un 10% més a poc a poc.

**Temps de resposta**

- És aconsellable indicar el tipus de fitxer i mida al fer una descàrrega.

**Imatges / Gràfics**

- Cal minimitzar l'ús de gràfics per pàgines transaccionals.
- Si s'han d'utilitzar gràfics, usar imatges petites que se carreguin ràpidament i, si és possible, repetir-les. No és aconsellable més de 20K en imatges en una pàgina.

- Minimitzar el nombre de colors en les imatges.
- Utilitzar GIF's per a imatges amb pocs colors i JPEG per a fotografies.
- Incloure l'etiqueta ALT en les imatges (text alternatiu).
- Informar l'usuari si la pàgina a la que va a accedir conté una imatge molt gran.

**Qualitat**

- Assegurar que la informació de contacte està situada en algun lloc destacat.
- Mantenir el contingut actualitzat.
- Indicar 'nou' per a la informació nova.
- Fer revisions contínues per cercar i eliminar links obsolets.
- No cometre errors ortogràfics.

## Criteris d'usabilitat en el disseny del lloc web

A continuació s'enumeraran les principals regles pràctiques d'usabilitat que s'haurien de seguir per dissenyar el lloc web:

**Homepage**

- Ha d'establir la identitat i missió del lloc i donar una perspectiva general del contingut.
- La informació important ha d'aparèixer en una pantalla, sense necessitat de fer scroll.
- Ha de mostrar les principals àrees del lloc.
- Ha d'establir credibilitat i confiança.

- Per llocs de més de 30 pàgines, es recomanable tenir una funció de cerca.

**Pàgines de contingut**

- Han de contenir informació específica.
- Han de mantenir el mateix disseny gràfic.
- Totes les pàgines han de tenir títol i aquest ha de ser únic.
- El logo de cada una de las pàgines és una bona opció per a crear un link a la homepage.

**Navegació**

- El lloc ha de tenir una estructura clara.
- La navegació ha de contestar a les següents preguntes:
    - On sóc?
    - On puc anar?
- S'han de duplicar els elements de navegació si es considera necessari.
- No s'han de crear pàgines que siguin un cul de sac.

**Navegador web**

- S'han de visualitzar les pàgines en diferents navegadors per assegurar la correcta visualització.
- S'ha de desenvolupar amb independència de la resolució, no s'han d'emprar tamanys fixes per realitzar les pàgines web.

## Conclusió

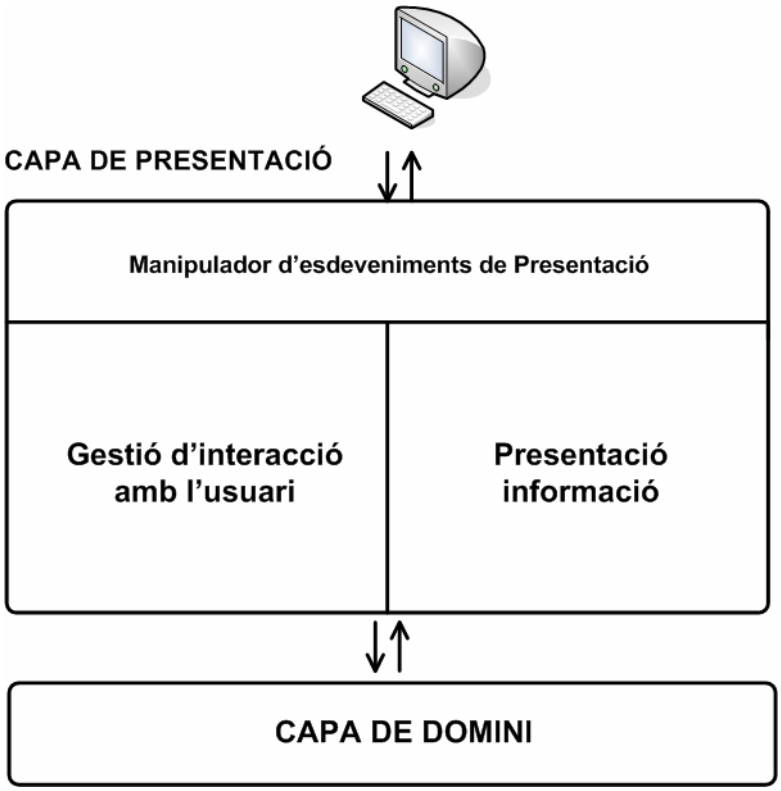
La usabilitat és un element que cal tenir en compte en el desenvolupament d'un lloc web, ja que la seva aplicació ens aportarà molts beneficis, dels que se'n destaquen:

- **Reducció dels costos de manteniment del lloc web**, en oferir al públic un producte homogeni i funcional des del principi.
- Obtenció de sistemes més fàcils d'entendre, usar i recordar, amb el que **s'augmenta la productivitat dels usuaris.**
- **Major qualitat del producte final**, amb el que l'acceptació del mateix per part dels usuaris finals es millor i més ràpida.
- **Augment de la satisfacció dels usuaris** i disminució de l'estrès ocasionat per l'ús de l'aplicació.
- **Reducció del temps d'execució de tasques** per part dels usuaris.

### 3.1.5 Disseny intern

El disseny intern de la capa de presentació consisteix en dissenyar tots els mecanismes que recullen, processen i donen resposta a les peticions dels usuaris.

A continuació es mostra un diagrama on es detalla l'estructura lògica de la capa de presentació.




Està composta per quatre parts:

- El **Manipulador d'Esdeveniments de Presentació**, és un conjunt de Interfícies d'Usuari Gràfiques (conegudes com GUI) basades en esdeveniments, que permeten realitzar la comunicació entre l'usuari i la Capa de Presentació. En el cas d'aquest projecte, les interfícies seran pàgines GSP visualitzades mitjançant un navegador. El navegador és el que s'encarrega de recollir els esdeveniments realitzats per l´usuari i comunicar-los a la capa de presentació.

- La **Gestió d'Interacció amb l'Usuari**, controla la comunicació dels esdeveniments de presentació del receptor, processa aquests esdeveniments i en genera d'altres externs als objectes que els han de

processar. En aquest projecte, la gestió d'esdeveniments d'interacció amb l'usuari la realitza la pàgina que conté les funcions en el controlador de domini, que està associat a la seva pàgina GSP corresponent.

- La **Presentació de la Informació** gestiona la recepció de les dades a presentar a l'usuari i la seva presentació en un determinat format. En el sistema que ens ocupa assumiran aquesta responsabilitat els controls html els quals generen estructures en codi html per evidenciar la informació obtinguda.

- La **Comunicació amb la capa Domini**, consisteix en enviar els esdeveniments externs a processar i rebre respostes a aquests aconteixements.

## 3.2 Capa de Domini

La capa de Gestió de dades és el component del sistema software encarregat de rebre els esdeveniments, controlar la validesa de los dades i comunicar els resultats a la capa de presentació. El disseny d'aquesta consistirà en definir els patrons de disseny utilitzats.

### 3.2.1 Patrons de disseny utilitzats

Un patró de disseny ofereix un esquema per refinar els subsistemes o components d'un sistema software, o les relacions entre elles. Descriu l'estructura d'una solució a un problema que apareix repetidament i l'estructura de components que es comuniquen entre ells. El patró de disseny resol un problema de disseny general en un context particular.

#### Patró controlador

La major part dels sistemes reben esdeveniments externs que han de ser rebuts per algun objecte que executi les accions corresponents. El problema plantejat consisteix en identificar quins objectes són els responsables de tractar els esdeveniments externs.

La solució proposada pel patró controlador consisteix en assignar la responsabilitat comentada a un objecte controlador d'una certa classe. Existeixen diferents tipus de patrons controlador:

- Façana-Sistema: Un objecte que representa tot el sistema.
- Façana-Empresa: Un objecte que representa tot el domini.
- Paper: Un objecte que representa un actor.
- Cas d'ús: Un objecte que representa una instància d'un cas d'ús.
- Transacció: Un objecte que representa un esdeveniment extern.

Per un domini com el que tenim, s'ha considerat convenient utilitzar un controlador Façana. Però per facilitar la comprensió s'ha preferit utilitzar una adaptació d'aquest model, de manera que el controlador representa un subconjunt del domini. D'aquesta forma, el controlador rep tots els esdeveniments externs relacionats amb aquesta part del domini. Un exemple seria el controlador Inscripció que rebria els esdeveniments relacionats amb el domini de les inscripcions, és a dir: alta inscripció, baixa inscripció, modificació inscripció, consulta inscripció, etc.

Els models Façana presenten en general un nivell de cohesió baix (grau de relació entre les diverses responsabilitats d'una classe). Amb l'adaptació d'aquest model s'aconsegueix un nivell de cohesió moderada i disminueix l'acoblament entre classes.

### 3.2.2 Normalització

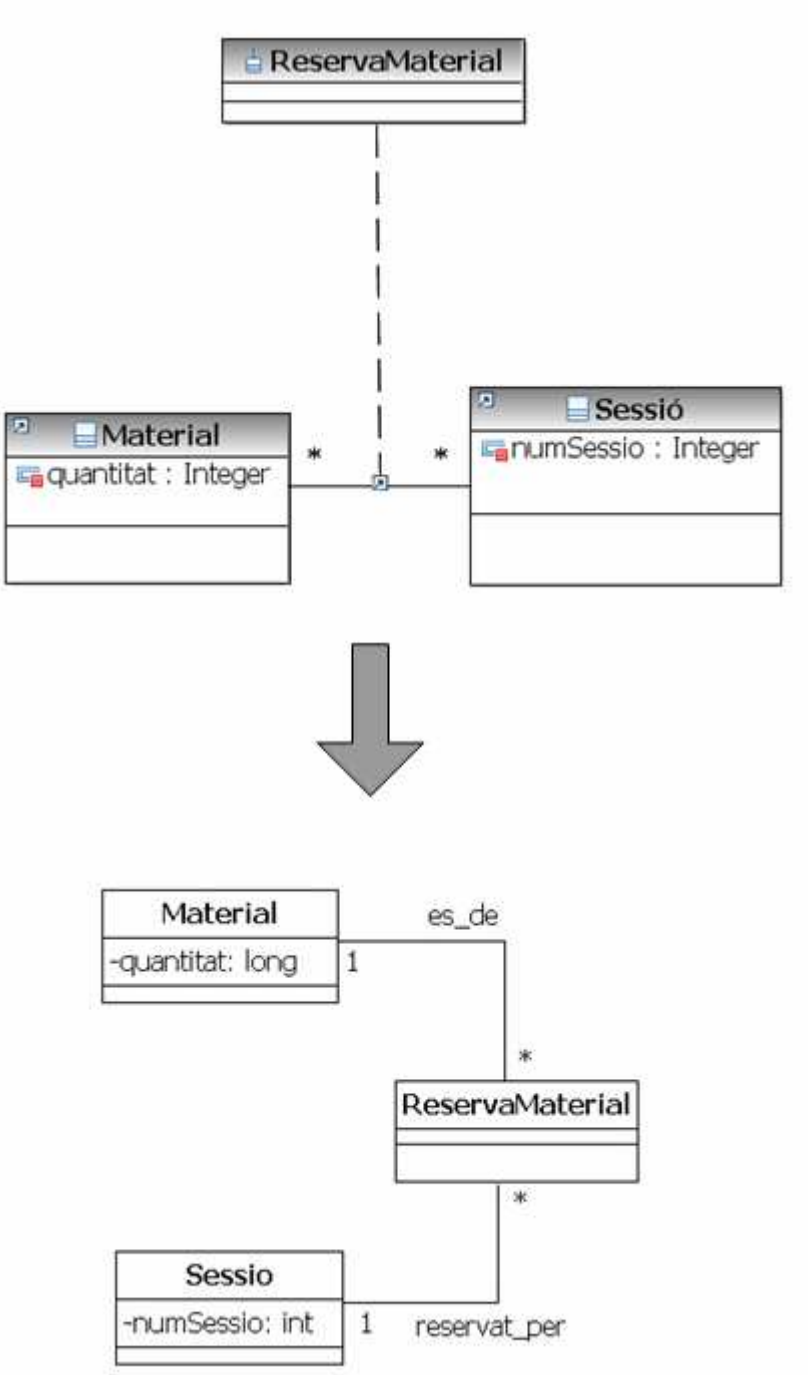
A partir del model conceptual obtingut en l'etapa d'especificació farem la normalització en l'etapa de disseny.

#### Normalització del diagrama de classes

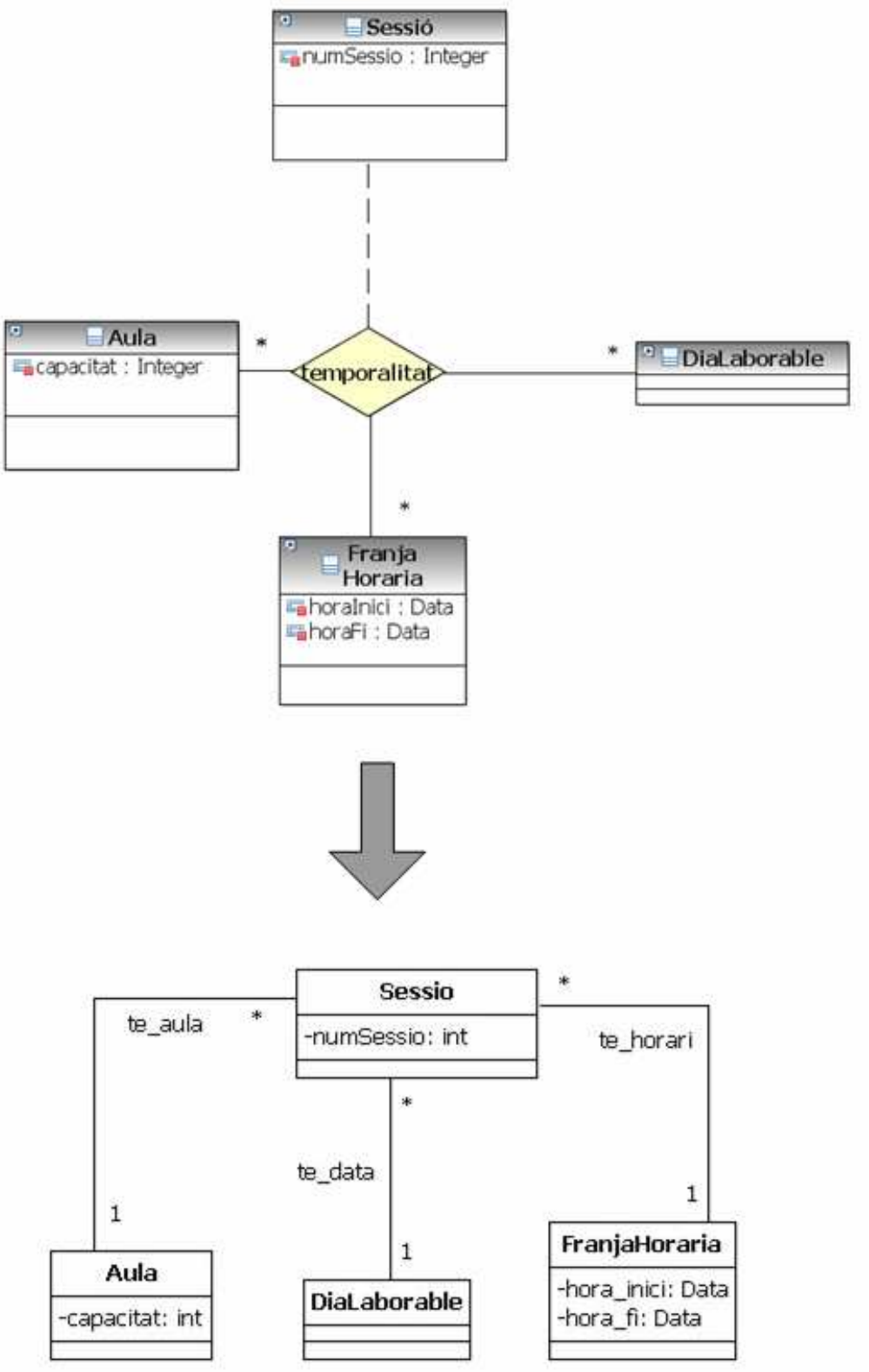
La tecnologia que emprem no permet implementar directament tots els conceptes que s'han utilitzat en l'especificació, com per exemple: associacions n-àries amb n>2, classes associatives, informació derivada o control de les restriccions d'integritat. És per això que cal una transformació prèvia, el procés de Normalització.

Normalitzar el diagrama de classes de l'especificació consisteix en transformar a binària totes les associacions que no ho siguin, així com eliminar les classes associatives. Com veurem a continuació, en eliminar les associacions n-àries i les classes associatives, apareixen noves restriccions d'integritat textual. Pels atributs derivats cal decidir si es materialitzen o es calculen quan es necessitin; en la decisió de materialitzar o calcular influeix el temps de càlcul la freqüència d'accés a aquest atribut i l'espai ocupat.

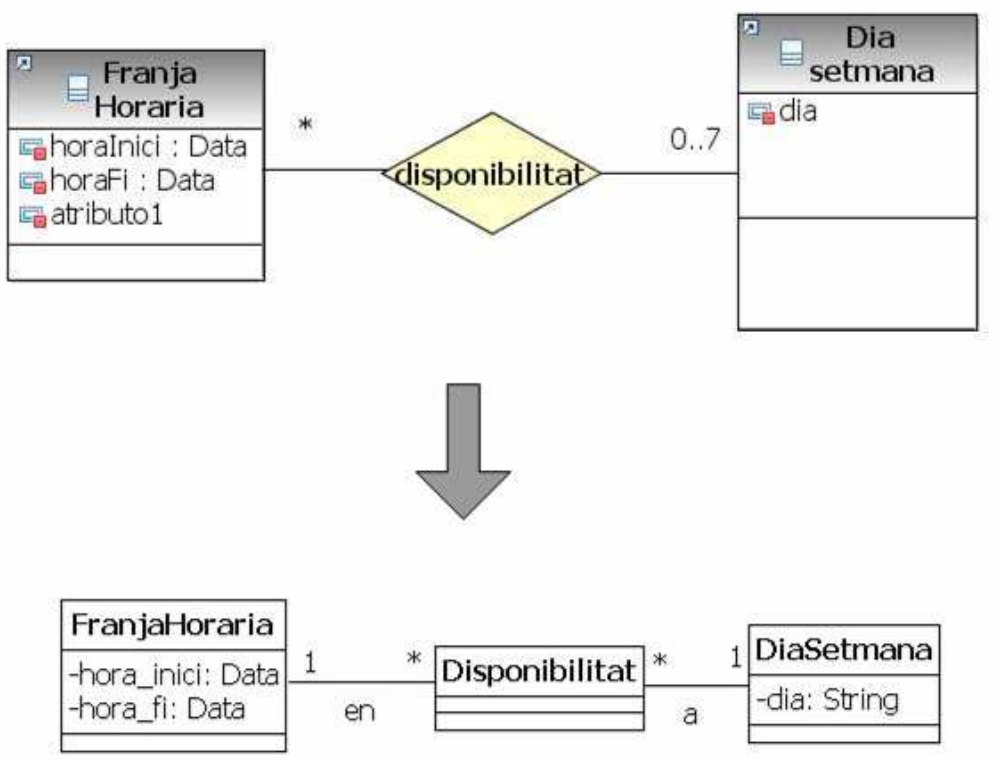
## Transformacions d'associacions n-àries i classes associatives



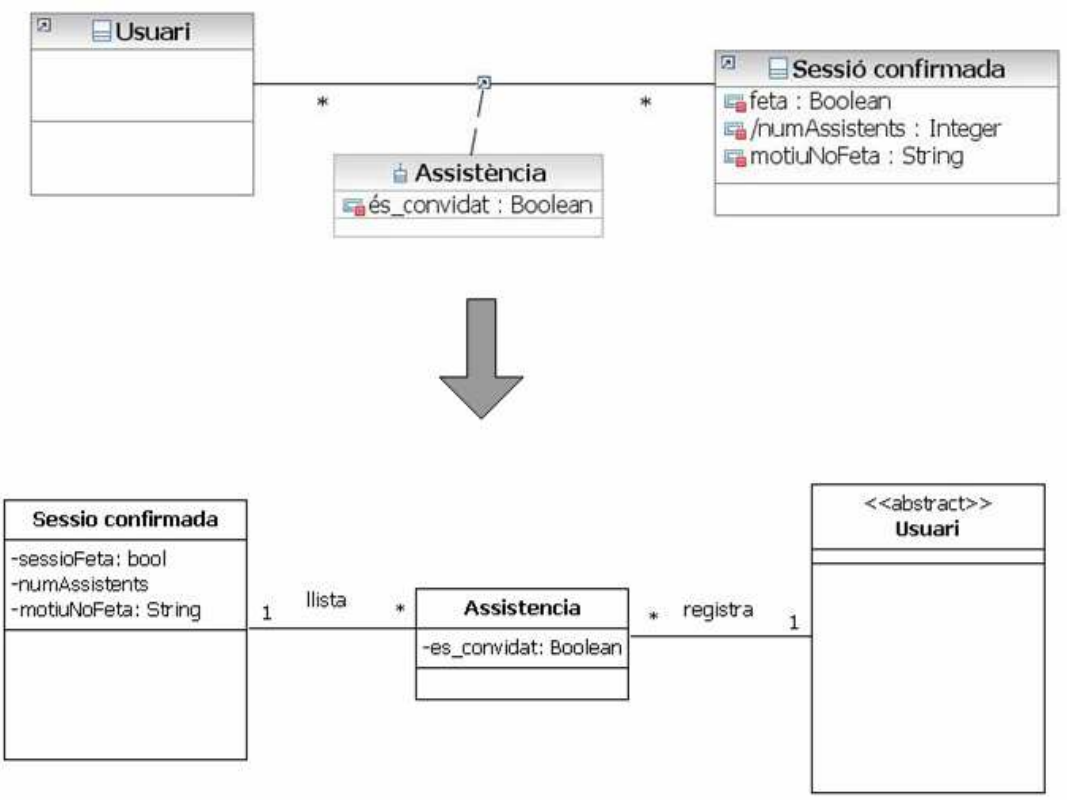

Restricció d'integritat addicional (RT10): No pot haver-hi dos objectes ReservaMaterial associats a la mateixa Sessió i Material.



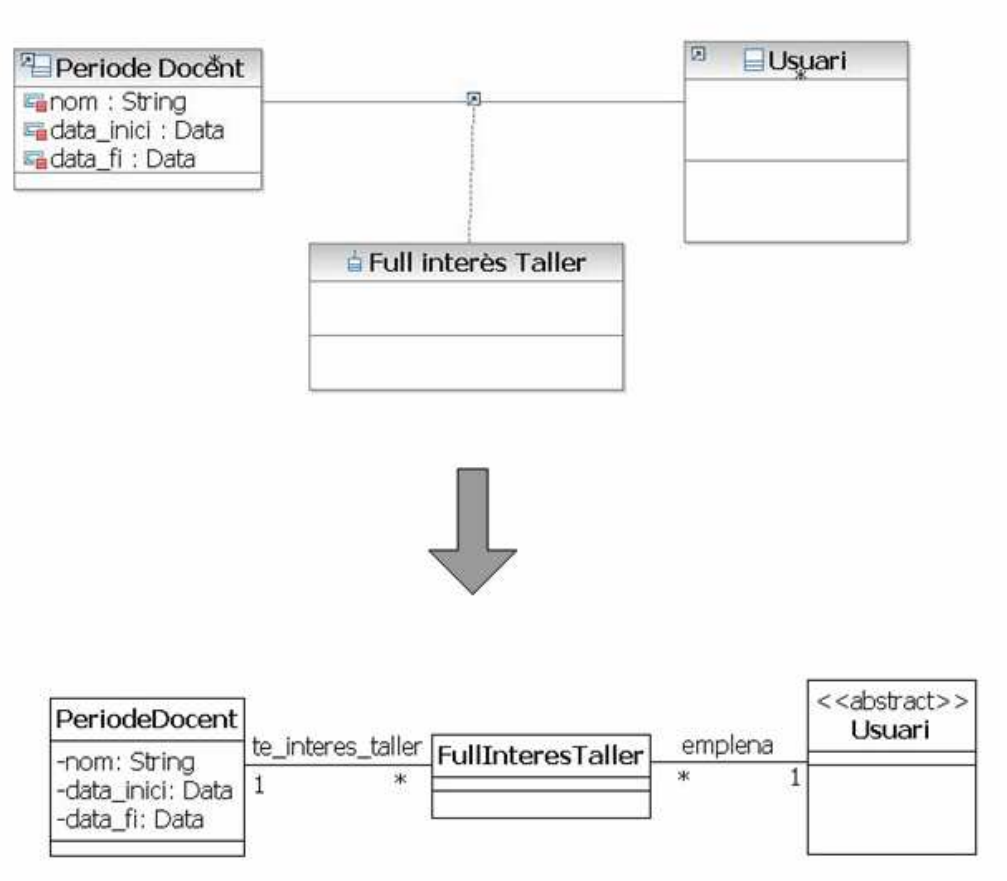

Restricció d'integritat addicional (RT11): No pot haver-hi dos objectes Sessio associats a la mateixa Aula, DiaLaborable i FranjaHoraria.



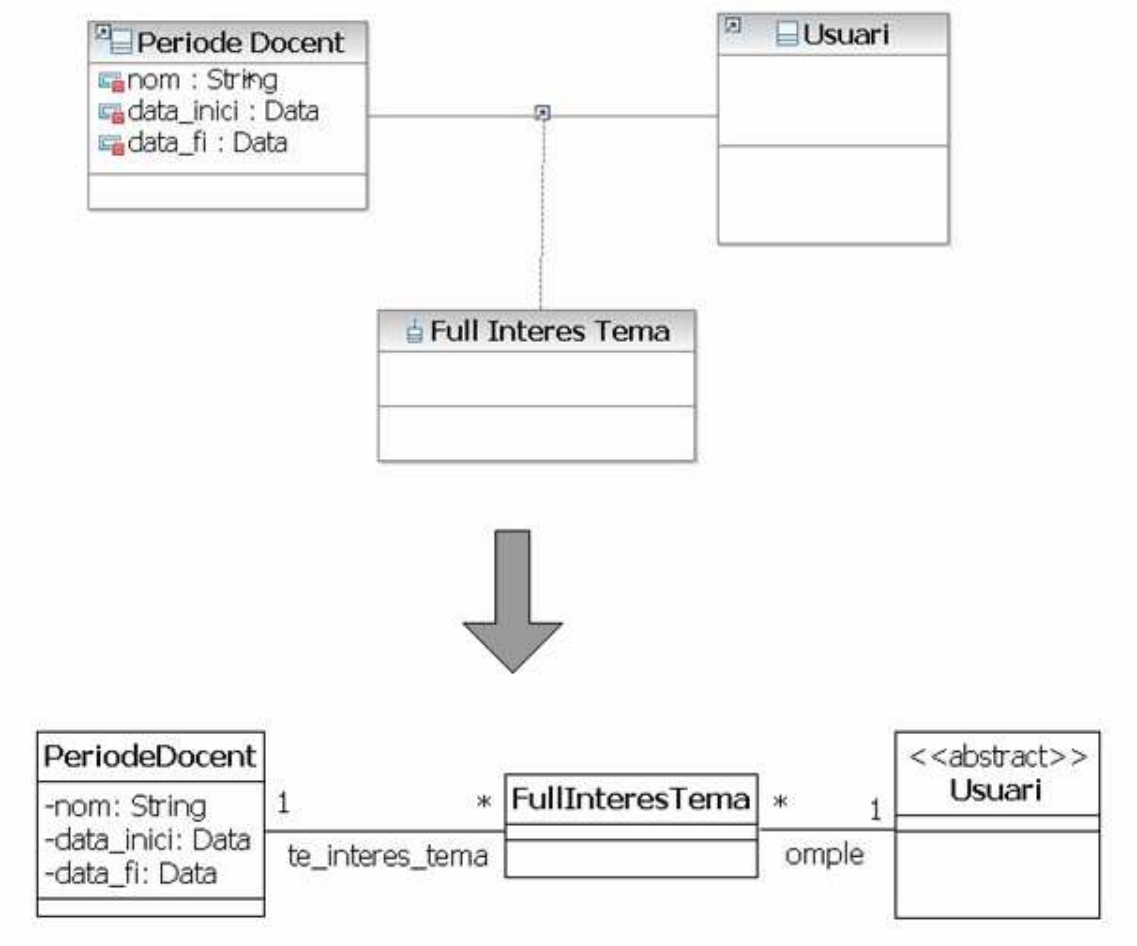

Restricció d'integritat addicional (RT12): No pot haver-hi dos objectes Disponibilitat associats a la mateixa FranjaHoraria i DiaSetmana.



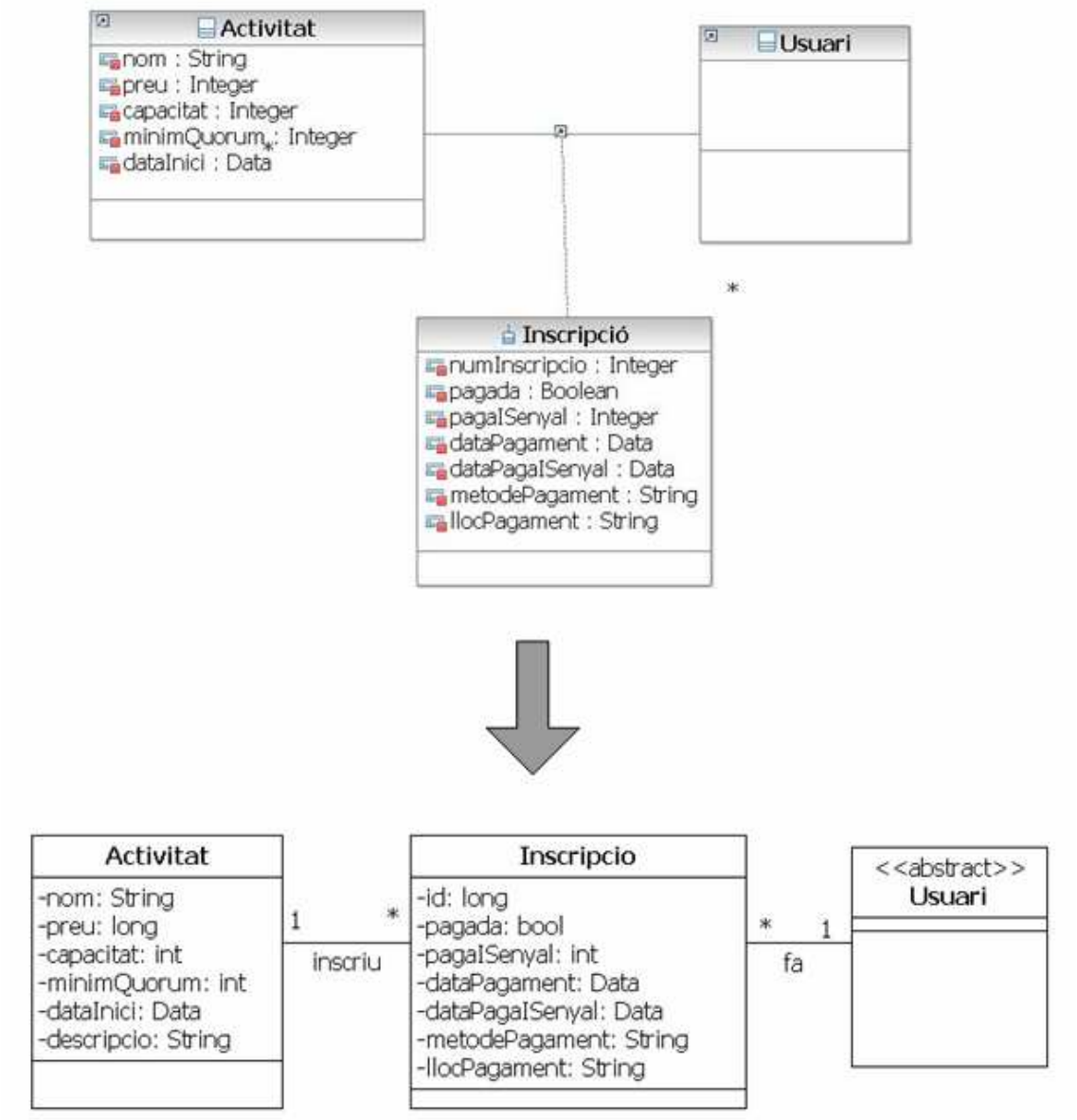

Restricció d'integritat addicional (RT13): No pot haver-hi dos objectes Assistència associats a la mateixa SessioConfirmada i Usuari.



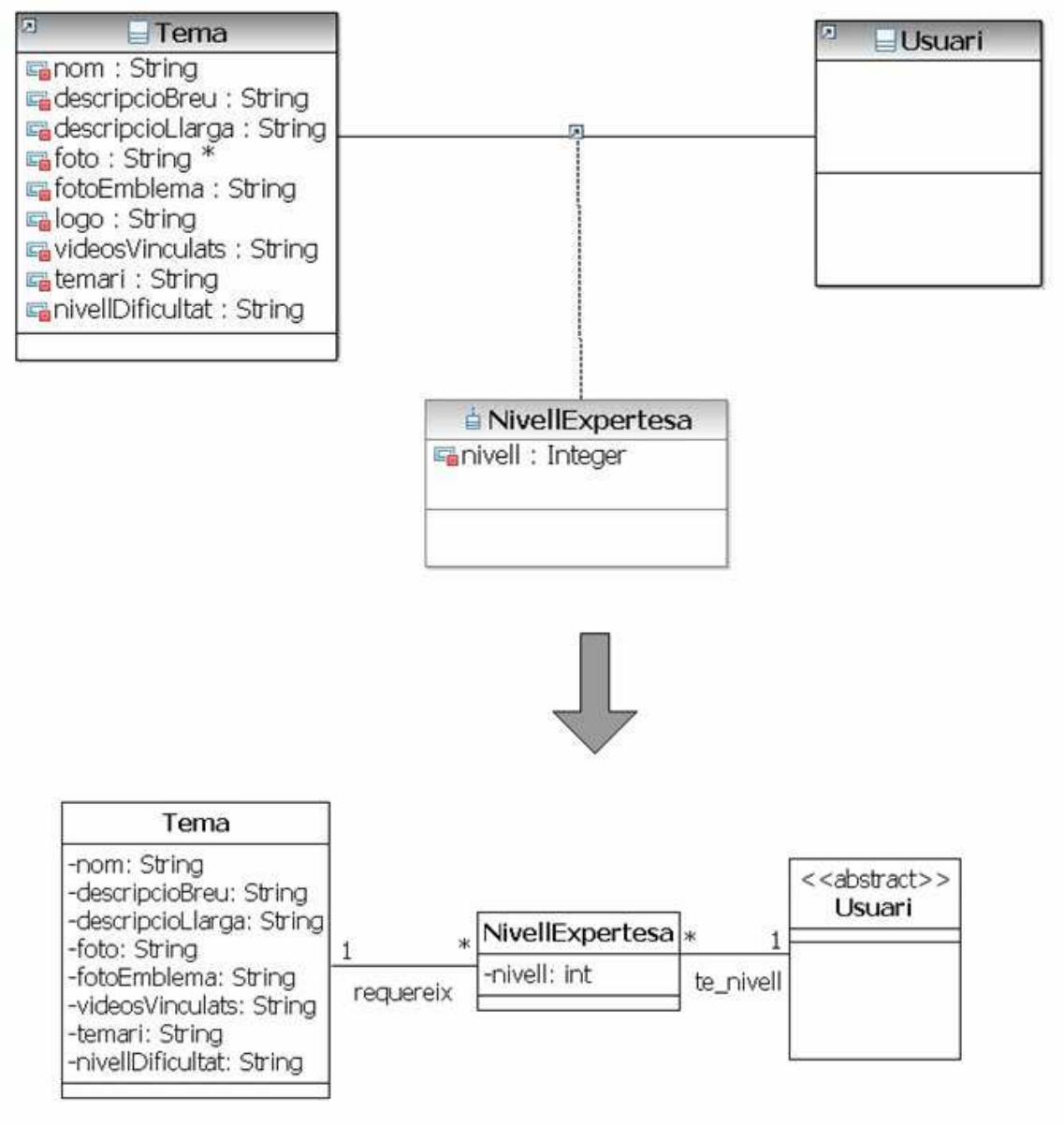

Restricció d'integritat addicional (RT14): No pot haver-hi dos objectes Full Interès Taller associats al mateix Període Docent i Usuari.




Restricció d'integritat addicional (RT15): No pot haver-hi dos objectes Full Interès Tema associats al mateix Període Docent i Usuari.




Restricció d'integritat addicional (RT16): No pot haver-hi dos objectes Inscripció associats al mateix Usuari i Activitat.

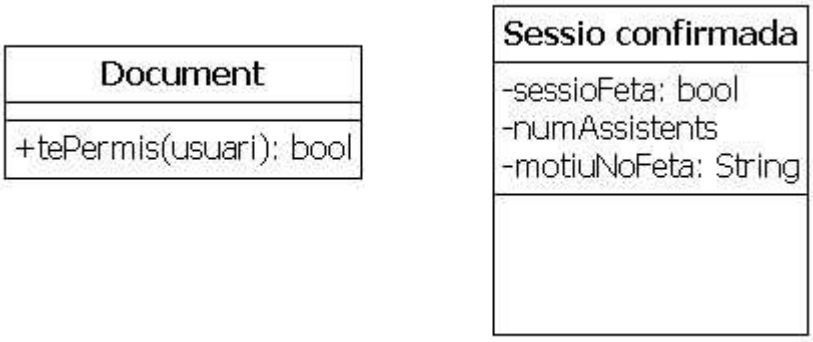



Restricció d'integritat addicional (RT17): No pot haver-hi dos objectes Nivell Expertesa associats al mateix Tema i Usuari.

### Atributos derivados

En el nostre model tenim un atribut derivat i una funció derivada. En el cas de l'atribut derivat s'ha decidit materialitzar-lo doncs ocupa poc espai i en el cas de la funció derivada, calcular-la ja que en principi no es farà massa sovint i per tant no sobrecarregarà el sistema.

Per tant, la classe Sessio Confirmada i la classe Document mostraran l'atribut i la funció respectivament en el diagrama de classes:




## Tractament de les restriccions d'integritat

Al dissenyar s'ha de garantir que el sistema resultant ha de satisfer les restriccions d'integritat que s'han definit en el model conceptual juntament amb les que han aparegut al realitzar el procés de normalització.

Donat que el diagrama de classes normalitzat no mostra aquestes restriccions d'integritat textuals, s'ha de decidir de quina forma seran controlades. Per això existeixen dues opcions: comprovar-les mitjançant la base de dades o realitzar la comprovació per programa. Si ho fem per base de dades és perquè sabem que el SGBD disposa de mecanismes per controlar-ho. En canvi, si es controla per programa s'ha d'afegir la restricció als contractes de les operacions per garantir el seu correcte funcionament.

A continuació es mostra una relació de totes les restriccions d'integritat i del mètode que s'utilitzarà per comprovar-les.

| **RESTRICCIÓ** | **MÈTODE** |
|---|---|
| **Claus externes** | **BD (Primary Key)** |
| RT1. Un alumne amb un nivell no pot inscriure's en un curs d'un nivell superior al seu. | Programa |
| RT2. Per tot taller, el nombre d'inscripcions ha de ser major o igual que el mínimquorum del taller. | Programa |
| RT3. Tots els períodes docents als que pertany un taller han de ser de la mateixa temporada. | Programa |
| RT4. Un formador assignat a un taller ha de ser expert en els temes que tracta el taller. | Programa |
| RT5. Un formador que imparteix una sessió d'un taller ha de ser expert en els temes que tracta el taller. | Programa |
| RT6. Tot formador assignat a un taller que te més formadors assignats ha d'estar en el conjunt d'afinitat de tots els formadors del taller. | Programa |
| RT7. Un grup només pot guanyar un premi d'un concurs en el que hi participa. | Programa |
| RT8. El dia de la setmana en que cau una data d'una sessió és el mateix dia de la setmana de l'espai-temps de la sessió | Programa |
| RT9. Un usuari no pot fer una inscripció a una activitat que tingui les sessions previstes. | Programa |
| RT10. No pot haver-hi dos objectes ReservaMaterial associats a la mateixa Sessió i Material. | BD (Primary Key) |
| RT11. No pot haver-hi dos objectes Nivell Expertesa associats al mateix Tema i Usuari. | BD (Primary Key) |
| RT12. No pot haver-hi dos objectes Inscripció associats al mateix Usuari i Activitat. | BD (Primary Key) |
| RT13. No pot haver-hi dos objectes Full Interès Tema associats al mateix Període Docent i Usuari. | BD (Primary Key) |
| RT14. No pot haver-hi dos objectes Full Interès Taller associats al mateix Període Docent i Usuari. | BD (Primary Key) |
| RT15. No pot haver-hi dos objectes Assistència associats a la mateixa SessioConfirmada i Usuari. | BD (Primary Key) |
| RT16. No pot haver-hi dos objectes Disponibilitat associats a la mateixa FranjaHoraria i DiaSetmana. | BD (Primary Key) |
| RT17. No pot haver-hi dos objectes Sessio associats a la mateixa Aula, DiaLaborable i FranjaHoraria. | BD (Primary Key) |

Les restriccions controlades per la base de dades es garanteixen mitjançant l'ús de PRIMARY KEY i de FOREIGN KEY en els atributs corresponents. En canvi, les restriccions controlades pel programa requereixen que es modifiquin els contractes de les operacions responsables de fer les comprovacions. Aquestes modificacions en els contractes les farem al següent apartat.

## Contractes de les operacions normalitzats

Els contractes de les operacions que s'han realitzat en la fase d'especificació no són vigents després de la normalització. Concretament es veuen afectats per dos canvis que s'han realitzat durant l'etapa de disseny: per una banda s'ha modificat les classes del model conceptual i s'han definit els atributs derivats i per l'altre banda, ara seran responsables de controlar algunes de les restriccions d'integritat textuals.

A continuació es mostra a mode d'exemple, alguns contractes que s'han vist afectats pels canvis (les modificacions es mostren en cursiva). Aquests canvis serien anàlogues per la resta de contractes afectats.

### Dóna d'alta inscripció activitat

**Nom:** AltaInscripcioActivitat(dadesInscripcio, usuari, activitat)
**Cas d'ús:** Dóna d'alta inscripció activitat
**Responsabilitats:** Dóna d'alta una inscripció d'un usuari a una activitat
**Excepcions:**

- No existeix un **Usuari** amb dni=usuari
- No existeix una **Activitat** amb nom=activitat
- *L'Usuari amb dni=usuari té nivell inferior al de l'Activitat amb nom=activitat (RT1)*
- *L'Activitat amb nom=activitat té les sessions previstes (RT9)*

**Precondicions:**

- Existeix un **Usuari** amb dni=usuari
- Existeix una **Activitat** amb nom=activitat
- *L'Usuari amb dni=usuari té nivell superior o igual al de l'Activitat amb nom=activitat (RT1)*

- *L'Activitat amb nom=activitat té les sessions confirmades (RT9)*

**Postcondicions:**

1. Es crea una nova instància de la classe ~~*associativa*~~ **Inscripció** i s'inicialitza amb els paràmetres de l'operació
2. *Es crea una instància de l'associació 'inscriu' entre la Inscripció i l'Activitat*
3. *Es crea una instància de l'associació 'fa' entre la Inscripció i l'Usuari*

**Sortida:** -

**Ompla full interès Taller**

**Nom:** AltaFullInteresTaller(usuari,periodedocent, {taller})

**Cas d'ús:** Ompla full interès Taller

**Responsabilitats:** Dóna d'alta un full d'interès d'un usuari en un període docent amb els tallers que li interessen

**Excepcions:**

- No existeix un **Usuari** amb dni=usuari
- No existeix un **Periode Docent** amb nom=periodedocent
- Per a cada Taller a {taller} no existeix el **Taller** amb nom=taller
- Existeix un **Full Interes Taller** de l'Usuari amb dni=usuari i el Periode Docent amb nom=periodedocent

**Precondicions:**

- Existeix un **Usuari** U amb dni=usuari
- Existeix un **Periode Docent** PD amb nom=periodedocent
- Per a cada Taller a {taller} existeix el **Taller** amb nom=taller
- No existeix un **Full Interes Taller** de l'Usuari amb dni=usuari i el Periode Docent amb nom=periodedocent

**Postcondicions:**

1. Dóna d'alta una nova instància FIT de la classe ~~*associativa*~~ **Full Interes Taller** entre l'Usuari U i el Periode Docent PD
2. *Es crea una instància de l'associació 'té_interès_taller' entre el Full Interès Taller FIT i el Periode Docent PD*
3. *Es crea una instància de l'associació 'emplena' entre el Full Interès Taller FIT i el Usuari U*
4. Per a cada Taller de {taller}
    c. Dóna d'alta una instància de l'associació '**interessa'** entre el Full Interes Taller FIT i el Taller amb id=taller

**Sortida:** -

**Alta llista assistència taller**

**Nom:** AltaLlistaAssistenciaTaller(taller,numsessio,{alumne, convidat})

**Cas d'ús:** Alta llista assistència taller

**Responsabilitats:** Dóna d'alta la llista d'assistència d'un conjunt d'alumnes a una sessió confirmada d'un taller.

**Excepcions:**

- No existeix un **Taller** amb nom=taller
- No existeix una **Sessio Confirmada** amb numSessio=numsessio
- Per algun **Usuari** a {alumne}, no existeix l'Usuari amb dni=alumne
- *El Formador que inicia el cas d'ús no és expert en els temes que tracta el taller*

**Precondicions:**

- Existeix un **Taller** amb nom=taller
- Existeix una **Sessio Confirmada** amb numSessio=numsessio
- Per a cada **Usuari** a {alumne}, no existeix una instància d'Usuari amb dni=alumne
- *El Formador que inicia el cas d'ús és expert en els temes que tracta el taller*

**Postcondicions:**

1. Per a cada Usuari a {alumne, convidat}
    a. Dóna d'alta una instància de la classe ~~*associativa*~~ **Assistència A** entre l'Usuari amb dni=alumne i el Taller amb nom=Taller. A actualitza el seu atribut es_convidat=convidat.
    b. *Es crea una instància de l'associació 'registra' entre l'Assistència A i l'Usuari*
    c. *Es crea una instància de l'associació 'llista' entre l'Assistència A i la Sessió Confirmada amb numSessio=numsessio*
2. Un cop finalitzada l'entrada d'assistencies, es recalcula l'atribut ~~*derivat*~~ **numAssistents** de la Sessio Confirmada amb numSessio=numsessio amb el nombre d'instàncies Assistència creades.

**Sortida:** -

**Dóna de baixa Tema**

**Nom:** BaixaTema(tema)

**Cas d'ús:** Dóna de baixa Tema

**Responsabilitats:** Dóna de baixa un tema.

**Excepcions:**

- No existeix un **Tema** amb nom=tema

**Precondicions:**

- Existeix un **Tema** amb nom=tema

**Postcondicions:**

1. Per cada instància de la associació 'pertany':
    a. Dóna de baixa la **Especialitat** E accessible a través de la instància de l'associació 'pertany'.
    b. Dóna de baixa la instància de l'associació 'pertany'.
2. Per cada instància de la associació 'es_composa_de':
    a. Dóna de baixa la **SessioTemari** ST accessible a través de la instància de l'associació 'es_composa_de'.
    b. Dóna de baixa la instància de l'associació 'es_composa_de'.
3. Per cada instància de la associació 'té':
    a. Dóna de baixa el **Document** D accessible a través de la instància de l'associació 'té'.
    b. Dóna de baixa la instància de l'associació 'té'.
4. Per cada instància de classe ~~*associativa*~~ **Nivell Expertesa** NE entre el Tema T i instàncies de la classe Usuari:
    a. *Esborra la instància de l'associació 'requereix' entre el Nivell Expertesa NE i el Tema T*
    b. *Esborra la instància de l'associació 'té_nivell' entre el Nivell Expertesa NE i l'Usuari*
    c. Dóna de baixa la instància NE de la classe Nivell Expertesa
5. Per cada instància de la associació 'mostra_interes_en':
    a. Dóna de baixa la instància de l'associació 'mostra_interes_en' .
6. Per cada instància de la associació 'tracta', dóna de baixa la instància de l'associació 'tracta'' entre la instància del Tema T i la instància de la classe Activitat.

**Sortida:** -

**Planificar sessions Taller**

**Nom:** PlanificarSessionsTaller(taller,aula)
**Cas d'ús:** Planificar sessions Taller
**Responsabilitats:** Planifica les sessions d'una taller en una aula tenint en compte la seva data d'inici, els dies que es fa el taller i la seva periodicitat, el nombre de sessions del taller i el calendari de dies festius.
**Excepcions:**

- No existeix un **Taller** amb nom=taller
- No existeix una **Aula** amb nom=aula

**Precondicions:**

- Existeix un **Taller** T amb nom=taller
- Existeix una **Aula** A amb nom=aula

**Postcondicions:**

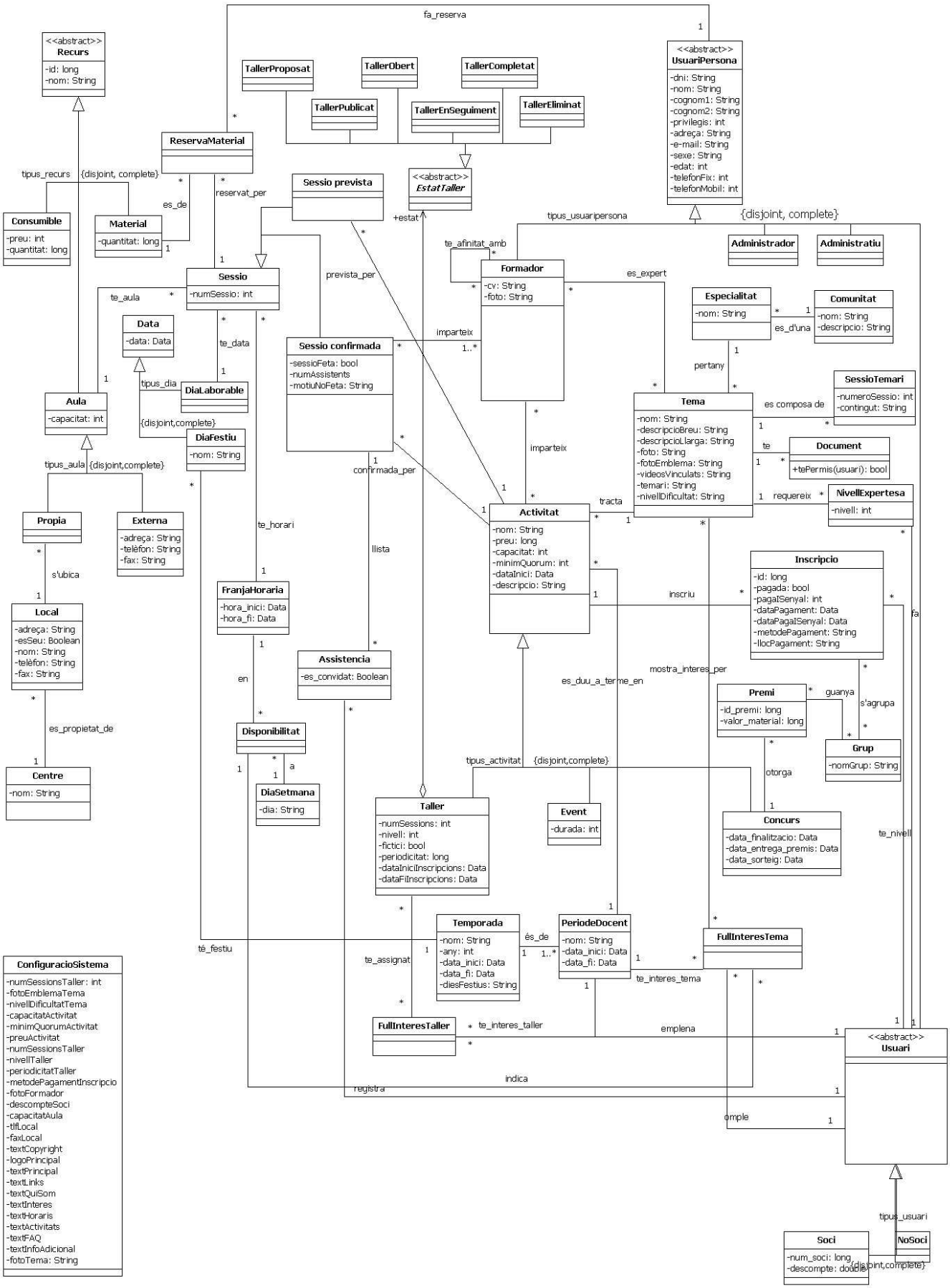
1. Crea una instància SP de la classe **Sessió Prevista** entre l'Aula A i la Data amb data igual a T.dataInici.
2. *Crea una instància de l'associació 'prevista_per' entre la Sessió Prevista SP i el Taller T.*
3. Mentre T.numSessions sigui major que 1
    a. Crea una instància SP2 de la classe **Sessió Prevista** entre l'Aula A i la Data amb data igual a T.dataInici + T.periodicitat sempre que no sigui festiu.
    b. *Crea una instància de l'associació 'prevista_per' entre la Sessió Prevista SP2 i el Taller T.*

**Sortida:** -

Els contractes **Confirma Taller** i **Consulta llistat fulls interes Tema** vistos en l'apartat romanen igual després de la formalització.

## Diagrama de classes normalitzat

Tot seguit es mostra el diagrama de classes resultant del procés de normalització descrit en els apartats anteriors.


fa_reserva
<<abstract>>
Recurs
-id: long
-nom: String
TallerProposat
TallerObert
TallerCompletat
TallerPublicat
TallerEnSeguiment
TallerEliminat
<<abstract>>
UsuariPersona
-dni: String
-nom: String
-cognom1: String
-cognom2: String
-privilegis: int
-adreça: String
-e-mail: String
-sexe: String
-edat: int
-telefonFix: int
-telefonMobil: int
ReservaMaterial
tipus_recurs
{disjoint, complete}
reservat_per
es_de
Consumible
-preu: int
-quantitat: long
Material
-quantitat: long
Sessio prevista
<<abstract>>
EstatTaller
+estat
tipus_usuaripersona
{disjoint, complete}
Administrador
Administratiu
te_afinitat_amb
Formador
-cv: String
-foto: String
es_expert
Especialitat
-nom: String
es_d'una
Comunitat
-nom: String
-descripcio: String
Sessio
-numSessio: int
te_aula
prevista_per
Data
-data: Data
te_data
imparteix
1..*
Sessio confirmada
-sessioFeta: bool
-numAssistents
-motiuNoFeta: String
pertany
SessioTemari
-numeroSessio: int
-contingut: String
Tema
-nom: String
-descripcioBreu: String
-descripcioLlarga: String
-foto: String
-fotoEmblema: String
-videosVinculats: String
-temari: String
-nivellDificultat: String
es composa de
te
Document
+tePermis(usuari): bool
requereix
NivellExpertesa
-nivell: int
tracta
Aula
-capacitat: int
tipus_dia
DiaLaborable
{disjoint,complete}
DiaFestiu
-nom: String
tipus_aula
{disjoint,complete}
confirmada_per
imparteix
Activitat
-nom: String
-preu: long
-capacitat: int
-minimQuorum: int
-dataInici: Data
-descripcio: String
Inscripcio
-id: long
-pagada: bool
-pagatSenyal: int
-dataPagament: Data
-dataPagatSenyal: Data
-metodePagament: String
-llocPagament: String
inscriu
Propia
Externa
-adreça: String
-telèfon: String
-fax: String
te_horari
llista
s'ubica
Local
-adreça: String
-esSeu: Boolean
-nom: String
-telèfon: String
-fax: String
FranjaHoraria
-hora_inici: Data
-hora_fi: Data
Assistencia
-es_convidat: Boolean
en
mostra_interes_per
es_duu_a_terme_en
Premi
-id_premi: long
-valor_material: long
guanya
s'agrupa
Grup
-nomGrup: String
es_propietat_de
Centre
-nom: String
Disponibilitat
a
DiaSetmana
-dia: String
tipus_activitat
{disjoint,complete}
otorga
Taller
-numSessions: int
-nivell: int
-fictici: bool
-periodicitat: long
-dataIniciInscripcions: Data
-dataFiInscripcions: Data
Event
-durada: int
Concurs
-data_finalitzacio: Data
-data_entrega_premis: Data
-data_sorteig: Data
te_nivell
Temporada
-nom: String
-any: int
-data_inici: Data
-data_fi: Data
-diesFestius: String
és_de
PeriodeDocent
-nom: String
-data_inici: Data
-data_fi: Data
FullInteresTema
te_interes_tema
té_festiu
te_assignat
FullInteresTaller
te_interes_taller
emplena
<<abstract>>
Usuari
indica
registra
omple
ConfiguracioSistema
-numSessionsTaller: int
-fotoEmblemaTema
-nivellDificultatTema
-capacitatActivitat
-minimQuorumActivitat
-preuActivitat
-numSessionsTaller
-nivellTaller
-periodicitatTaller
-metodePagamentInscripcio
-fotoFormador
-descompteSoci
-capacitatAula
-tlfLocal
-faxLocal
-textCopyright
-logoPrincipal
-textPrincipal
-textLinks
-textQuiSom
-textInteres
-textHoraris
-textActivitats
-textFAQ
-textInfoAdicional
-fotoTema: String
tipus_usuari
Soci
-num_soci: long
-descompte: double
NoSoci
{disjoint,complete}

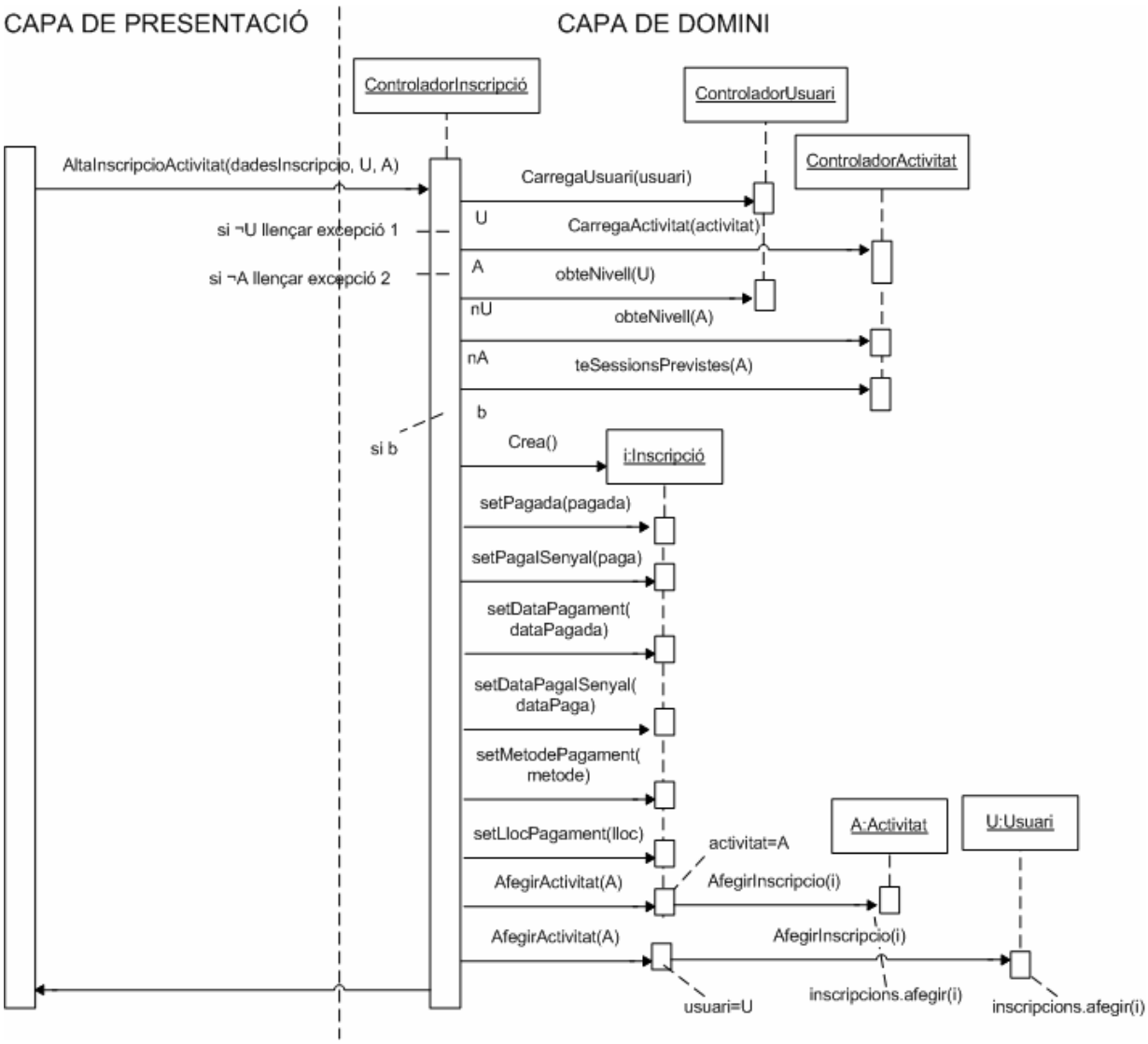
### 3.2.3 Diagrames de seqüència de les operacions

En aquest apartat es mostren els diagrames de seqüència corresponents a els casos d'ús vistos anteriorment, on es pot observar la interacció entre les capes del sistema.

En alguns dels diagrames que veurem a continuació, la capa de presentació ha d'accedir inicialment als controladors corresponents per tal d'obtenir els valors possibles de les dades que pot seleccionar l'usuari. En general s'ha decidit que aquestes trucades no quedaran reflectides visualment en els diagrames de seqüència, degut a que s'ha preferit explicar-les textualment en un comentari.

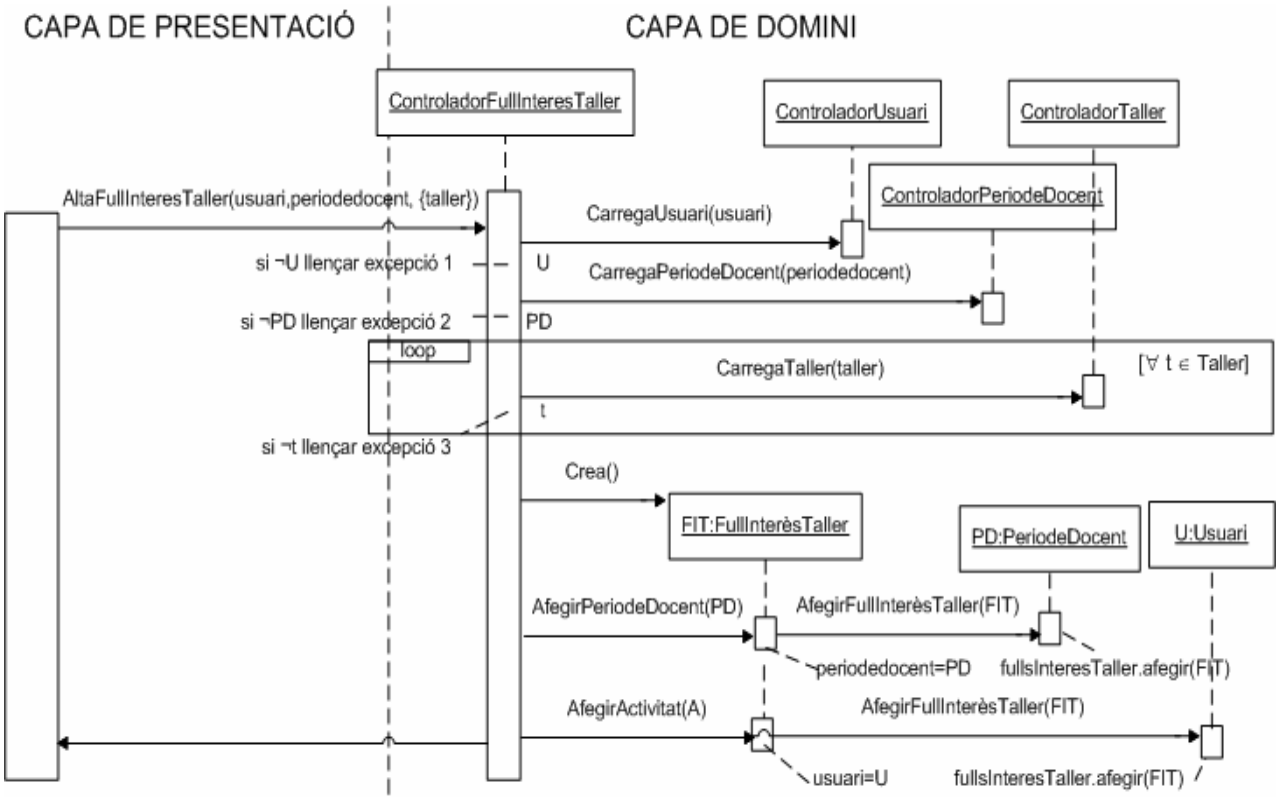
L'accés a la informació de la base de dades es realitza a través dels controladors, que s'encarreguen de tractar els esdeveniments i executar les accions corresponents.

**Dóna d'alta inscripció Activitat**

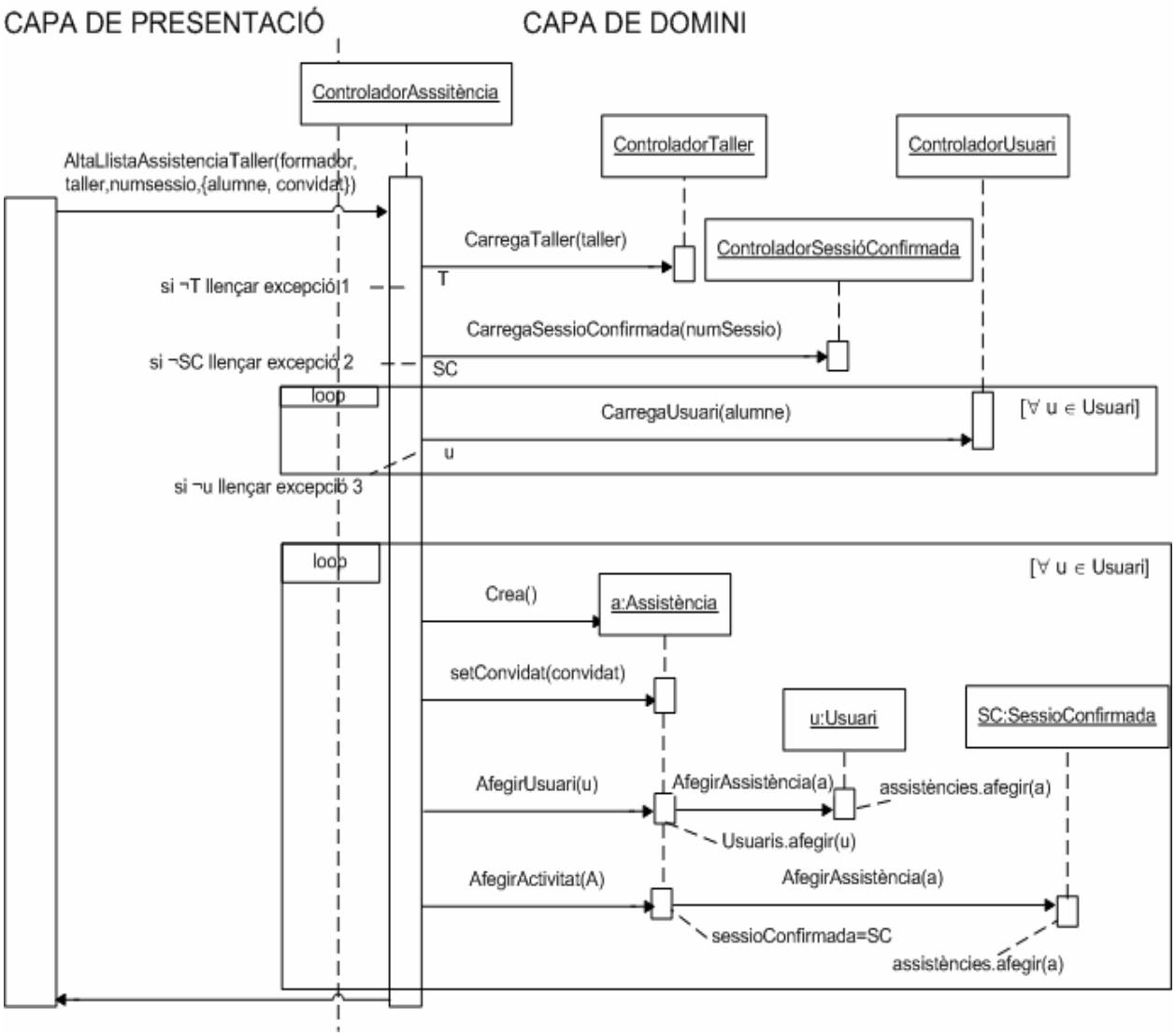



dadesInscripcio=(pagada, pagaISenyal, dataPagament, dataPagaISenyal, metodePagament, llocPagament)

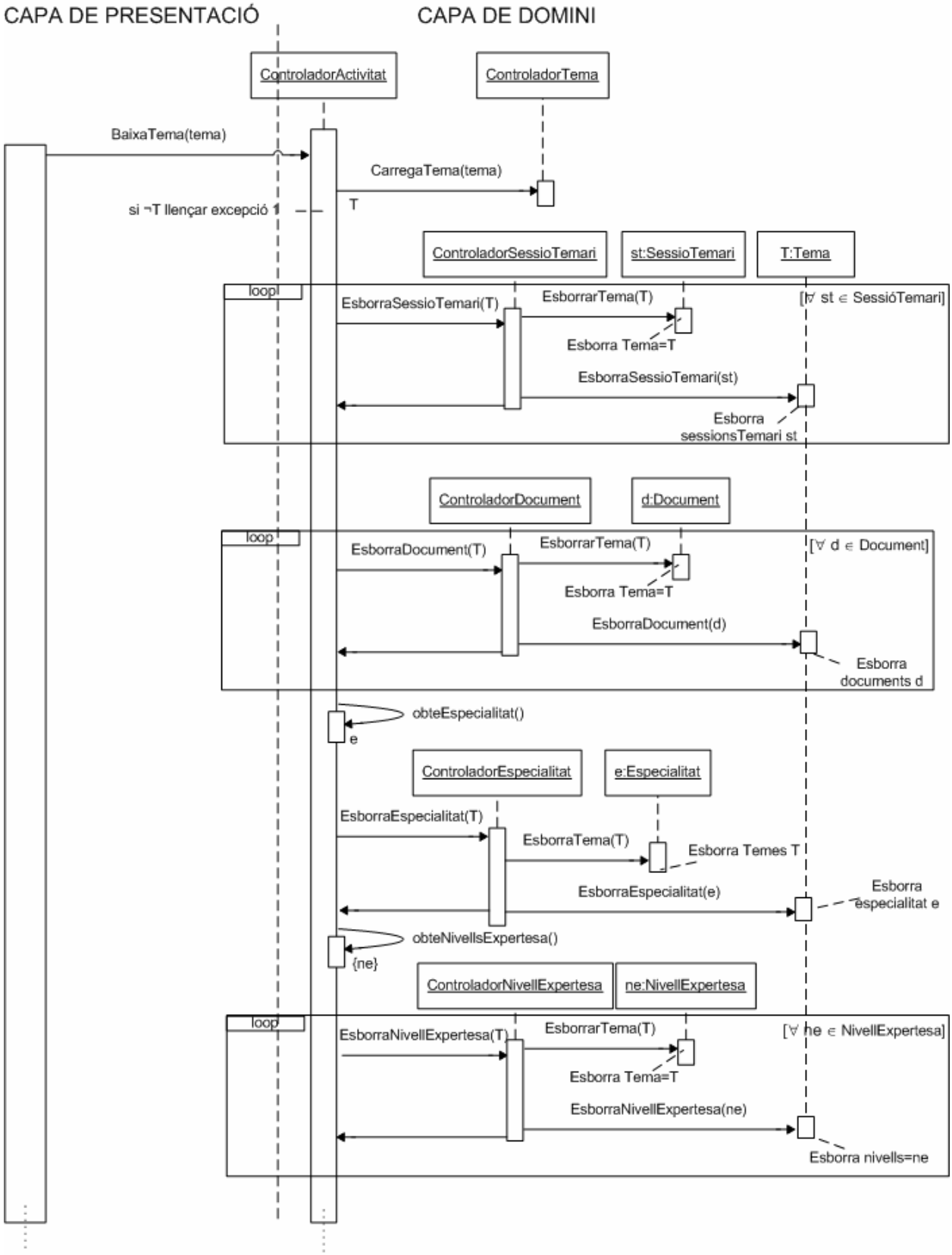
**Ompla full interès Taller**

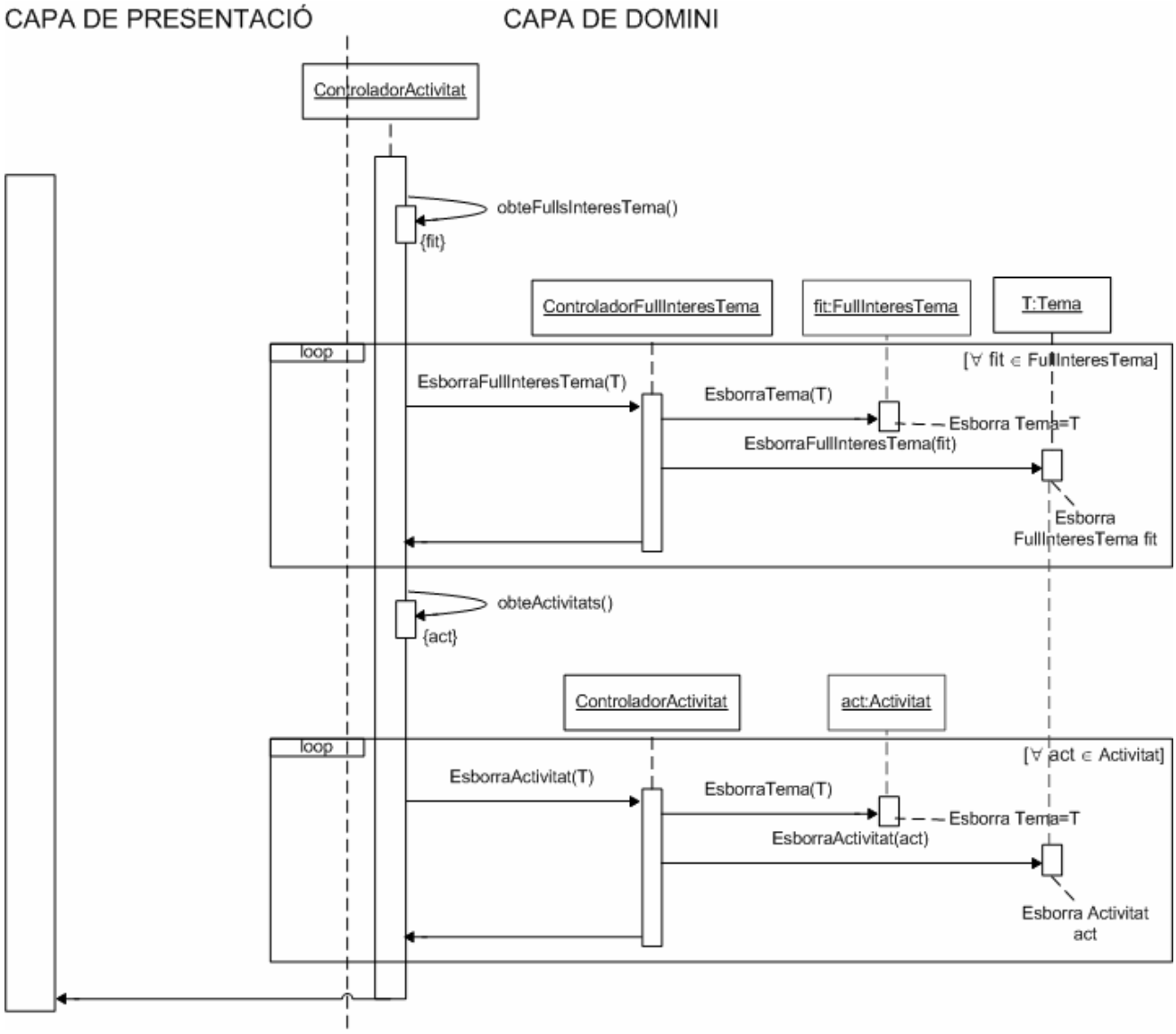


**Alta llista assistència taller**

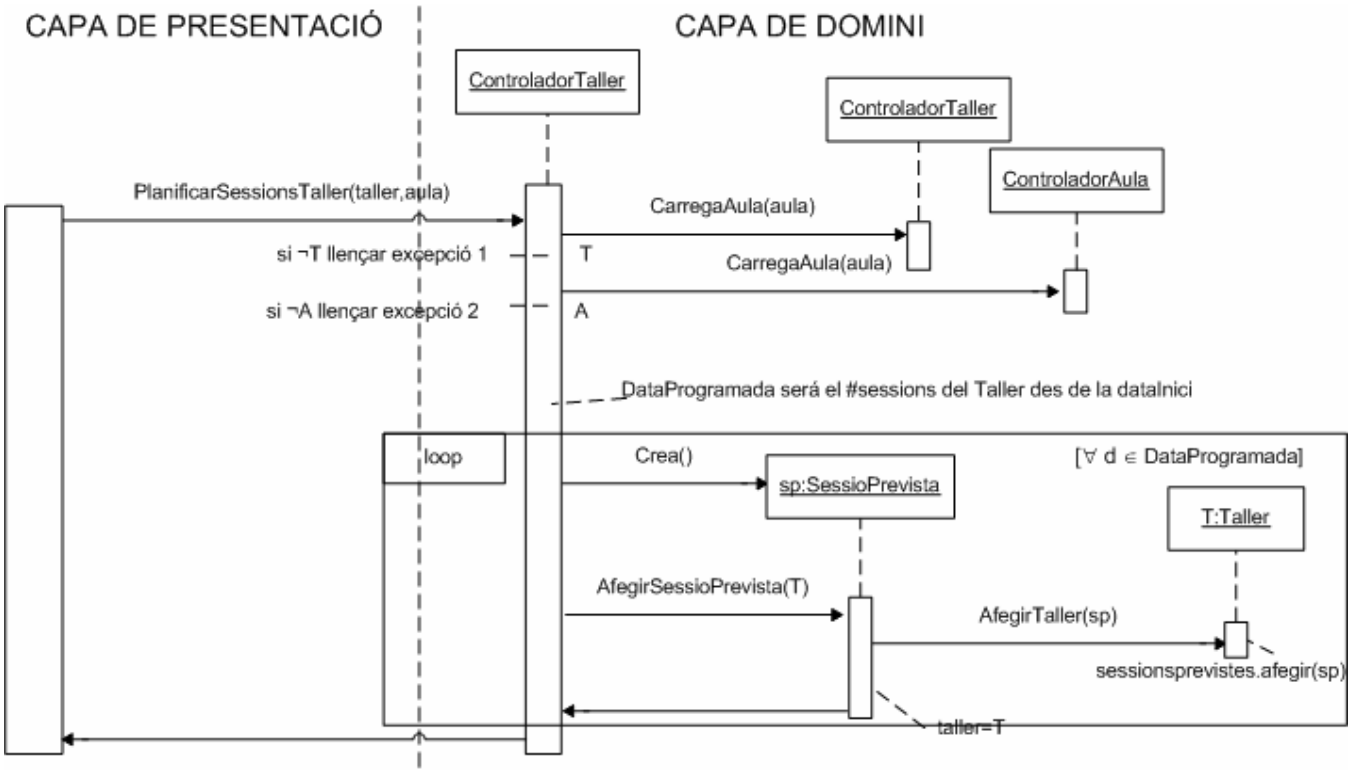


**Dóna baixa Tema**

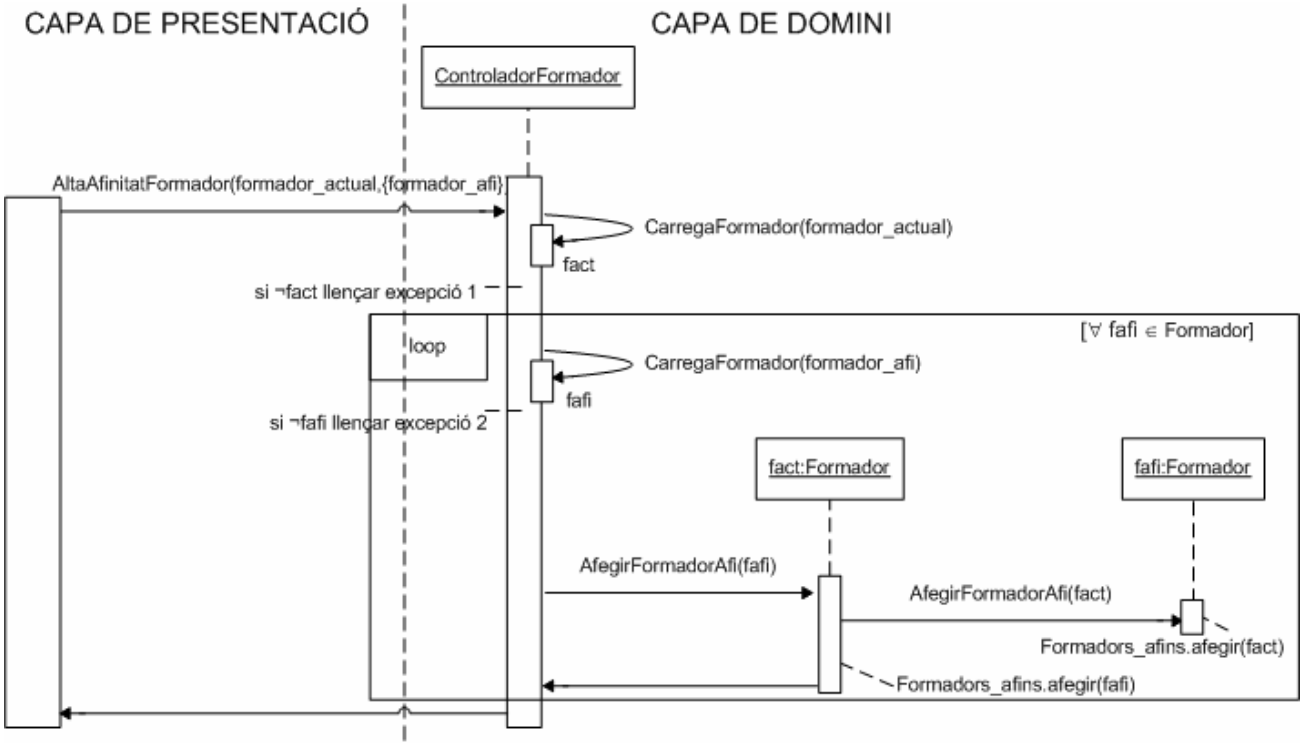



CAPA DE PRESENTACIÓ
CAPA DE DOMINI
ControladorActivitat
obteFullsInteresTema()
{fit}
ControladorFullInteresTema
fit:FullInteresTema
T:Tema
loop
[∀ fit ∈ FullInteresTema]
EsborraFullInteresTema(T)
EsborraTema(T)
Esborra Tema=T
EsborraFullInteresTema(fit)
Esborra
FullInteresTema fit
obteActivitats()
{act}
ControladorActivitat
act:Activitat
loop
[∀ act ∈ Activitat]
EsborraActivitat(T)
EsborraTema(T)
Esborra Tema=T
EsborraActivitat(act)
Esborra Activitat
act

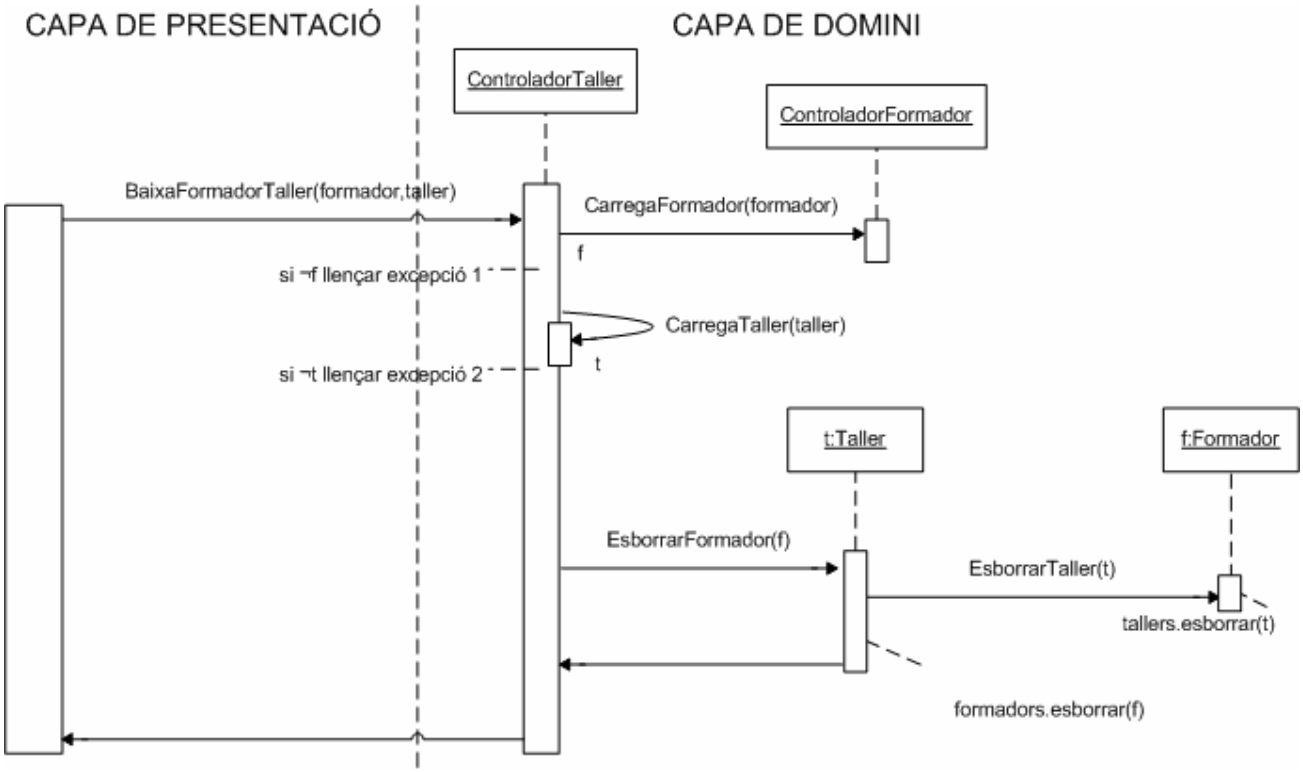
**Planificar sessions Taller**



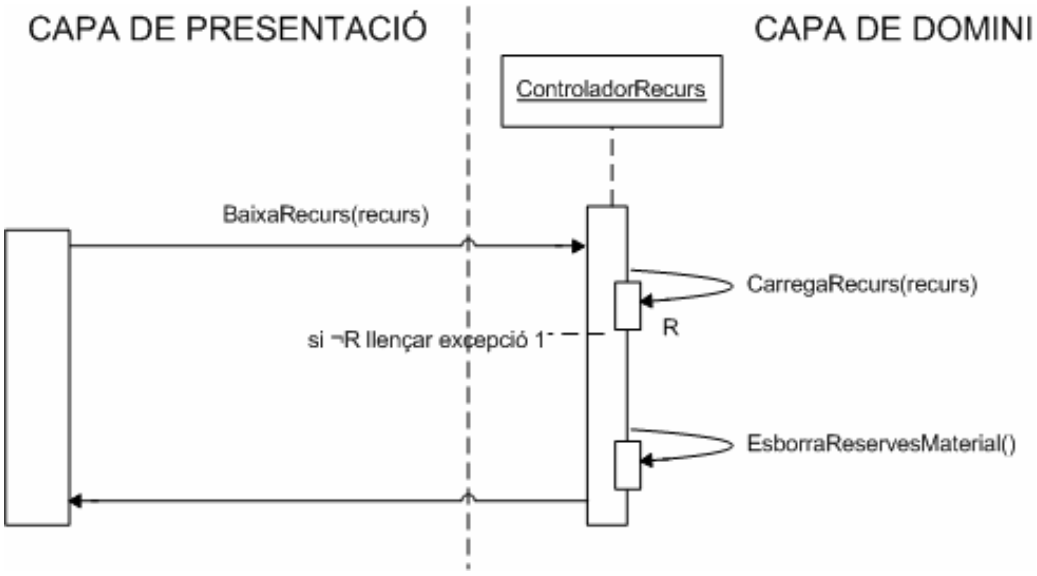
**Dóna d’alta afinitat formador**

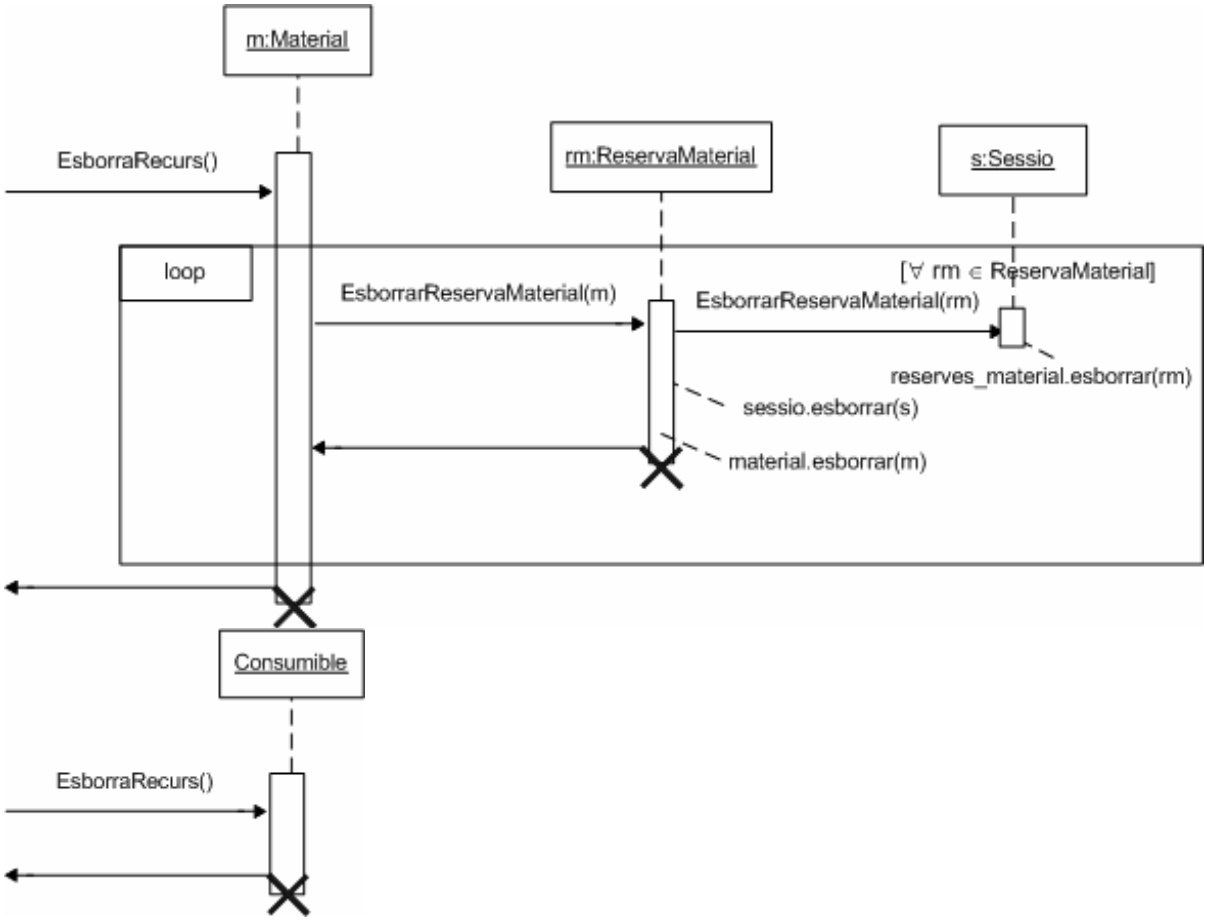


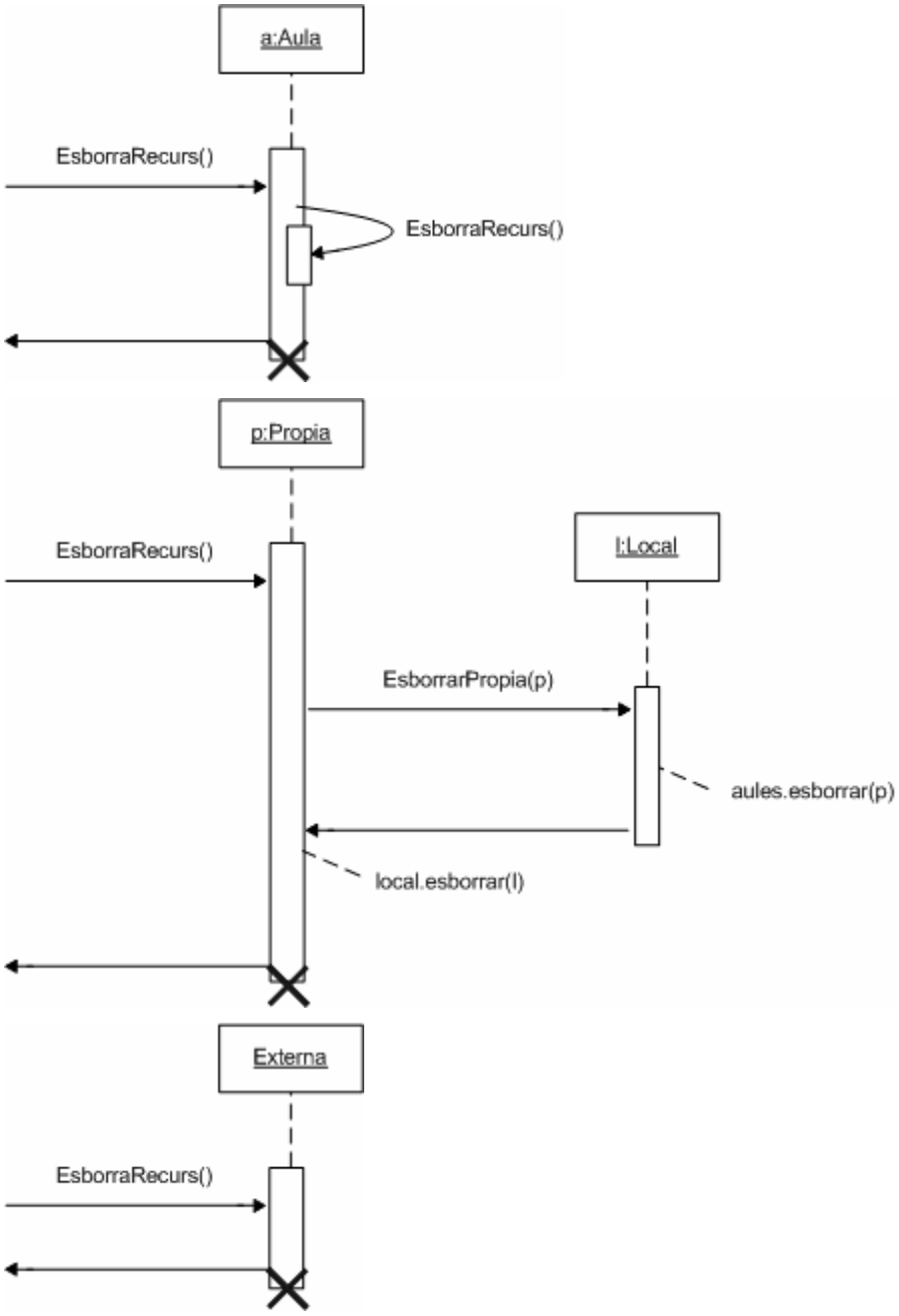
**Dóna de baixa formador a taller**



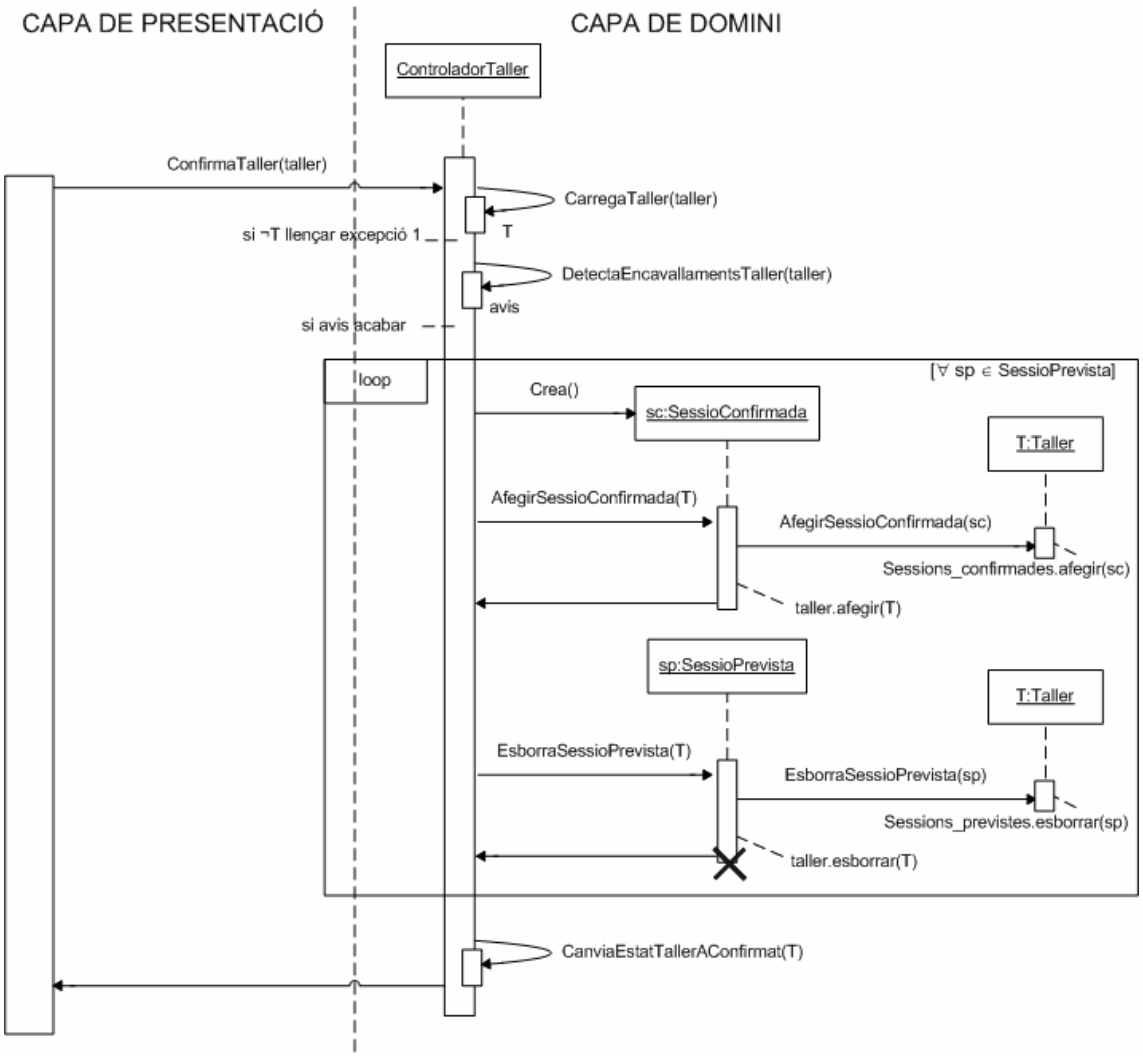
**Dóna de baixa recurs**

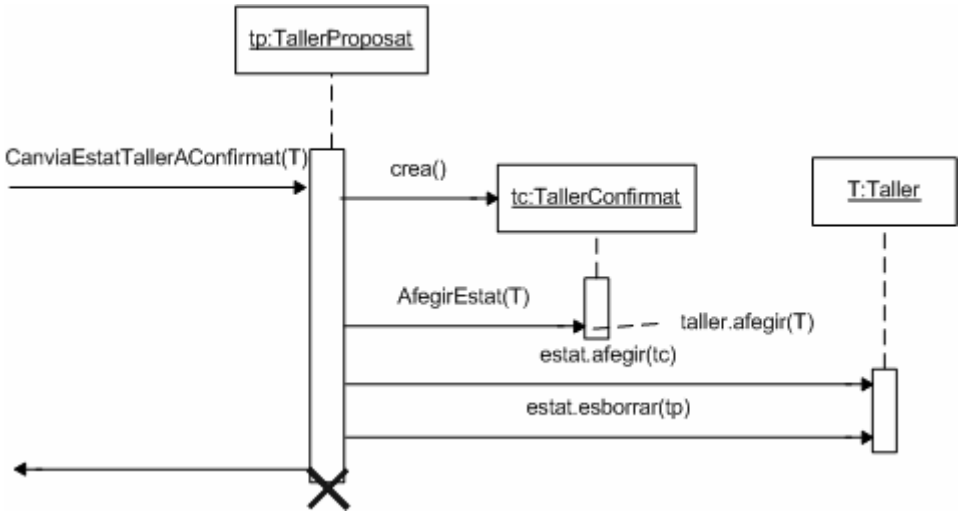


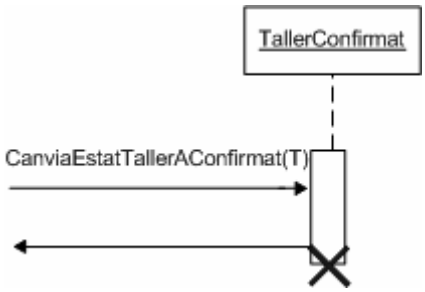



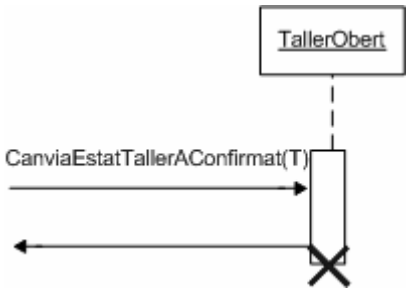

a:Aula
EsborraRecurs()
EsborraRecurs()
p:Propia
EsborraRecurs()
l:Local
EsborrarPropia(p)
aules.esborrar(p)
local.esborrar(l)
Externa
EsborraRecurs()

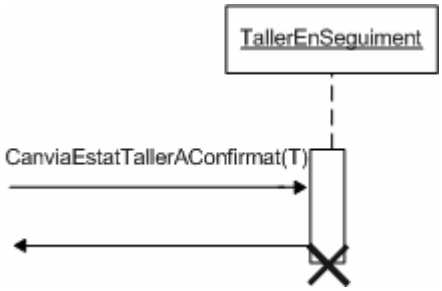
**Confirma Taller**

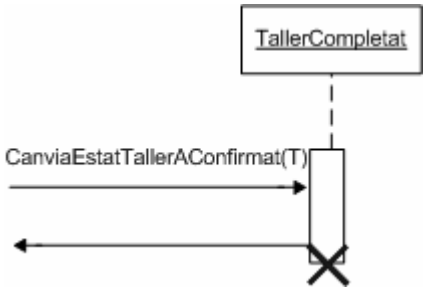


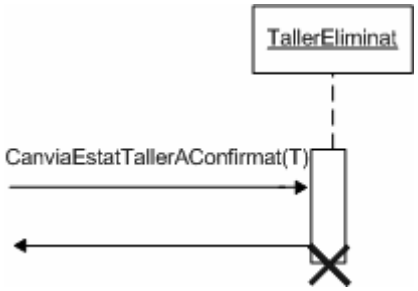



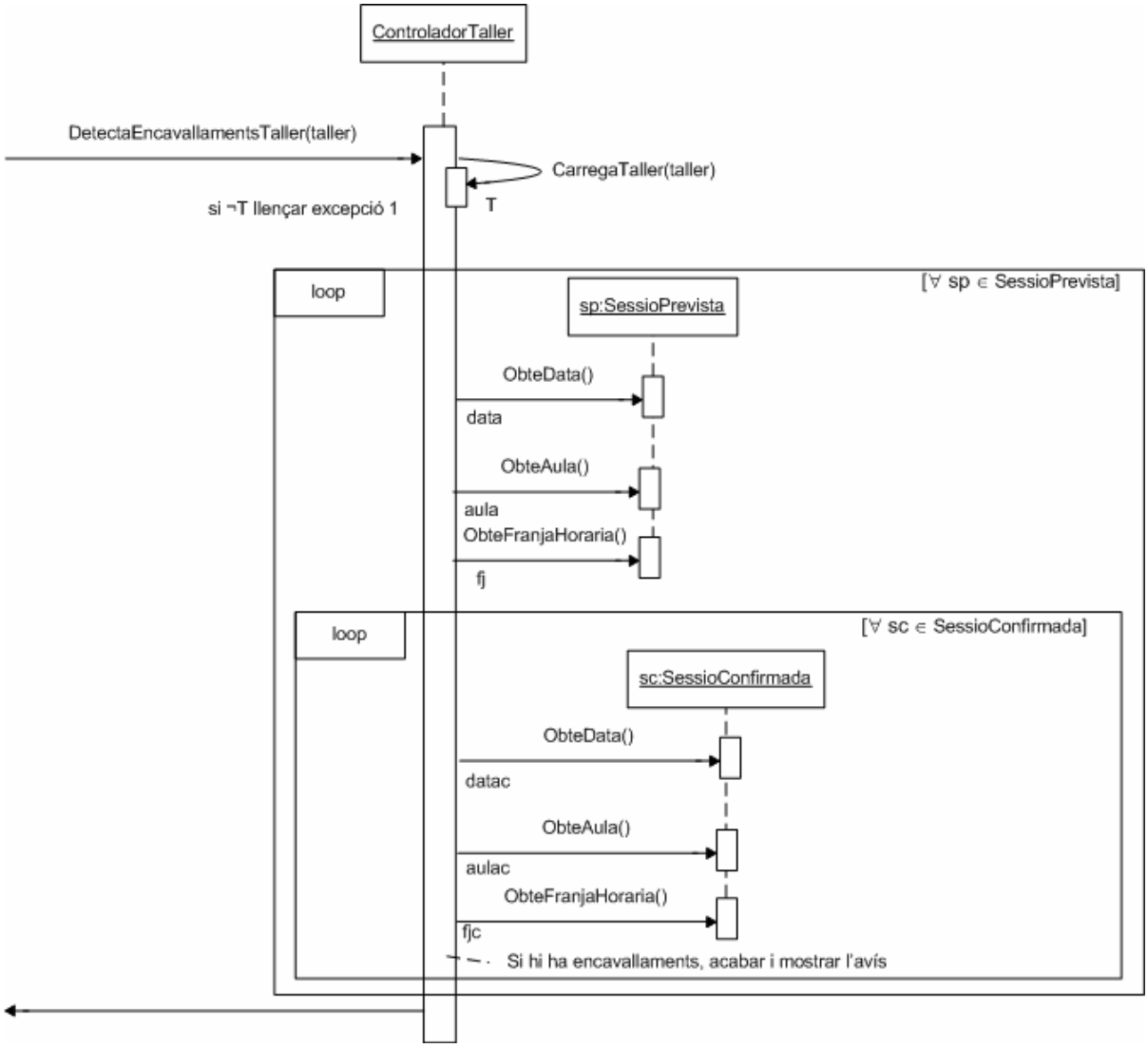

TallerConfirmat
CanviaEstatTallerAConfirmat(T)


TallerObert
CanviaEstatTallerAConfirmat(T)


TallerEnSeguiment
CanviaEstatTallerAConfirmat(T)


TallerCompletat
CanviaEstatTallerAConfirmat(T)


TallerEliminat
CanviaEstatTallerAConfirmat(T)


ControladorTaller
DetectaEncavallamentsTaller(taller)
CarregaTaller(taller)
T
si ¬T llençar excepció 1
loop
[∀ sp ∈ SessioPrevista]
sp:SessioPrevista
ObteData()
data
ObteAula()
aula
ObteFranjaHoraria()
fj
loop
[∀ sc ∈ SessioConfirmada]
sc:SessioConfirmada
ObteData()
datac
ObteAula()
aulac
ObteFranjaHoraria()
fjc
Si hi ha encavallaments, acabar i mostrar l'avís

### 3.2.4 Disseny lògic de la base de dades

El disseny lògic de la base de dades s'obté en transformar el diagrama de classes del disseny resultant de la normalització al model lògic relacional, de manera que totes les taules que apareguin estan en la Tercera Forma Normal.

La notació utilitzada per descriure les taules és la següent:

**Taula** (atribut$_1$, atribut$_2$, ... , atribut$_n$)
**Clau forània** (atribut$_k$)
**Referència** Taula Forània (atribut)
**Comprova** (expresió)

Els atributs subratllats formen part de la Clau Privada (Primary Key).
Tot seguit es mostren les taules que formen part de la base de dades d'aquest projecte:

**Centre**(nomCentre)

**Local**(nomLocal, adreça, telèfon, fax, seu, centre)

on { centre } referencia Centre

**Recurs**(nomRecurs, tipus_recurs)

**Consumible**(nomRecurs, preu, quantitat)

**Material**(nomRecurs, quantitat)

**Aula**(nomRecurs, tipus_aula, capacitat)

**Propia**(nomRecurs, local)

on { local } referencia Local

**Externa**(nomRecurs, adreça, telèfon, fax)

**Franjahoraria**(hora-inici,hora-fi)

**Diasetmana**(dia)

**DiaLaborable**(dataLaborable)

**Temporada**(anyTemporada, data_inici, data_fi)

**Periodedocent**(nomPeriodeDocent, data_inici, data_fi, temporada)

on { temporada } referencia Temporada

**DiaFestiu**(dataFestiva, temporada)

on { temporada } referencia Temporada

**Sessio**(aula, data,franjahoraria)

on { aula } referencia Aula

on { data } referencia DiaLaborable

on { franjahoraria } referencia FranjaHoraria

**SessioPrevista**(aula, data,franjahoraria,activitatPrevista)

on { aula } referencia Aula

on { data } referencia DiaLaborable

on { franjahoraria } referencia FranjaHoraria

on { activitatPrevista } referencia Activitat

**SessioConfirmada**(aula, data,franjahoraria, sessioFeta, numAssistents, motiuNoFeta,activitatConfirmada)

on { aula } referencia Aula

on { data } referencia DiaLaborable

on { franjahoraria } referencia FranjaHoraria

on { activitatConfirmada } referencia Activitat

**ReservaMaterial**(material,sessio,usuari)

on { material } referencia Material

on { sessio } referencia Sessio

on { usuari } referencia UsuariPersona

**UsuariPersona**(dni, tipus_usuari_persona, nom, cognom1, cognom2, privilegis, adreça, e-mail, telefon_fix, telefon_mobil, sexe, edat)

**Usuari**(dni, tipus_usuari)

**Soci**(dni, num_soci, descompte)

**Nosoci**(dni)

**Formador**(dni, CV, foto)

**Afinitat**(dni1,dni2)

on { dn1 } referencia Formador

on { dn2 } referencia Formador

**Administrador**(dni)

**Administratiu**(dni)

**FullinteresTema**(periode-docent, usuari)

on { periode-docent } referencia PeriodeDocent

on { usuari } referencia Usuari

**Disponibilitat**(franjaHoraria, diaSetmana, fullInteresTema)

on { franjaHoraria } referencia FranjaHoraria

on { diaSetmana } referencia DiaSetmana

on { fullInteresTema } referencia FullInteresTema

**Assistencia**(sessio,usuari, esConvidat)

on { sessio } referencia SessioConfirmada

on { usuari } referencia Usuari

**Imparteix**(sessio, formador)

on { sessio } referencia SessioConfirmada

on { formador } referencia Formador

**Activitat**(nomActivitat, tipus_activitat, preu, capacitat, minim_quorum, data_inici, periode-docent, tema)

on { periode-docent } referencia PeriodeDocent

on { tema } referencia Tema

**Taller**(nomActivitat, nombre_sessions, nivell, fictici, periodicitat, estat, data_inici_inscripcions, data_fi_inscripcions)

**Concurs**(nomActivitat, data_finalitzacio, data_entrega_premis, data_sorteig)

**Event**(nomActivitat, durada)

**FullinteresTaller**(periode-docent, usuari)

on { periode-docent } referencia PeriodeDocent

on { usuari } referencia Usuari

**Interes-Taller**(fullinteres, taller)

on { fullinteres } referencia FullinteresTaller

on { taller } referencia Taller

**Premi**(id, valor_material, concurs)

on { concurs } referencia Concurs

**Grup**(nom-grup)

**Participacio**(grup, concurs)

on { grup } referencia Grup

on { concurs } referencia Concurs

**Inscripcio**(id, activitat,usuari, pagada, paga_i_senyal, data_pagament, data_paga_i_senyal, metode_pagament, lloc_pagament)

on { activitat } referencia Activitat

on { usuari } referencia Usuari

**Grup-inscripcio**(grup, inscripcio )

on { grup } referencia Grup

on { inscripcio } referencia Inscripcio

**Tema**(nomTema, descripcio_breu, descripcio_llarga, foto, foto_emblema, logo, videos_vinculats, especialitat, temari, nivellDificultat)

on { especialitat } referencia Especialitat

**Especialitat**(nomEspecialitat, comunitat)

on { comunitat } referencia Comunitat

**Comunitat**(nomComunitat, descripcio)

**Interes-Tema**(fullinteres, tema)

on { fullinteres } referencia FullinteresTema

on { tema } referencia Tema

**SessioTemari**(numSessio, tema, contingut)

on { tema } referencia Tema

**Nivellexpertesa**(tema, usuari, nivell)

on { tema } referencia Tema

on { usuari } referencia Usuari

**Document**(nomDocument, publicat, enllaç, nivell_acces, tema)

on { tema } referencia Tema

**ConfiguracioSistema**(nomConfiguracio, fotoTema, fotoEmblemaTema, nivellDificultatTema, capacitatActivitat, minimQuorumActivitat, preuActivitat, numSessionsTaller, nivellTaller, periodicitatTaller, metodePagamentInscripcio, fotoFormador, descompteSoci, capacitatAula, tlfLocal, faxLocal, textCopyright, logoPrincipal, textPrincipal, textLinks, textQuiSom, textInteres, textHoraris, textActivitats, textFAQ, textInfoAdicional)

# 4 Implementació

## 4.1 Tecnologies utilitzades

A continuació es descriuen les tecnologies utilitzades en el desenvolupament del projecte.

### 4.1.1 Tecnologia web

World Wide Web es podria definir com un sistema d'informació multimèdia, hipertext i distribuït. És multimèdia perquè pot incloure text, gràfics, so i vídeo. L'hipertext és una forma d'escriure documents basada en la utilització del ratolí o el teclat per seleccionar una frase o imatge del document que ens dóna accés a un altre document hipertext (enllaç) que pot estar ubicat en qualsevol altre lloc.

El llenguatge d'escriptura de documents hipertext que s'utilitza en pàgines WWW és el HTML. L'arquitectura del WWW és una arquitectura client-servidor, és a dir, existeix una màquina encarregada de servir els documents que un client demana.

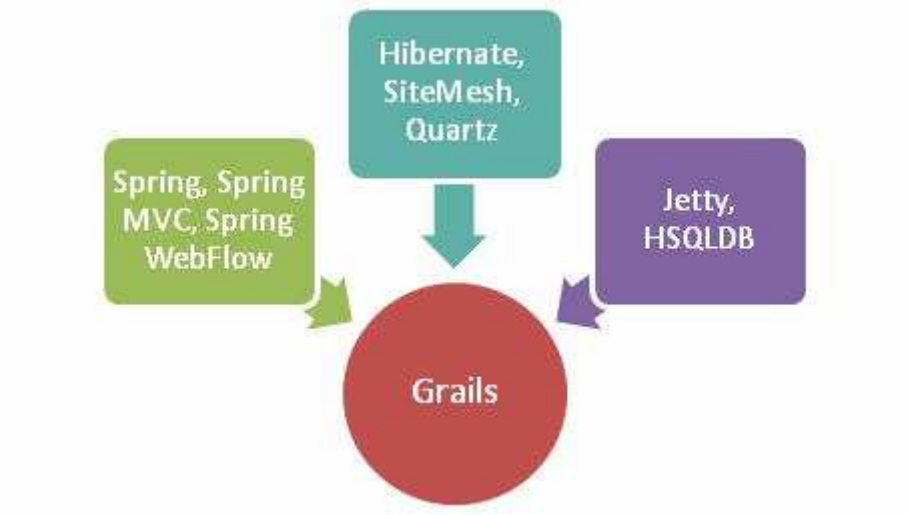
El protocol de comunicació utilitzat entre el navegador del client i el servidor web és el HTTP (Hypertext Transfer Protocol). Les capçaleres HTTP serveixen per a què el servidor web i el navegador es passen informació entre ells. Hi ha una versió de HTTP per a la transferència segura d'informació anomenada HTTPS (Hypertext Transfer Protocol Secure) que pot utilitzar qualsevol mètode de xifrat sempre que sigui entès tant pel servidor com pel client.

La comunicació en aplicacions web entre el client i el servidor és unidireccional, és a dir, el client realitza una petició al servidor i rep una resposta, i no al revés.

### 4.1.2 Grails

Grails és un framework web amb llicència open source desenvolupat sobre el llenguatge Groovy, que pretén ser un marc de treball altament productiu seguint els principis *Don't Repeat Your Self* i *Convention over Configuration,* proporcionant un entorn de desenvolupament estandarditzat i ocultant gran part dels detalls de

configuració al programador. Grails és alguna cosa més que un framework MVC, també ens ofereix capa de persistència, capa de servei, contenidor de servlets i gestor de bases de dades. Se sustenta sobre varis frameworks i llibrerias Java molt conegudes i provades com són Spring Framework, Hibernate, Sitemesh, Log4j, Jetty... i el llenguatge de programació Groovy.




*Combinació de tecnologies que engloba Grails*

Grails s'ha desenvolupat amb una sèrie d'objetius en ment:

- Oferir un framework web d'alta productivitat per la plataforma Java.
- Reutilitzar tecnologies Java ja provades com Hibernate i Spring sota una interfície simple i consistent.
- Oferir un framework consistent que redueixi la confusió i que sigui fàcil d'aprendre.
- Oferir documentació per les parts del framework relevants pels seus usuaris.
- Proporcionar el que els usuaris necessiten en àrees que tot sovint són complexes e inconsistents:
    - Framework de persistència potent i sòlid.
    - Patrons de visualització potents i fàcils d'usar amb GSP (*Groovy Server Pages*).
    - Biblioteques d'etiquetes dinàmiques per crear fàcilment components web.
    - Bon suport d'Ajax que sigui fàcil d'estendre i personalitzar.
- Proporcionar aplicacions d'exemple que mostrin la potència del framework.
- Proporcionar un entorn complet de desenvolupament, incloent un servidor web i recàrrega automàtica de recursos.

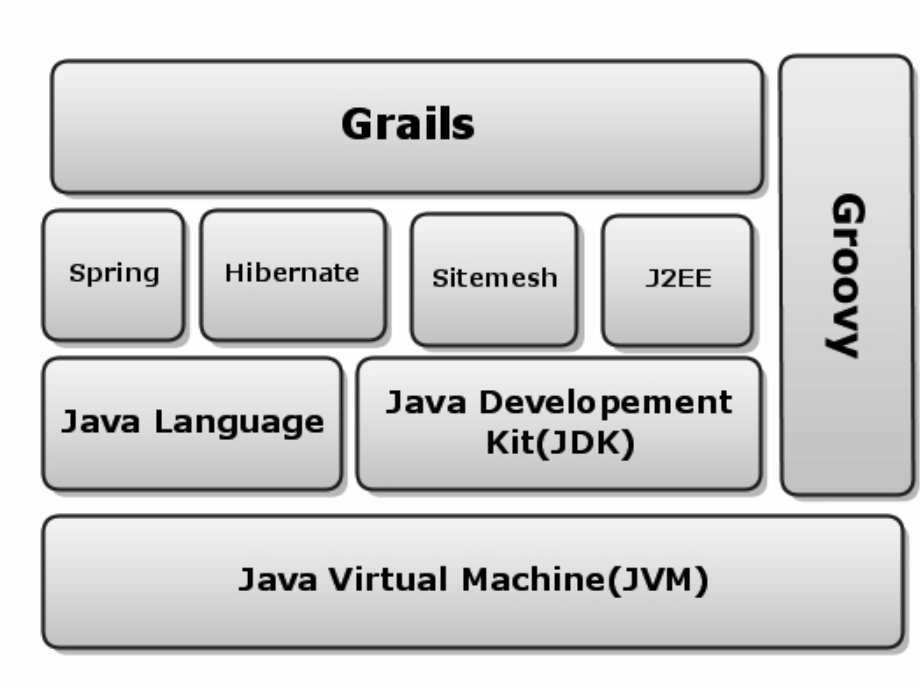
Grails s'ha dissenyat amb la finalitat que sigui fàcil d'aprendre, fàcil per desenvolupar aplicacions i extensible. Intenta oferir el balanç adequat entre consistència i funcionalitats potents. No obstant això, totes les funcionalitats que vagin més enllà del que Grails proveeix tenen una corba d'aprenentatge significativa, perquè requereixen el coneixement avançat de les tecnologies en les que es basa.

## Grails Framework

Grails està format per 5 components principals que defineixen la seva arquitectura:

- **Hibernate**: un gestor de la persistència per a BD relacionals. En aquest mateix capítol es descriu amb més profunditat.
- **Spring**: un framework d'aplicacions de la plataforma Java
- **Quartz**: un framework que permet realitzar tasques programades
- **SiteMesh**: un framework robust i estable per renderitzar layouts

A continuació es mostra una imatge sobre l'arquitectura de Grails:



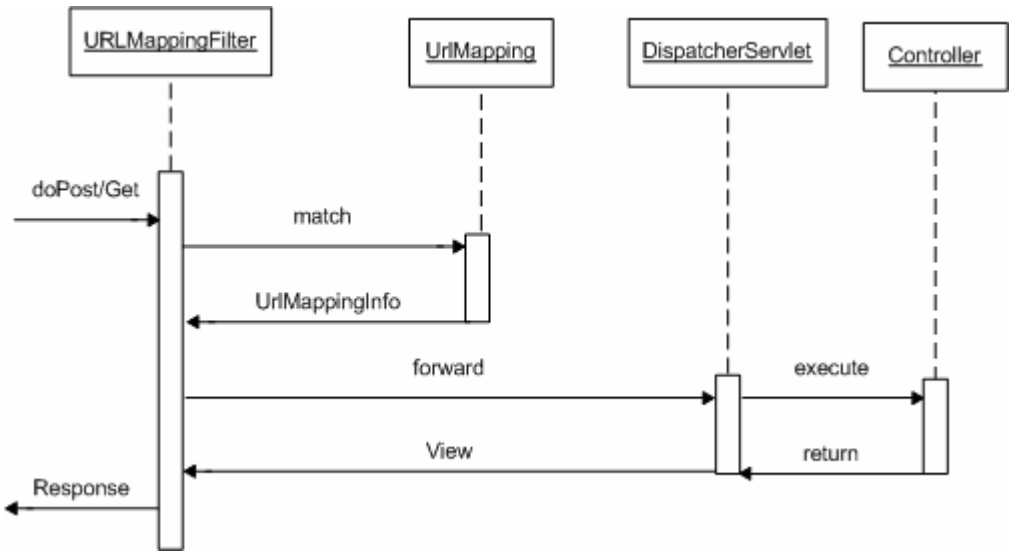

*Arquitectura de Grails*

Com podem veure, Grails és un framework de frameworks. Però la gran avantatge és que no necessitem entendre cap dels frameworks Spring, Quartz o SiteMesh ni saber com funcionen per treballar-hi. Aquest fet simplifica enormement la tasca del programador i a la vegada proveeix l'eina de certa robustesa, ja que totes 3 tecnologies són àmpliament utilitzades actualment.

## Algunes consideracions

- **Resposta a una petició HTML**

Grails codifica el mapeig de controladors del domini en la URL de l'aplicació. Per entendre millor com funciona es presenta el següent diagrama de seqüència.




*Resposta a una petició HTML a Grails*

- **Què suposa utilitzar un framework com Grails per al programador?**

Tot i que Grails té funcionalitats que facilitin la tasca de programació de sistemes, cal tenir present que disminuir la càrrega de treball en un desenvolupament no significa que es necessiti un equip humà menys capacitat. La càrrega de treball disminueix perquè aquests tipus de frameworks segueixen el principi *No et Repeteixis* (DRY). Això significa que el desenvolupador pot produir més ràpid perquè no ha de realitzar tasques repetitives. Això és tot. Però el programador encara haurà d'enfrontar-se a problemes, i haurà de prendre decisions sobre el disseny per resoldre-les.

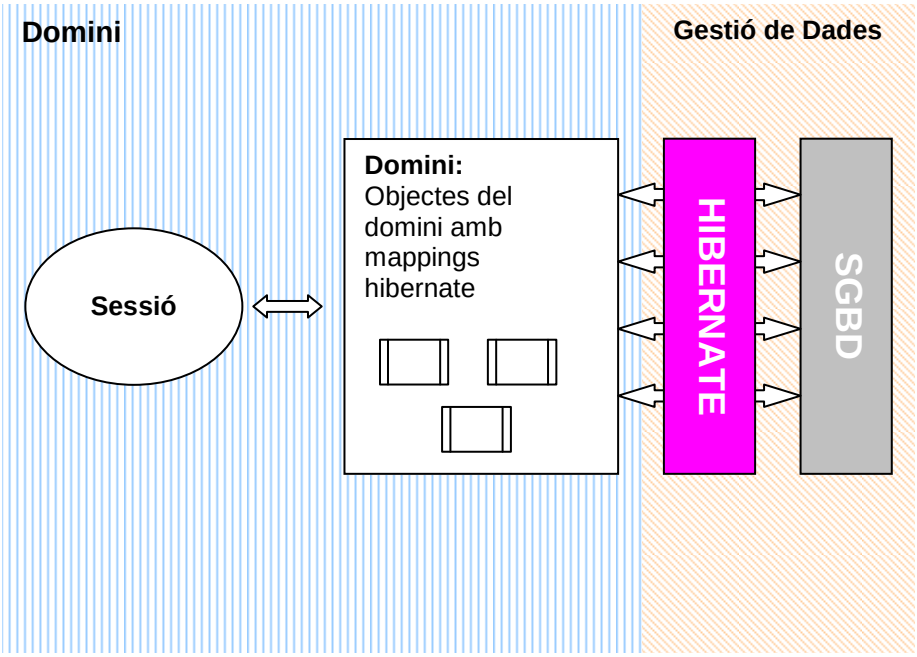
A més, seguint el principi de Grails, el codi de les bones pràctiques per programar amb aquest framework és molt més curt que amb d'altres llenguatges com Java, provocant certes dificultats en la resolució de problemes. Aquest fet es detallarà més en l'apartat 4.2.1 del llenguatge Groovy.

### 4.1.3 Hibernate

El sistema treballa amb una BD Relacional, mentre que el model està orientat a objectes, i això podria suposar feina addicional a l'hora de guardar i recuperar informació. Per això Grails utilitza Hibernate. Hibernate és una tecnologia de persistència d'objectes que intenta simplificar la feina de connectar bases de dades relacionals i el llenguatge Java utilitzant un model de programació molt transparent i poc exigent.

Per guardar objectes Java s'han de serialitzar objectes estructurats en forma d'arbre a una base de dades relacional estructurada de forma tabular i a la inversa. Per a fer això és essencial mapejar els objectes Java en columnes i registres de la base de dades d'una manera optimitzada en velocitat i eficiència.

Hibernate és un marc de treball Java que proporciona mecanismes de mapeig objecte/relacional per a definir com s'emmagatzemen, eliminen, actualitzen i recuperen els objectes Java. A més ofereix serveis de consulta i recuperació que poden optimitzar els esforços de desenvolupament dins d'entorns SQL i JDBC.




*Esquema d'integració d'Hibernate a una aplicació*

A més, ofereix un llenguatge de consultes que agrupa un potent i flexible mecanisme de consulta, emmagatzemament, actualització i recuperació d'objectes des d'una base de dades. Aquest llenguatge, l'HQL (Hibernate Query Language), és una extensió orientada a objectes d'SQL. Permet accedir a les dades de diferents formes, incloent consultes orientades a objectes. Grails proporciona per defecte la configuració per utilitzar HQL però donat que s'ha utilitzat una BD relacional MySQL no s'ha fet servir, sinó que s'han programat directament les consultes en SQL.

## 4.2 Llenguatges utilitzats

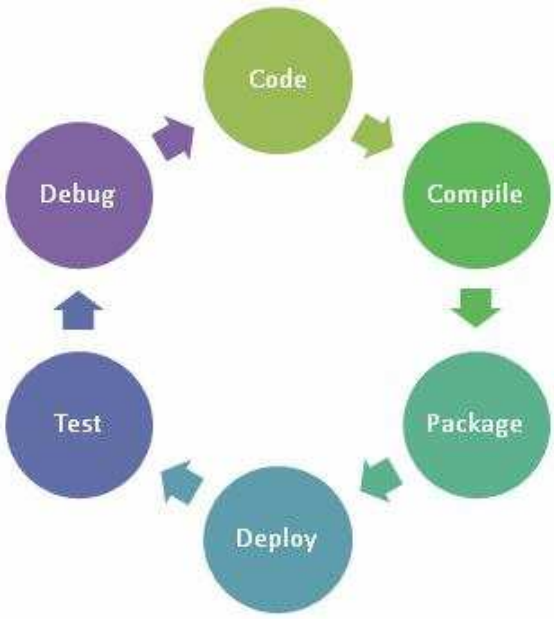
A continuació es descriuen els llenguatges utilitzats per la implementació del projecte.

### 4.2.1 Groovy

**Groovy** és un llenguatge de programació orientat a objectes implementat sobre la plataforma Java amb llicencia open source sota Apache 2.0. Té característiques similars a d'altres llenguatges coneguts com Python, Ruby, Perl i Smalltalk. Groovy és un estàndard (el JR-241) i els seus desenvolupadors volen que sigui inclós com a component oficial de la plataforma Java.

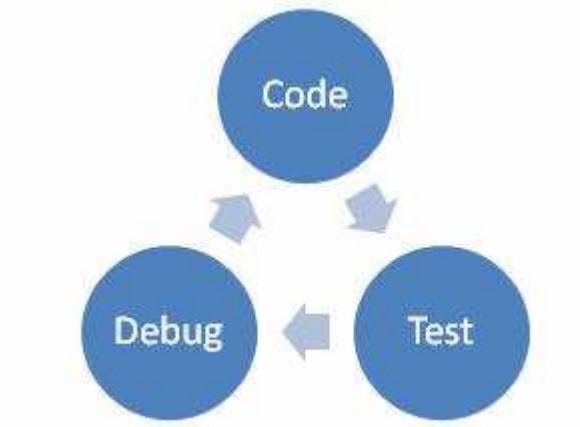
Groovy usa una sintaxi molt semblant a Java, comparteix el mateix model d'objectes, de fils i de seguritat. Des de Groovy podem accedir directament a totes les API existents de Java. Pot ser emprat en qualsevol aplicació Java doncs està preparat per ser utilitzat per la Java Virtual Machine (JVM). Així mateix la major part de codi escrit en Java és totalment vàlid en Groovy. També resulta un llenguatge més compacte ja que permet deixar de banda alguns elements que són requerits en Java. Aquestes característiques fan que aquest llenguatge sigui de molt fàcil adopció per a programadors Java donat que la corba d'aprenentatge es redueix molt més que amb d'altres llenguatges.

Groovy es un llenguatge dinàmic, és a dir, és capaç de fer en temps d'execució coses que d'altres llenguatges (anomenats estàtics) només poden fer durant la compilació. Per exemple, un llenguatge estàtic només permet definir tipus de dades en el seu codi font i emprant el compilador per generar la seva representació binària. Només llavors podem executar una aplicació que utilitzi aquests tipus de dades que hem creat.




*Cicle de desenvolupament d'una aplicació Java*

En canvi, utilitzar un llenguatge dinàmic ens permet crear tipus de dades o modificar existents **mentre l'aplicació s'està executant**. D'aquesta manera, podem escriure programes dels quals el seu comportament pot adaptar-se a les condicions en que estiguin essent executats.




*Cicle de desenvolupament d'una aplicació com Grails basada en el llenguatge Groovy*

**Llenguatges dinàmics**

La gran avantatge dels llenguatges dinàmics és que, en general, es necessita molt menys codi per realitzar una determinada tasca que si la escrivíssim amb un llenguatge estàtic. La seva sintaxi sol ser molt més expressiva i breu, el

qual fa que tinguem una major productivitat a l'hora de programar i obtinguem un codi més fàcil de comprendre (en llenguatge natural) i mantenir.

Però els llenguatges dinàmics per si mateixos tampoc són la solució perfecta a tots els problemes. La seva avantatge és a la vegada l'origen de la seva limitació: gran part del que passa en un programa escrit en un d'aquests llenguatges no figura per cap part del seu codi font. Els tipus i funcions generats de forma dinàmica no estan descrits en cap lloc, i per tant els errors que puguin contenir són més difícils d'identificar i resoldre. A més la seva mateixa natura fa que sigui més difícil escriure entorns de desenvolupament per aquest tipus de llenguatges que inclouen ajudes al desenvolupament com auto-completat o operacions de debug.

D'aquesta manera, els llenguatges estàtics aporten suport per traces, estabilitat i són robustos, el que els fa més aptes per resoldre tasques crítiques ja sigui per la seva funcionalitat o pels requisits de rendiment. A canvi, exigeixen més esforç en forma de codis font més llargs, difícils de llegir i escriure. Els llenguatges dinàmics, en canvi, ens presenten un entorn àgil en el que els problemes es resolen amb molt poques línies de codi, i no necessitem seguir el cicle "escriure – compilar – executar" ja que la majoria es poden executar mitjançant intèrprets directament des de el codi font. A canvi, perdem una part del control sobre el que passa durant l'execució dels nostres programes, el que pot dificultar la resolució d'incidències en alguns tipus d'aplicacions.

### 4.2.2 SQL

Degut a la diversitat de llenguatges i de bases de dades existents, la manera de comunicar entre els uns i els altres seria realment complicada de gestionar de no ser per l'existència d'estàndards que ens permetin realitzar les operacions bàsiques d'una forma universal.

Aquesta és la funció del **SQL** (**Structured Query Language**) que és un llenguatge estàndard de comunicació amb bases de dades. Es tracta d'un llenguatgie normalitzat que ens permet treballar amb qualsevol tipus de llenguatge (ASP, PHP, etc.) juntament amb qualsevol tipus de base de dades (MS Access, SQL Server, MySQL...). La utilització d'aquest llenguatge ens assegura que les consultes són independents del tipus de Base de Dades que tinguem.

### 4.2.3 HTML

El **HTML** (**HyperText Markup Language**), és un llenguatge senzill basat en etiquetes utilitzat per a a creació de pàgines web, independentment de la plataforma en la que s'utilitzi.

La presentació de la pàgina depèn directament del navegador utilitzat. El mateix document no té el mateix format en pantalla si es visualitza amb diferents navegadors (Chrome, Explorer, Mozilla...). HTML es limita a descriure l'estructura i el contingut d'un document i no el format de la pàgina en pantalla.

Una de les claus principals del HTML, a més de la presentació, són la organització i la coherència. Totes las pàgines web tenen una estructura basada en etiquetes que facilita el seu ús i aprenentatge. Les etiquetes o tags descriuen el contingut del document i el comportament dels elements. Aquestes etiquetes són fragments de text que van entre els símbols < > i poden tenir atributs.

Un document HTML ha de ser delimitat per l'etiqueta <html> i </html>. Dins del document, podem així mateix distingir dues parts principals: la capçalera, delimitada per <head> i </head> on colocarem etiquetes de tipus informàtic i el cos, delimitat per les etiquetes <body> i </body>, que serà on colocarem el contingut de la pàgina.

El resultat és un document amb la següent estructura:

```
<HTML>
        <HEAD>
         …
        </HEAD>
        <BODY>
         …
        </BODY>
</HTML>
```

HTML és un llenguatge molt senzill, però bastant limitat. Per aquesta raó generalment s'utilitza juntament amb d'altres llenguatges com són Javascript, que ofereixen funcionalitats addicionals.

### 4.2.4 JavaScript

**JavaScript** és un llenguatge de programació bastant senzill, fins i tot per a persones no expertes. S'utilitza per a crear petites funcions encarregades de realitzar certes accions dintre de l'àmbit d'una pàgina web. Es tracta d'un llenguatge de programació que actua sobre el client, pel que és el navegador que suporta la càrrega de processament. Gràcies a la seva compatibilitat amb la majoria dels navegadors moderns, és el llenguatge de programació més utilitzat en la part client.

Amb JavaScript podem crear efectes especials a les pàgines i definir interactivitats amb l'usuari. El navegador del client és l'encarregat d'interpretar les instruccions JavaScript i executar-les, de manera que, el major recurs de què disposa aquest llenguatge és el propi navegador. Entre les accions típiques que es poden realitzar en JavaScript tenim tres vessants:

- Efectes especials sobre pàgines web per a crear continguts dinàmics i elements de la pàgina que tinguin moviment, canviïn de color o qualsevol altre dinamisme.
- Executar instruccions com resposta a les accions de l'usuari, amb el que podem crear **pàgines interactives**.
- Verificacions de les dades inserides per un usuari en un formulari que deguin tenir un tractament previ abans de ser enviats al servidor. D'aquesta manera s'aconsegueix descarregar una mica el treball del servidor disminuint el temps de resposta del mateix.

JavaScript és un llenguatge amb moltes possibilitats. Permet la programació de petits *scripts*, i també de programes més grans, orientats a objectes, amb funcions, estructures de dades complexes, etc. A més, JavaScript posa a la disposició del programador tots els elements que composen la pàgina web, perquè aquest pugui accedir-hi i modificar-los dinàmicament.

Al contrari que d'altres llenguatges com Java, JavaScript no és un llenguatge que necessiti que els seus programes es compilin, sinó que aquests s'interpreten per part del navegador quan aquest llegeix la pàgina.

El més important i bàsic és que la programació de JavaScript es realitza dintre del propi document **HTML**. Això vol dir que en la pàgina es barregen els dos llenguatges de programació. Perquè aquests dos llenguatges es puguin barrejar sense problemes s'han d'incloure uns delimitadors que separen les etiquetes HTML de les instruccions. Aquests delimitadors són les etiquetes <SCRIPT> i </SCRIPT> Tot el codi JavaScript que s'inclogui en la pàgina ha de ser introduït entre aquestes dues etiquetes. En una mateixa pàgina podem introduir diversos scripts dintre d'etiquetes <SCRIPT> diferents. La col·locació d'aquests scripts dintre de la pàgina és indiferent, excepte en el cas de treballar amb capes. També es pot escriure JavaScript dins de determinats atributs de la pàgina relacionats amb les accions de l'usuari i l'ús d'esdeveniments. En aquest cas no cal posar els delimitadors <SCRIPT>.

### 4.2.5 CSS

Les fulles d'estil en cascada (***Cascading Style Sheets***) són un llenguatge formal utilitzat per a definir la presentació d'un document estructurat escrit en HTML o XML. El W3C (World Wide Web Consortium) és l'encarregat de formular l'especificació de les fulles d'estil que servirà d'estàndard per als navegadors. La idea que motiva el desenvolupament de CSS és separar l'estructura d'un document de la seva presentació.

Per exemple, l'element d'HTML <H1> indica que un bloc de text és un encapçalament i que és més important que un bloc etiquetat com <H2>. HTML permet atributs extra dins de l'etiqueta oberta per donar-li format (color, tamany de la font...). No obstant, cada etiqueta <H1> ha de disposar d'aquesta informació si es desitja un disseny consistent per una pàgina, i a més, una persona que llegeixi aquesta pàgina amb un navegador perd totalment el control sobre la visualització del text.

Amb el CSS, l'etiqueta <H1> no hauria de proporcionar informació sobre com es visualitzarà, només marca l'estructura del document. La informació d'estil

separada en una fulla d'estil, especifica com s'ha de mostrar <H1>: color, font, alineació del text, tamany i altres.

La informació d'estil es pot adjuntar de diferents formes, com ara un document separat, en el mateix document HTML o en cada etiqueta. Aquesta sintaxi CSS permet aplicar al document un format de manera molt més exacta:

- Deixa definir la distància entre línies del document.
- Es pot aplicar indentitat a les primeres línies del paràgraf.
- Permet col·locar elements en la pàgina amb major precisió, i sense lloc a errors.
- Es pot definir la visibilitat dels elements, marges, subratllats, ratllats...
- Definir atributs amb píxels (*px), percentatges (%), polzades (*in), punts (*pt) i centímetres (cm).

Els principals avantatges d'utilitzar CSS són:

- Control centralitzat de la presentació d'un lloc web complet, de manera que se n'agilitza de forma considerable l'actualització.
- Els navegadors permeten als usuaris especificar la seva pròpia fulla d'estil local que serà aplicada a un lloc web remot, amb el que augmenta considerablement l'accessibilitat. Persones amb deficiències visuals poden definir tamanys de lletra grans o remarcar més els enllaços.
- Una pàgina pot disposar de diferents fulles d'estil en funció del dispositiu que la mostri o fins i tot a elecció de l'usuari. Per exemple, per a ser impresa, mostrada en un dispositiu mòbil o ser "llegida" per un sintetitzador de veu.
- El document HTML és més fàcil d'entendre i s'aconsegueix reduir considerablement el seu tamany.

Existeixen vàries versions: CSS1 i CSS2, amb CSS3 en desenvolupament per el W3C. Els navegadors moderns implementen CSS2 força bé malgrat que existeixen petites diferències d'implementació en funció de les marques i les versions dels navegadors.

Exemple de codi CSS:

```
body {
  padding: 2em 1em 2em 70px;
  margin: 0;
  font-family: sans-serif;
  color: black;
  background: white;
  background-position: top left;
  background-attachment: fixed;
  background-repeat: no-repeat;
}
:link { color: #00C; background: transparent }
:visited { color: #609; background: transparent }
a:active { color: #C00; background: transparent }

a:link img, a:visited img { border-style: none }
```

## 4.3 Codificació

Com a exemple de la codificació realitzada, s'ha considerat interessant mostrar algunes funcions d'algun controlador de l'aplicació. S'ha escollit el controlador **d'UsuariPersona**. A continuació es mostra una selecció de les operacions implementades en aquest controlador, ja que per motius de volum i claredat s'ha considerat innecessari mostrar-les al complet. S'ha preferit fer-ho així per a que l'apartat no resulti massa extens.

### 4.3.1 Controlador d'UsuariPersona

```
package gwalproject

class UsuariPersonaController {

    static allowedMethods = [save: "POST", update: "POST", delete: "POST"]

    def index = {
        redirect(action: "list", params: params)
    }

////////////////////////////////////
//codi per a l'autoritzacio
////////////////////////////////////

  def jcaptchaService

  def beforeInterceptor = [action:this.&auth,
      except:['login', 'logout', 'autenticar', 'register',
        'handleRegistration']]

  def auth() {

    if(session.user && session.user.usuari){
      flash.message = "No tens privilegis per accedir a aquesta zona"
      redirect(controller:"prohibit", action:"index")
      return false
    }
  }

  def debug(){
    println "DEBUG: ${actionUri} called."
    println "DEBUG: ${params}"
  }

  def login = {
      if(request.getSession(false) && session.user){
      flash.message = "Has de fer Logout abans de fer Login"
      redirect(controller:"taller", action:"list")
      }
  }

  def autenticar = {
    def usuari = UsuariPersona.findByLoginAndPassword(params.login,
         params.password.encodeAsSHA())
     if(usuari){
```

```
        session.user = usuari
        flash.message = "Benvingut ${usuari.nom}!"
        redirect(controller:"avisos", action:"index")
      }else{
        flash.message = "Usuari ${params.login} o contrasenya incorrectes.
               Torna-ho a intentar."
         redirect(action:"login")
      }
    }

    def register = {
        if(request.getSession(false) && session.user){
        flash.message = "Has de fer Logout abans de poder fer un registre"
        redirect(controller:"avisos", action:"index")
        }
        else{
      def usuariPersonaInstance = new UsuariPersona()
          usuariPersonaInstance.properties = params
          return [usuariPersonaInstance: usuariPersonaInstance]
        }
    }

    def handleRegistration = {
      def usuariPersonaInstance = new UsuariPersona(params)
      def captchaText = session.captcha
      session.captcha = null
      if (params.captcha.equals("") || !jcaptchaService.validateResponse(
         "codi", session.id, params.captcha)){

          /* No ha omplert bé el captcha */
          log.info "Captcha Not Filled In"
          flash.message = "No has introduit bé el codi de la imatge"
          redirect(action:'register')
      }
      else{    /* Usuari ha omplert bé el captcha */
            if(params.password != params.confirm) {
                flash.message = "Les contrasenyes que has introduit no
                      coincideixen"
                redirect(action:register)
            }
            else {
                log.info "Abans de salvar"
                // Let's hash the password
                usuariPersonaInstance.properties = params
                params.password = params.password.encodeAsSHA())
                if (usuariPersonaInstance.save(flush: true)) {

                flash.message = "Registre finalitzat.Pots entrar amb
                      l'usuari i la contrasenya escollides"
                 redirect(controller:'login')
                }
                else {
                log.info "no s'ha pogut salvar"
                flash.usuariPersonaInstance = usuariPersonaInstance
                render(view: "register", model: [usuariPersonaInstance:
                       usuariPersonaInstance])
                }
            }
      }
    }

    def logout = {
      session.user = null
      session.invalidate()
      flash.message = "Has deixat d'estar autenticat en l'aplicació"
      redirect(controller:"avisos", action:"index")
    }
```

```
def list = {
    params.max = Math.min(params.max ? params.int('max') : 10, 100)
    [usuariPersonaInstanceList: UsuariPersona.list(params),
      usuariPersonaInstanceTotal: UsuariPersona.count()]
}

def create = {
    def usuariPersonaInstance = new UsuariPersona()
    usuariPersonaInstance.properties = params
    return [usuariPersonaInstance: usuariPersonaInstance]
}

def save = {
    def usuariPersonaInstance = new UsuariPersona(params)
    if (usuariPersonaInstance.save(flush: true)) {
        flash.message = "${message(code: 'default.created.message',
          args: [message(code: 'usuariPersona.label', default:
          'UsuariPersona'), usuariPersonaInstance.id])}"
        redirect(action: "show", id: usuariPersonaInstance.id)
    }
    else {
        render(view: "create", model: [usuariPersonaInstance:
          usuariPersonaInstance])
    }
}

def show = {
    def usuariPersonaInstance = UsuariPersona.get(params.id)
    if (!usuariPersonaInstance) {
        flash.message = "${message(code: default.not.found.message',
          args: [message(code: 'usuariPersona.label', default:
          'UsuariPersona'), params.id])}"
        redirect(action: "list")
    }
    else {
        [usuariPersonaInstance: usuariPersonaInstance]
    }
}

def edit = {
    def usuariPersonaInstance = UsuariPersona.get(params.id)
    if (!usuariPersonaInstance) {
        flash.message = "${message(code: default.not.found.message',
          args: [message(code: 'usuariPersona.label',
          default: 'UsuariPersona'), params.id])}"
        redirect(action: "list")
    }
    else {
        return [usuariPersonaInstance: usuariPersonaInstance]
    }
}

def update = {
    def usuariPersonaInstance = UsuariPersona.get(params.id)
    if (usuariPersonaInstance) {
        if (params.version) {
            def version = params.version.toLong()
            if (usuariPersonaInstance.version > version) {

                usuariPersonaInstance.errors.rejectValue("version",
                    "default.optimistic.locking.failure",
                     [message(code: 'usuariPersona.label', default:
                     'UsuariPersona')] as Object[], "Another user
                      has updated this UsuariPersona while you were
                       editing")
                render(view: "edit", model: [usuariPersonaInstance:
```

```
                        usuariPersonaInstance])
                    return
                }
            }
            usuariPersonaInstance.properties = params
            if (!usuariPersonaInstance.hasErrors() &&
usuariPersonaInstance.save(flush: true)) {
                flash.message = "${message(code:
                    default.updated.message', args: [message(code:
                    'usuariPersona.label', default: 'UsuariPersona'),
                     usuariPersonaInstance.id])}"
                redirect(action: "show", id: usuariPersonaInstance.id)
            }
            else {
                render(view: "edit", model: [usuariPersonaInstance:
                    usuariPersonaInstance])
            }
        }
        else {
            flash.message = "${message(code:
               'default.not.found.message', args: [message(code:
                'usuariPersona.label', default:
                 UsuariPersona'), params.id])}"
            redirect(action: "list")
        }
    }

    def delete = {
        def usuariPersonaInstance = UsuariPersona.get(params.id)
        if (usuariPersonaInstance) {
            try {
                usuariPersonaInstance.delete(flush: true)
                flash.message = "${message(code:
                   'default.deleted.message', args: [message(code:
                    'usuariPersona.label', default:
                    'UsuariPersona'), params.id])}"
                redirect(action: "list")
            }
            catch
(org.springframework.dao.DataIntegrityViolationException e) {
                flash.message = "${message(code:
                   'default.not.deleted.message', args: [message(code:
                   'usuariPersona.label', default:
                   UsuariPersona'), params.id])}"
                redirect(action: "show", id: params.id)
            }
        }
        else {
            flash.message = "${message(code:
                 'default.not.found.message', args: [message(code:
                  'usuariPersona.label', default:
                  'UsuariPersona'), params.id])}"
            redirect(action: "list")
        }
    }
}
```

## 4.4 Control d'accès

Una part important de moltes aplicacions web recau en la capacitat d'identificar usuaris i controlar l'accés a les pàgines que formen el lloc web.

Es coneix com **autenticació** l'acte de determinar la identitat de l'entitat sol·licitant. En general, l'usuari haurà de presentar les seves credencials, com el nom d'usuari i la contrasenya, per a ser autenticat. Quan es trobi disponible una identitat autenticada, s'haurà de determinar si aquesta pot tenir accés a un recurs específic. Aquest procés es coneix com **autorització**.

En existir diferents perfils d'usuari, el control d'accés ha estat molt present en la implementació de la nostra aplicació. Tot seguit es descriuen els controls d'accés que s'han realitzat en el procés de desenvolupament:

- **Autenticació dels usuaris des de la pàgina principal**. Com s'ha comentat, l'autenticació es realitza des de la pàgina principal mitjançant un mòdul d'accés que comprova si l'usuari està donat d'alta a la base de dades i, si és així, obté el seu perfil.

- **Autorització d'accés a una pàgina.** Cada perfil d'usuari té accés a diferents pàgines i no té permès l'accés a d'altres. Es controla a nivell de filtre si el perfil que intenta accedir pot visualitzar la pàgina o no. És bàsic que no es permeti l'accés a una pàgina concreta si es coneix el seu URL i s'escriu al navegador, ja que estaríem comprometent la seguritat del lloc. A la nostra aplicació, sempre que l'accés no sigui permès, es torna a enviar a l'usuari a la pàgina principal.

- **Control d'inici de sessió.** L'aplicació ha de controlar que l'inici de sessió es faci des de qualsevol pàgina del sistema. D'aquesta manera, l'usuari pot consultar informació del lloc i quan necessiti obtenir més informacions o fer alguna operació, podrà entrar a la intranet i continuar treballant.

- **Accés a pàgines visitades en cache.** S'ha d'evitar que si dos usuaris comparteixen el mateix ordinador, un d'ells pugui visualitzar alguna de les pàgines visitades per l'altre guardades dins la memòria cache. Això es controla des de cadascuna de les pàgines a través del paràmetre de Sessió, que obliga a que aquesta expiri o bé quan es tanca o bé després d'un llarg temps d'inactivitat. D'aquesta forma no es pot accedir a les dades o pàgines visitades d'un altre usuari.

- **Evitar registres malintencionats d'usuaris.** Un dels perills que hi ha avui en dia a la xarxa és l'accés obert a tothom. Això pot suposar un problema a l'hora de programar formularis doncs existeixen nombrosos programes robot anomenats aranyes que exploren la xarxa omplint formularis de webs amb publicitat. Per evitar que això passi en l'únic formulari públic del sistema GWAL s'ha decidit utilitzar un mecanisme anomenat Capcha. Capcha és una funcionalitat de Grails que consisteix en representacions gràfiques d'una cadena alfanumèrica generada aleatòriament que serveix precisament per validar les dades introduïdes dins un formulari, i al mateix temps preveuen l'spam en els formularis.

- **Protecció de la contrasenya.** Després una cerca sobre mecanismes de seguretat, s'ha decidit implementar l'algorisme SHA *(Secure Hash Algorithm)* per la protecció de la contrasenya. SHA consisteix en un sistema de funcions criptogràfiques Tot i que SHA té la seguretat compromesa (al 2005 va rebre atacs al seu mecanisme de seguretat) calen $2^{69}$ operacions per obtenir la contrasenya. S'està treballant en una nova versió d'aquest algorisme per resistir els atacs. Tot i així, existeixen mecanismes per augmentar la dificultat en temps per poder esbrinar la contrasenya. Un dels més senzills i eficaços és el de codificar la contrasenya un nombre elevat de vegades per tal de multiplicar per aquest nombre el temps necessari per desencriptar una paraula. En aquest projecte, s'ha encriptat la contrasenya dels usuaris amb SHA-1.

En Grails existeixen dues maneres d'implementar permisos en un sistema: mitjançant **filtres** o bé **interceptors**. Els interceptors permeten una granularitat molt fina ja que es col·loca un interceptor a cada controlador de domini. No obstant, els interceptors tenen l'inconvenient de que moltes vegades es repeteix codi de comprovacions comunes innecessàriament. En canvi, els filtres s'estableixen en un nivell més global i permeten un control total sobre els controladors i les seves accions. Els filtres no tenen una granularitat tan fina com els interceptors però es pot arribar a fer el mateix amb ambdues solucions. La declaració d'un filtre es fa amb llenguatge d'expressions regulars que són avaluades en tres estats possibles: abans de que el controlador rebi la petició, després de que el controlador respongui i abans que la vista rebi les dades i per últim després que la vista hagi estat renderitzada.

S'ha considerat d'interès incloure aquí els cinc fitxers que descriuen els permisos al sistema GWAL.

```
package gwalproject

class LectorFilters {

//definim els filtres per a l'usuari no autenticat.Tindrà els seus propis mètodes al controlador
def filters = {
    lectorBloquejar(controller:'*', action:"(list|show|create|edit|update|delete|save)") {
      before = {
        if(!session.user){
          flash.message = "No tens privilegis per accedir a aquesta zona"
          redirect(controller:"prohibit", action:"index")
          return false
        }
      }

    }

    lectorNoPotInscripcions(controller:'inscripcio', action:"(salvarInscripcio|crearInscripcio)") {
        before = {
          if(!session.user){
          flash.message = "No tens privilegis per realitzar aquesta operació"
          redirect(controller:"prohibit", action:"index")
          return false
          }
        }
    }
  }
}
```

```
package gwalproject

class UsuariFilters {

    def filters = {
      llistarTipusGestio(controller:'activitat|taller|event|concurs', action:"(list)") {
          before = {
            if(session.user && session?.user?.usuari){
              redirect(controller:"prohibit", action:"index")
              return false
            }
          }
      }
    }
}
```

```
package gwalproject

class FormadorFilters {

    def filters = {
      formadorAndUpOnly(controller:'tema|nivellExpertessa|sessioTemari|especialitat|comunitat|
                        document|usuariPersona|usuari|usuariSoci|usuariNoSoci|assistencia|
                        reservaMaterial', action:"(list|create|show|edit|update|delete|save)") {
        before = {
            if(session.user && !(session?.user?.admin || session?.user?.administratiu || session?.user?.formador)){
              redirect(controller:"prohibit", action:"index")
              return false
            }
        }
      }

      //casos especials formador.
      formadorEspecials(controller:'sessioConfirmada',action:"(list|show|edit|update|save)") {
          before = {
            if(session.user && !(session?.user?.admin || session?.user?.administratiu || session?.user?.formador)){
              redirect(controller:"prohibit", action:"index")
              return false
            }
          }
      }
    }
}
```

```
package gwalproject

class AdministratiuFilters {

    def filters = {
        //funcionalitats que nomes pot fer l'administrador i l'administratiu
       administratiuAndUpOnly(controller:'activitat|taller|event|concurs|inscripcio|
                              grup|premi|formador|sessio|sessioPrevista|periodeDocent|
                              temporada|fullInteresTema|fullInteresTaller|disponibilitat|
                              diaSetmana|franjaHoraria|dataDia|diaFestiu|diaLaborable|recurs|
                              consumible|material|aula|propia|externa|
                              local|centre', action:"(list|create|show|edit|update|delete|save)") {
         before = {
           if(session.user && !(session?.user?.admin || session?.user?.administratiu)){
             redirect(controller:"prohibit", action:"index")
             return false
           }
         }
       }

       administratiuAndUpOnlyMesConcret(controller:'sessioConfirmada',action:"(create|delete)") {
           before = {if(session.user==null && !(session?.user?.admin || session?.user?.administratiu)){
               redirect(controller:"prohibit", action:"index")
               return false
             }
           }
       }
    }
}
```

```
package gwalproject

class AdminFilters {
  def filters = {
    //filtres nomes per l'administrador
    adminOnly(controller:'administratiu|administrador|configuracioSistema',
              action:"(list|create|show|edit|update|delete|save)") {
      before = {
        if(!session?.user?.admin){
          redirect(controller:"prohibit", action:"index")
          return false
        }
      }
    }
  }
}
```

## 4.5 Proves

Les proves han de permetre determinar si una implementació del sistema satisfà la funcionalitat i les restriccions establides en l'anàlisi de requeriments i l'especificació.

En el nostre cas, les proves de l'aplicació s'han anat repartint al llarg de tota l'etapa d'implementació. És a dir, per a cada operació que s'ha implementat s'han realitzat un conjunt de verificacions i comprovacions i fins que no s'ha satisfet la fase de prova no s'ha passat a la següent operació. Per últim, s'ha provat tot el sistema en conjunt per comprovar el correcte funcionament a nivell global.

Per provar cada operació s'ha revisat el seu contracte i s'han introduït les dades que causaven les excepcions per comprovar que el programa les controlava i que funcionava correctament. També s'ha provat l'operació amb dades correctes per comprovar si el resultat era l'espera't. S'ha controlat que els valors de tornada en les funcions fossin sempre els esperats i en cas d'obtenir un comportament estrany, s'ha modificat la implementació per tal de corregir-lo i s'ha tornat a realitzar la prova. A més a més, s'han realitzat proves amb valors no esperats, nuls i incomplerts.

També s'han fet unes quantes proves amb operacions complexes que implicaven una seqüència de creacions o actualitzacions de registres en diferents taules de la base de dades. Aquestes operacions, que s'han d'executar com si d'una única transacció es tractés, s'han provat a consciència. S'ha comprovat tant la correcta actualització de les dades en cas de que el funcionament fos l'espera't, com la correcta restauració de totes les dades en el cas de que hagués alguna errada a qualsevol punt del procés.

També s'han realitzat diverses proves per provar el funcionament del control d'accés. S'ha intentat accedir a la aplicació introduint diferents URLs conegudes i s'ha comprovat que l'aplicació no permet que es comenci sessió des de una pàgina diferent de la pàgina principal, tornant a adreçar sempre cap aquesta. També s'ha comprovat que un usuari amb un perfil determinat no pugui accedir a pàgines que no són visibles amb el seu perfil.

## 4.6 Pas a producció

El pas a producció és la tasca mitjançant la qual es posa en marxa l'aplicació a l'entorn definitiu. Per tal que aquest procés es realitzi amb èxit s'han de tenir en compte una sèrie de passos.

En primer lloc s'ha de instal·lar una màquina que serà el servidor i s'ha de configurar per tal que tingui accés a Internet.

S'ha de instal·lar un servidor web que sigui compatible amb la plataforma Java. A més a més, s'ha de instal·lar també una base de dades relacional MySQL; aquesta es pot obtenir de forma gratuïta des de la web de MySQL.

La creació de les taules de la base de dades i la introducció d'informació inicial per una correcta posta en marxa, es realitzarà mitjançant una sèrie de scripts de CREATE e INSERTS que s'adjuntaran amb l'aplicació.

L'aplicació con tots els seus components (pàgines .groovy, .js, .html, .css, imatges, fitxers de configuració, controladors, arxius d'assemblatge, etc.) s'hauran de copiar dins d'una carpeta del servidor web.

Aquesta fase, però, no s'ha realitzat en aquest projecte. Principalment perquè no es disposava d'un servidor on fer-ho i perquè degut a la càrrega de treball, s'ha optat per no dedicar temps a configuracions i posada en marxa del servidor ja que això serien tasques de sistemes. Tan sols s'ha configurat l'entorn a nivell local i es té la confiança de que desplegar aquesta aplicació en un servidor no resultaria difícil, segons el que s'ha llegit a la documentació oficial.

# 5 Planificació

## 5.1 Recursos asignats

La planificació té com a objectiu determinar quines seran les activitats a realitzar, fer una estimació del temps que requerirà realitzar cadascuna d'elles i definir els recursos que s'utilitzaran per dur-les a lloc.

En aquest apartat es descriuen els recursos destinats a la realització d'aquest projecte, que es poden classificar en els següents tipus:

- **Recursos Humans:** En aquest projecte disposem d'una sola persona que farà d'analista i programador, per tractar-se del seu projecte final de carrera.
- **Recursos Hardware:** 1 ordinador Pentium-IV SU4100 1,3 GHz.
- **Recursos Software:**

| Software | Utilització |
|---|---|
| Linux | Sistema operatiu |
| Open Office | Documentació del projecte |
| Microsoft Visio 2003 | Diagrames de la especificació i disseny |
| Gimp | Creació d'imatges i botons |
| MySQL | Sistema de gestió de base de dades |
| Grails | Framework de l'aplicació |

A l'estudi econòmic d'aquest apartat es veurà amb detall el cost associat a cadascun d'aquests recursos en el desenvolupament del projecte.

## 5.2 Planificació inicial

Per realitzar la planificació s'han agafat les diferents tasques del model clàssic i les seves subtasques i s'ha estimat l'esforç necessari per cadascuna d'elles.

El projecte és de 37,5 crèdits, per tant, a partir de la estimació de les 20 hores per crèdit que especifica la normativa, partim de la base de que s'han de invertir aproximadament unes 740 hores.

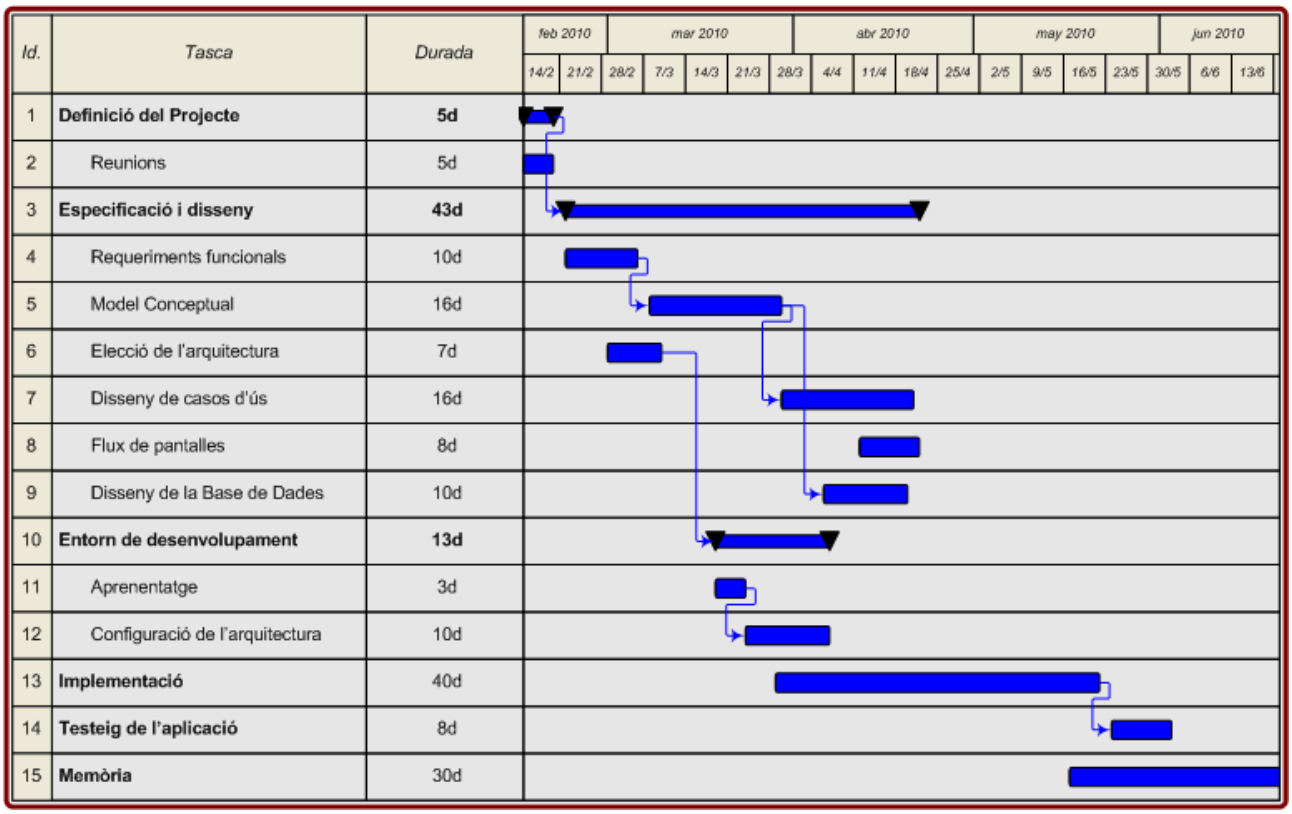
La duració que s'ha estimat per cada tasca del projecte es presenta en la següent taula:

| Tasca | Durada |
|---|---|
| **Estudi previ** | **20 h** |
| **Anàlisis de requeriments** | **80 h** |
| Requeriments funcionals | 60 h |
| Requeriments no funcionals | 20 h |
| **Especificació** | **120 h** |
| Model conceptual | 40 h |
| Model de casos d'us | 40 h |
| Model de comportament | 40 h |
| **Disseny** | **180 h** |
| Decissions de disseny | 40 h |
| Capa de presentació | 60 h |
| Capa de domini | 40 h |
| Capa de gestió de dades | 20 h |
| Base de dades relacional | 20 h |
| **Implementació** | **260 h** |
| Estudi de las opcions tecnològiques | 45 h |
| Instal·lació plataforma desenvolupament | 20 h |
| Aprenentatge de la tecnologia a utilitzar | 55 h |
| Codificació | 120 h |
| Base de dades relacional | 20 h |
| **Proves** | **40 h** |
| **Elaboració memòria** | **40 h** |
| **Total** | **740 h** |

Per definir els dies que s'invertiran en cada tasca, utilitzant l'eina Microsoft Visio, s'han tingut en compte que com a terme mig es treballarà al projecte 4 hores diàries, els dies feiners.

Llavors, es necessitaran 226 dies per acabar el projecte. Si suposem que en un mes hi ha 20 dies feiners, una sola persona fent d'analista i de programador trigaria aproximadament 11 mesos en finalitzar el projecte.

La Figura següent mostra la planificació de les diferents fases de disseny i implementació que es duran a terme en el projecte amb un diagrama de Gantt.




| Id. | Tasca | Durada |
|---|---|---|
| 1 | **Definició del Projecte** | **5d** |
| 2 | Reunions | 5d |
| 3 | **Especificació i disseny** | **43d** |
| 4 | Requeriments funcionals | 10d |
| 5 | Model Conceptual | 16d |
| 6 | Elecció de l'arquitectura | 7d |
| 7 | Disseny de casos d'ús | 16d |
| 8 | Flux de pantalles | 8d |
| 9 | Disseny de la Base de Dades | 10d |
| 10 | **Entorn de desenvolupament** | **13d** |
| 11 | Aprenentatge | 3d |
| 12 | Configuració de l'arquitectura | 10d |
| 13 | **Implementació** | 40d |
| 14 | **Testeig de l'aplicació** | 8d |
| 15 | **Memòria** | 30d |

*Diagrama de Gantt amb la planificació inicial del projecte*

La durada de les tasques s'ha definit tenint en compte la pròpia experiència en la planificació de projectes. Al següent apartat es veurà si ha hagut desviacions respecte a la planificació inicial.

## 5.3 Planificació real

En aquest apartat es fa una revisió de la planificació inicial i es compara amb el temps real que s'ha necessitat per desenvolupar cadascuna de les tasques. S'analitzen els desviaments que s'han produït i es justifiquen els motius.

| Tasca | Durad estimada | Durada real | Diferència |
|---|---|---|---|
| **Estudi previ** | **20 h** | **20 h** | **0 h** |
| **Anàlisis de requisits** | **80 h** | **80 h** | 0 h |
| Requisits funcionals | 60 h | 60 h | 0 h |
| Requisits no funcionals | 20 h | 20 h | 0 h |
| **Especificació** | **120 h** | **150 h** | **30 h** |
| Model conceptual | 40 h | 50 h | 10 h |
| Model de casos d'us | 40 h | 50 h | 10 h |
| Model de comportament | 40 h | 50 h | 10 h |
| **Disseny** | **180 h** | **180 h** | **0 h** |
| Decisions de disseny | 40 h | 40 h | 0 h |
| Capa de presentació | 60 h | 60 h | 0 h |
| Capa de domini | 40 h | 40 h | 0 h |
| Capa de gestió de dades | 20 h | 20 h | 0 h |
| Base de dades relacional | 20 h | 20 h | 0 h |
| **Implementació** | **260 h** | **290 h** | **30 h** |
| Estudio de les opcions tecnològiques | 45 h | 45 h | 0 h |
| Instal·lació plataforma desenvolupament | 20 h | 20 h | 0 h |
| Aprenentatge la tecnologia a utilitzar | 55 h | 65 h | 10 h |
| Codificació | 120 h | 160 h | 40 h |
| Base de dades relacional | 20 h | 20 h | 0 h |
| **Proves** | **40 h** | **40 h** | **0 h** |
| **Elaboració memòria** | **40 h** | **50 h** | **10 h** |
| **Total** | **740 h** | **810 h** | **70 h** |

A continuació es mostren les desviacions, obtingudes en percentatges, entre la planificació inicial i la planificació real.

| Tasca | Desviació |
|---|---|
| **Estudi previ** | **0 %** |
| **Anàlisis de requisits** | **0 %** |
| **Especificació** | **40 %** |
| **Disseny** | **0%** |

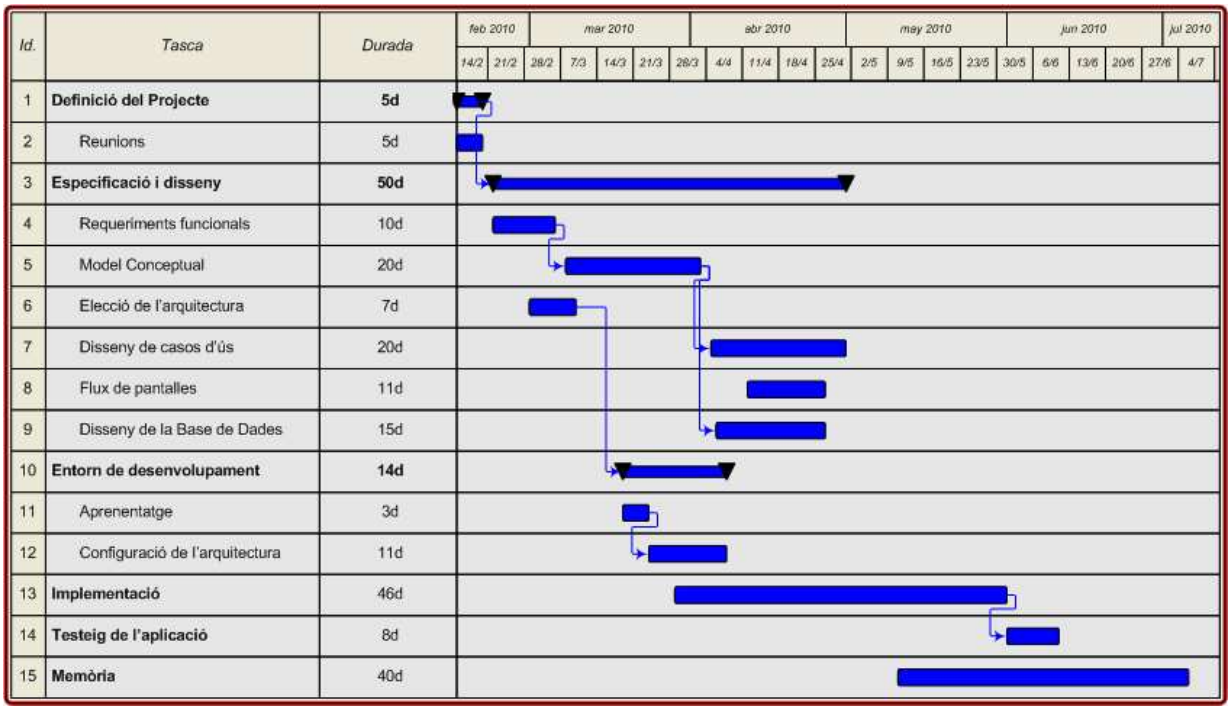
| | |
|---|---|
| **Implementació** | **8,6** |
| **Proves** | **0 %** |
| **Elaboració memòria** | **25 %** |
| **Total** | **10,5 %** |

Com es pot observar ha hagut un total de 70 hores de desviació, que en percentatge ha estat aproximadament un 10,5%.

Excepte en les tasques d'estudi previ, disseny i proves, en les que no trobem diferencia entre la planificació estimada i la real, a la resta de tasques s'han produït desviacions. Les més importants les trobem a les etapes d'especificació, implementació i elaboració de la memòria.

La etapa d'especificació ha estat més llarga del que s'esperava degut bàsicament degut a les contínues revisions del model conceptual per tal que s'adaptés el millor possible i de forma genèrica al context. El fet que la implementació es basa en el model conceptual impedia que es pogués començar a desenvolupar amb tranquil·litat i confiança que la lògica no es modificaria. També va ser una de les racons per les quals es va escollir el framework Grails degut a que s'escau en el conjunt de metodologies àgils. Si bé s'ha trobat que el desenvolupament amb Grails no és tan senzill ni tan ràpid com es creia en un principi.

El diagrama de Gantt amb els canvis actualitzats en la planificació han estat els següents:



| Id. | Tasca | Durada |
|---|---|---|
| 1 | **Definició del Projecte** | **5d** |
| 2 | Reunions | 5d |
| 3 | **Especificació i disseny** | **50d** |
| 4 | Requeriments funcionals | 10d |
| 5 | Model Conceptual | 20d |
| 6 | Elecció de l'arquitectura | 7d |
| 7 | Disseny de casos d'ús | 20d |
| 8 | Flux de pantalles | 11d |
| 9 | Disseny de la Base de Dades | 15d |
| 10 | **Entorn de desenvolupament** | **14d** |
| 11 | Aprenentatge | 3d |
| 12 | Configuració de l'arquitectura | 11d |
| 13 | **Implementació** | 46d |
| 14 | **Testeig de l'aplicació** | 8d |
| 15 | **Memòria** | 40d |



*Diagrama de Gantt amb la planificació real del projecte*

## 5.4 Estudi econòmic

En aquest apartat s'avalua el cost econòmic que ha representat el projecte. Quan ja es coneix el temps real invertit per la realització del projecte es pot realitzar una valoració del cost econòmic que ha representat el desenvolupament.

Aquest estudi econòmic es divideix en dues parts:

- El cost dels recursos humans.
- El cost del hardware utilitzat en el desenvolupament.
- El cost del software utilitzat en el desenvolupament.

### Recursos humans

El càlcul del cost dels recursos humans ens ve determinat pel total d'hores

de cada perfil assignades al projecte. Els perfils que s'han tingut en compte són: analista i programador.

Com hem vist el projecte ha tingut una duració real de 675 hores, que segons el recurs assignat, es divideixen de la següent forma:

| Tasca | Dedicació | Preu/Hora | Cost |
|---|---|---|---|
| Analista | 400 h | 35 € | 14000 € |
| Programador | 410 h | 20 € | 8.200 € |
| **Total** | **810 h** | | **22.200 €** |

## Hardware

En la següent taula s'especifiquen els recursos hardware emprats durant el desenvolupament i el seu cost associat.

| Producte | Cost |
|---|---|
| Pentium-IV SU4100 1,3 GHz | **450 €** |

Per determinar el cost final, s'ha tingut en compte que els equips utilitzats tenen una vida útil de tres anys i per tant, l'equip s'amortitzarà en aquest període de temps. Considerant que la duració del projecte ha estat aproximadament de 6 mesos, assignarem al desenvolupament el 30% del cost total.

Una vegada realitzades aquestes consideracions obtenim els resultats que es mostren a continuació:

| Producte | Cost |
|---|---|
| PC Pentium IV 1800 MHz | **135 €** |

## Software

En la següent taula es detallen els recursos software utilitzats durant la realització del projecte i el seu cost.

| Producte | Cost |
|---|---|
| Linux | 0 € |
| Open Office | 0 € |
| Microsoft Visio 2003 | 475 € |
| Gimp | 0 € |
| MySQL | 0 € |
| Grails | 0 € |
| **Total** | **475 €** |

Aquest cost no més es dona en el desenvolupament del projecte; el cost una vegada el sistema de informació estigui en producció serà només la quota mensual pel servei d'allotjament web (web hosting), ja que el servidor MySQL és gratuït.

Amb aquestes dades ja es pot determinar el cost total real del projecte:

| Concepte | Cost |
|---|---|
| Recursos humans | 22200 € |
| Hardware | 135 € |
| Software | 475 € |
| **Total** | **22810 €** |

# 6 Conclusions

## 6.1 Objetius assolits

Un cop acabat el procés de desenvolupament de l'aplicació GWAL i tenint en compte l'anàlisi de requisits plantejat al principi d'aquesta memòria, podem concloure que el producte final satisfà els objectius inicials que ens vàrem fixar. S'han realitzat les fases d'especificació, disseny, implementació i testeig que caracteritzen un projecte d'aquestes característiques.

Les funcionalitats que s'han implementat es detallen a continuació seguint les que es van fixar a l'inici del projecte:

- En el mòdul de **Gestió d'activitats, temes i especialitats**, s'han implementat totes les funcionalitats de l'aplicació. S'han implementat les funcions per introduir dies festius, períodes docents i temporades. Quan es creen les activitats aquestes es donen d'alta al sistema i es generen les seves sessions previstes d'acord amb el calendari de dies festius. L'aplicació suporta la generació de sessions previstes amb encavallaments i detecta els encavallaments quan es vol confirmar un taller i alguna de les sessions colisiona amb una sessió confirmada d'un altre taller. En aquest moment l'administratiu ha de resoldre de forma manual els encavallaments canviant alguna de les sessions en conflicte d'hora, dia o aula, o bé donant de baixa el taller.

  Es controla adequadament que els usuaris registrats només es puguin inscriure a tallers en "estat obert" o en "seguiment obert", i es fan les comprovacions automàtiques per a canviar d'estat el taller quan supera les dates establertes.

- El mòdul de **Gestió d'usuaris** també s'ha desenvolupat completament, permetent l'accés personalitzat al sistema, mostrant una interfície diferent depenent del rol i bloquejant l'accés a les àrees no permeses per manca de privilegis.

  Inicialment la web té aparença de web estàtica per oferir la informació que vol veure un usuari de tipus lector. S'ha implementant un procés de registre per generar un usuari registrat amb login i contrasenya. S'ha afegit un control "captcha" per tal d'evitar registres de tipus spam introduïts pels robots que fan registres automàticament. Un cop el registre s'ha realitzat correctament l'usuari pot entrar a la web dinàmica omplint el camps d'usuari i contrasenya en l'apartat de login.

Aspectes com la seguretat han estat molt estudiats i meticulosament programats tenint en compte que s'havien de bloquejar intents d'accés a àrees restringides a rols d'administració i que s'havia d'impedir a un usuari registrat realitzar inscripcions d'altres usuaris en les activitats. També s'assegura la no accessibilitat a les dades d'altres usuaris ni la seva modificació.

- El mòdul de **Gestió de fulls d'interès** també té les seves funcionalitats completament implementades, tot i que s'hagués volgut complementar amb una visualització més àmplia de tots els fulls d'interés d'un període docent.

- El mòdul de **Gestió d'inscripcions** també té les seves funcionalitats completament implementades. Els usuaris registrats poden tramitar la seva inscripció als tallers disponibles a través del lloc web. Els gestors poden modificar i donar de baixa les inscripcions així com fer-ne de noves des de l'apartat de gestió d'inscripcions.

- El mòdul de **Gestió de l'assistència** també té les seves funcionalitats completament implementades. Els formadors poden emplenar les sessions que imparteixen introduint l'assistència dels usuaris, indicant si l'usuari es convidat o si està inscrit al curs. Quan s'ha omplert l'assistència a un taller es calcula el nombre d'assistents i s'enmagatzema en la sessió.

- El mòdul de **Gestió de recursos** té les seves funcionalitats cobertes. El personal del centre pot donar d'alta, manipular i esborrar recursos. També es poden realitzar reserves de materials per les sessions.

- El mòdul de **Configuració del sistema** té les seves funcionalitats cobertes.

Aquest mòdul permet adaptar l'aplicació a cada centre lúdic real. Els formularis de creació de tots els objectes del sistema tenen alguns camps per defecte que agafen de la configuració del sistema. L'administrador pot modificar aquests valors per defecte en les pantalles de creació canviant els valor editant la configuració. Com aquest objecte es un singleton, s'ha implementat que la lògica de domini controli que només existeix una instància i que no és pot esborrar. Aquesta instància es crea quan es desplega l'aplicació i en cas de que s'esborri per algun mitjà accidental només cal reiniciar l'aplicació perquè es torni a generar.

L'administrador pot donar d'alta usuaris de qualsevol tipus de rol (formadors, administratius, administradors,...) per a què aquests puguin realitzar les seves tasques de gestió específiques a l'aplicació. S'ha implementant en el desplegament inicial de l'aplicació la creació de **dos** comptes d'usuari **per defecte**, un de tipus administrador per permetre administrar l'aplicació i crear nous rols, i un altre de usuari testejador per tal de poder veure a la web pública els canvis en la configuració que l'administrador introdueix en les preferències default i el template de la web. Es detalla a que ens referim amb la paraula template en el següent paràgraf.

Un dels requisits no funcionals del projecte era l'adaptabilitat a qualsevol tipus d'entitat lúdica (generalització) i per tant calia cobrir aquest requisit en dos aspectes: que fos adaptable per mitjà de les entitats objecte que es poden crear i que tingués una capa de presentació adaptable. Evidentment fer un mòdul d'adaptabilitat en la capa de presentació de l'estil dels CMS (Content Manager System) hagués estat molt encertat però hagués portat suficient càrrega de treball com per ser un altre projecte per si mateix. La forma en la que s'ha adaptat la capa de presentació que es mostra a l'usuari, en aquest cas la web pública, ha estat amb la implementació de la classe ConfiguracioSistema on s'incluen un seguit de camps de text per omplir i que van associats a cadascuna de les pestanyes de la web principal. Ja que s'ha fet un gran esforç en l'anàlisi de requisits per mitjà de l'estudi del funcionament de centres lúdics reals, es pot considerar que la disposició de botons i menús escollida es pot considerar un esquelet "template" adequat i suficient per cobrir les necessitats d'informació d'aquest tipus de centres.

Tal i com s'ha pogut veure al capítol 5 el temps dedicat a l'especificació del sistema ha superat amb escreix la planificació inicial que s'havia fixat retardant les fases depenents d'aquesta. A més cal tenir en compte que s'ha decidit treballar amb una tecnologia desconeguda per l'estudiant i això ha comportat una corva d'aprenentatge amb la que no s'havia comptat inicialment. El motiu pel qual s'ha decidit fer servir el framework Grails ha estat per l'interés de l'estudiant en conèixer una tecnologia àgil d'última generació i tenir la oportunitat de treballar amb una metodologia que comença a ser molt demandada en el nostre mercat de treball. Si bé hi ha hagut detalls de la implementació que no s'han pot realitzar com s'hagués volgut si que es pot considerar que s'ha cobert l'objectiu del projecte de crear un sistema d'informació que oferís una única eina genèrica i adaptable per a cobrir les necessitats de diferents tipus d'usuaris d'un centre lúdic real. Podem estar segurs

que aquesta eina es pot implantar en un centre d'aquestes característiques oferint les funcionalitats principals.

## 6.2 Possibles ampliacions

En la aplicació es poden realitzar algunes ampliacions interessants:

- Afegir un formulari de comunicació per tal de que els usuaris realitzin consultes al centre (fàcil de realitzar).
- Afegir un sistema de agenda i notícies vinculades a les i al centre en general que vagi mostrant cada dia les notícies creades. Comportaria canvis en la lògica del sistema i en les classes de domini.
- Afegir informes per als gestors del centre amb informació útil en forma de taules conjuntes, com per exemple llistats de usuaris i tallers en els que s'han inscrit al llarg del temps (Grails incorpora un plugin que facilita la tasca de generació d'informes)
- Afegir un sistema de pagament segur per poder pagar les inscripcions als tallers.

## 6.3 Conclusió personal

La realització d'aquest projecte ha estat una bona experiència pel següents motius:

M'ha aportat una visió global de tot el procés de creació d'una aplicació des de l'estudi del cas, l'anàlisi de requisits fins al disseny i implementació de l'aplicació. M'ha permès aplicar molts dels coneixements adquirits en les assignatures de la carrera, especialment en aquelles de la branca d'especificació i disseny de software, bases de dades i sistemes de seguretat informàtica.

M'ha permès enfrontar-m'hi amb la dificultat de dividir la feina en tasques i estimar el temps per cadascuna d'elles sense tenir l'experiència de quan temps es triga en realitzar un projecte d'aquest estil.

He desenvolupat el projecte a mitja jornada conjuntament amb la feina de programador que tenia en una empresa privada i això m'ha estat difícil de compaginar.

El fet de que el sistema GWAL es volgués aplicar en una escola de ball real amb usuaris que farien servir el sistema quan estigués acabat ha fet que el nivell d’exigència en especificar i dissenyar l'aplicació hagi estat alt, i la implementació robusta i fiable.

Per últim considero una aportació positiva haver-me format en les tecnologies de Grails, Groovy, GORM, Hibernate, i Spring MVC. Tecnologies que es fan servir a les empreses actualment i que tenen gran demanda.

En definitiva crec que la realització d'aquest projecte ha completat la meva formació d'Enginyer en Informàtica en la Facultad de Informática de Barcelona, permetent-me consolidar i fer servir molts dels coneixements adquirits al llarg dels anys de la carrera.

# 7 Bibliografia

## 7.1 Llibres

"**Enginyeria del software: Especificació".**

Dolors Costal – M.Ribera Sancho-Ernest Teniente.

Edicions UPC, 1999

**"Enginyeria del software: Disseny I".**

Cristina Gómez – Enric Mayol – Antoni Olivé

Edicions UPC, 1999

**"Enginyeria del software: Disseny II".**

Carles Farré – Antoni Olivé i Carme Quer.

Edicions UPC, 1999.

**"Disseny de bases de dades".**

Jaume Sistac i Planas, Albert Abelló Gamazo, Blai Cabré i Segarra, Elena Rodríguez González i Ramon Segret i Sala.

Editorial UOC, 2002

**"Fundamentos bases de datos con Java".**

Kevin Mukhar ... [et al.]

Madrid, Anaya Multimedia, 2001

**"UML y patrones"**

Craig Larman

Editorial Prentice Hall, 1999.

**"Designing Web Usability"**

Jakob Nielsen

New Riders Publishing, 2000

**"Shaping Web Usability"**

Albert N. Badre

Addison Wesley, 2002

**"No me hagas pensar. Una aproximación a la usabilidad en la Web"**

Steve Krug

Pearson Educación, 2000

**"Groovy and Grais. From Novice to Professional"**

Christopher M. Judd, Joseph Faisal Nusairat, James Shingler

Apress, 2008

## 7.2 Web

Departament de Llenguatges i Sistemes informàtics

- http://www.lsi.upc.es/~es-e/

MySQL i PostgreSQL

- http://www.mysql.com
- http://www.microsoft.com/presspass/itanalyst/docs/06-30-09enterprisedatabasemanagementsystems.aspx
- http://dev.mysql.com/doc/
- http://www.postgresql.org/
- http://www.wikivs.com/wiki/MySQL_vs_PostgreSQL#Insert_Ignore_.2F_Replace
- http://blog.taragana.com/index.php/archive/postgresql-vs-mysql-comparative-review/

Usabilitat

- http://www.desarrolloweb.com/articulos/221.php
- http://www.ainda.info/que_es_usabilidad.htm
- http://czsa.enlaces.cl/montegrande/Documentos_MG/charla_diseno/diseno.htm
- http://www.htmlweb.net/usabilidad/usabilidad_1/usabilidad_1_1.html

Grails

- http://www.grails.org/
- http://www.ibm.com/developerworks/java/library/j-grails03118/index.html