Integració i millora del Sistema gestor de Reserves MRBS 1.4.8 amb Moodle 2.2 i Drupal 7.x

Alumne: Isaac Alavedra del Rio
Consultor: David García Solórzano
Responsable de l'assignatura: David García Solórzano

Treball final de Grau de Tecnologies de Telecomunicació
Departament Eines web per l'ensenyament d'enginyeria a distància

ABSTRACT

El projecte es basa en la integració del sistema gestor de reserves MRBS 1.4.8 en una interfície web creada mitjançant Drupal 7.12 i que compta amb Moodle 2.2 per a realitzar les funcionalitats de Campus Virtual. La integració consta de tres fases principals. En la primera fase es realitza el mòdul que permet integrar MRBS i Drupal, sent aquest últim el sistema que s'encarrega del registre, gestió i control de permisos dels usuaris. En la segona fase s'implementa la funcionalitat que permet la integració d'usuaris en les reserves de recursos de la família aula al gestor de reserves. Finalment, en la tercera fase, es realitza la integració d'usuaris entre Drupal i Moodle. La integració permet a un usuari registrat a la interfície de Drupal accedir a Moodle i/o MRBS amb diferents funcionalitats depenent dels permisos dels que disposa. L'usuari pot realitzar reserves d'aules incloent-hi els alumnes desitjats respectant la limitació de la capacitat de l'aula, accedint a MRBS, tant des de Drupal com des de Moodle, sent Drupal la interfície que fa de nexe entre el Campus Virtual i el gestor de reserves.

El proyecto se basa en la integración del sistema gestor de reservas MRBS 1.4.8 en una interfaz web creada mediante Drupal 7.12 y que cuenta con Moodle 2.2 para realizar las funciones de Campus Virtual. La integración cuenta con tres fases principales. En la primera fase se realiza el modulo que permite la integración entre MRBS y Drupal, siendo este ultimo el sistema que realiza el registro, gestion y control de permisos de los usuarios. En la segunda se implementa la funcion que permite la integración de usuarios en las reservas de recursos de la familia aula en el gestor de reservas. Finalmente, en la tercera fase, se realiza la integración de usuarios entre Drupal y Moodle. La integración permite a un usuario registrado en la interfaz de Drupal acceder a Moodle y/o MRBS con diferentes funciones dependiendo de los permisos de los que dispone. El usuario puede realizar reservas de aulas incluyendo los estudiantes deseados respetando la limitación de la capacidad máxima de la aula, accediendo a MRBS, tanto desde Drupal como desde Moodle, siendo Drupal la interfaz nexo entre Campus Virtual y gestor de reservas.

The project is based on the integration of MRBS 1.4.8 booking management system in a web interface created using Drupal 7.12 and with Moodle 2.2 to perform the functions of Virtual Campus. The integration has three main phases. In the first phase the module allows integration between MRBS and Drupal, the latter being the system that performs logging, management and permissions control of the user. The second implements the function that allows the integration of users in the bookings of the room family resources in the reservation manager. Finally, the third phase, performs

the integration of the users between Drupal and Moodle. The integration allows a user registered in the Drupal interface to access Moodle and/or MRBS with different functions depending on the permissions of the user. The user can make reservations of rooms including desired students respecting the limitation of the maximum capacity of the classroom, gaining access to MRBS, from both Drupal and Moodle, being Drupal the interface link between Virtual Campus and reservations manager.

RESUM

El Projecte Final de Grau de Tecnologies de Telecomunicació, dedicat al desenvolupament d'eines web per a l'ensenyament de l'enginyeria a distància, es centrarà en el desenvolupament i la integració d'un sistema gestor de Reserves que permeti controlar, per una banda, la disponibilitat dels recursos d'un campus virtual i, per una altra, l'accés a aquests recursos per part dels usuaris del campus.

Per a dur a terme el sistema gestor de Reserves per al campus virtual, s'integrarà i modelarà el Sistema Gestor de Reserves de Programari Lliure MRBS (Meeting Room Booking System), en la seva versió mes actual 1.4.8, amb el LMS (Learning Management System) open-source més utilitzat a dia d'avui, Moodle, en la seva versió més actual 2.2, i amb un dels CMS (Content Management System) més utilitzats actualment, Drupal, en la seva versió 7.x, que farà la funció d'interfície de captació d'usuaris i de nexe entre el campus virtual i el sistema gestor de reserves.

El projecte constarà amb la integració de Drupal7, Moodle2 i MRBS 1.4.8, sent Drupal7 el sistema de gestió d'usuaris i de rols, des d'on el campus virtual proporcionat per Moodle2 i el sistema gestor de reserves MRBS prendran les dades per a poder comprovar els cursos en que pot participar cada usuari, i crear reserves d'aules i insertar-hi usuaris en concordança amb la capacitat de les mateixes, respectivament.

La realització del projecte permetrà disposar d'un sistema Gestor de Reserves que compleix amb la gran majoria de requisits que es poden requerir per a gestionar les reserves d'aules i/o recursos d'un centre educatiu, a través del seu campus virtual i/o de la interfície de captació i gestió d'usuaris.

A més permetrà posar en mans de la comunitat de cada projecte els mòduls o blocs creats que compleixin els estàndards específics de cada projecte.

Índex de Continguts

Índex de Figures

Índex de taules

Capítol 1: Introducció

1.1 Abast del Treball

El Projecte Final de Grau de Tecnologies de Telecomunicació, dedicat al desenvolupament d'eines web per a l'ensenyament de l'enginyeria a distància, es centrarà en el desenvolupament i la integració d'un sistema gestor de Reserves que permeti controlar, per una banda, la disponibilitat dels recursos d'un campus virtual i, per una altra, l'accés a aquests recursos per part dels usuaris del campus.

Per a dur a terme el sistema gestor de Reserves per al campus virtual, serà necessària la integració i modelatge del Sistema Gestor de Reserves de Programari Lliure MRBS (Meeting Room Booking System), en la seva versió mes actual 1.4.8, amb el LMS (Learning Management System) open-source més utilitzat a dia d'avui, Moodle, en la seva versió més actual 2.2, i amb un dels CMS (Content Management System) més utilitzats actualment, Drupal, en la seva versió 7.x, que farà la funció d'interfície de captació d'usuaris.

Els sistemes gestors de reserves que hi ha disponibles actualment com a blocs contribuïts per part de la comunitat, per a Moodle, tals com Reservation [1], tenen un ús bastant limitat a nivell de funcionalitats, i una aplicació de les reserves no gaire correcta, ja que ens trobem que un mateix ítem es pot reservar mes d'una vegada en un mateix període de temps. També ens trobem amb d'altres problemes com són la impossibilitat de independitzar el gestor de reserves del curs, o la no integració dels alumnes a les reserves en algun dels blocs, que fan que la utilització d'un dels blocs contribuïts que podem trobar al projecte de Moodle per a realitzar les reserves, no sigui una bona elecció.

Per una altra banda, la utilització de Drupal com a CMS per la captació i gestió d'usuaris, a més de millorar el sistema de captació, botiga virtual,... obre la possibilitat a la utilització de mòduls contribuïts de Drupal per a implementar el Gestor de Reserves. La gran majoria de mòduls de Drupal desenvolupats per a gestionar reserves de recursos estan implementats per a ser eficients en les reserves de recursos diaris, tals com una habitació d'un hotel. D'altres mòduls de Drupal que sí

1 http://drupal.org/project/reservation

que permeten la reserva de recursos amb una granularitat d'hores (i fins i tot minuts), tals com Reservations API [2], encara estan en desenvolupament, i continuen sense ser pensats per a permetre la múltiple inserció d'usuaris en la reserva d'un recurs, com seria el cas d'una aula on es necessita la inserció de diversos alumnes al recurs.

És per això que la integració d'un sistema gestor de Reserves independent, és una bona opció a dia d'avui, com a elecció per a gestionar la reserva i participació d'usuaris en una aula d'un campus virtual creat amb Moodle i amb Drupal com a interfície de captació d'usuaris.

L'elecció de MRBS com a gestor de Reserves es deu a què és un dels sistemes gestors distribuïts sota llicència de programari lliure més robustos i amb major recorregut de la comunitat, a més de disposar de blocs o mòduls per a versions anteriors de Moodle i Drupal. El fet de què es distribueixi sota llicència de programari lliure fa que es pugui adaptar a les necessitats del projecte, així com permetre la fàcil integració amb Moodle i Drupal.

El que es comenta en el paràgraf anterior també es vàlid per a Drupal i Moodle, que també es distribueixen sota llicencia de Programari lliure, amb el que, complint amb les 4 llibertats que es deriven de la seva definició, es permet la seva distribució, re-utilització i modificació del codi font per tal de millorar-ne les característiques, per la qual cosa els 3 sistemes seleccionats son vàlids com a base del projecte.

El projecte constarà amb la integració de Drupal7, Moodle2 i MRBS 1.4.8, sent Drupal7 el sistema de gestió d'usuaris i de rols, des d'on el campus virtual proporcionat per Moodle2 i el sistema gestor de reserves MRBS prendran les dades per a poder comprovar els cursos en que pot participar cada usuari, i crear reserves d'aules i insertar-hi usuaris en concordança amb la capacitat de les mateixes, respectivament, tal i com es pot observar de forma esquemàtica en les figures 1, que mostra el procés d'integració en l'autenticació als diversos sistemes que componen el projecte i la figura 2 que mostra el procés de reserva d'aules.

2 http://drupal.org/project/reservations

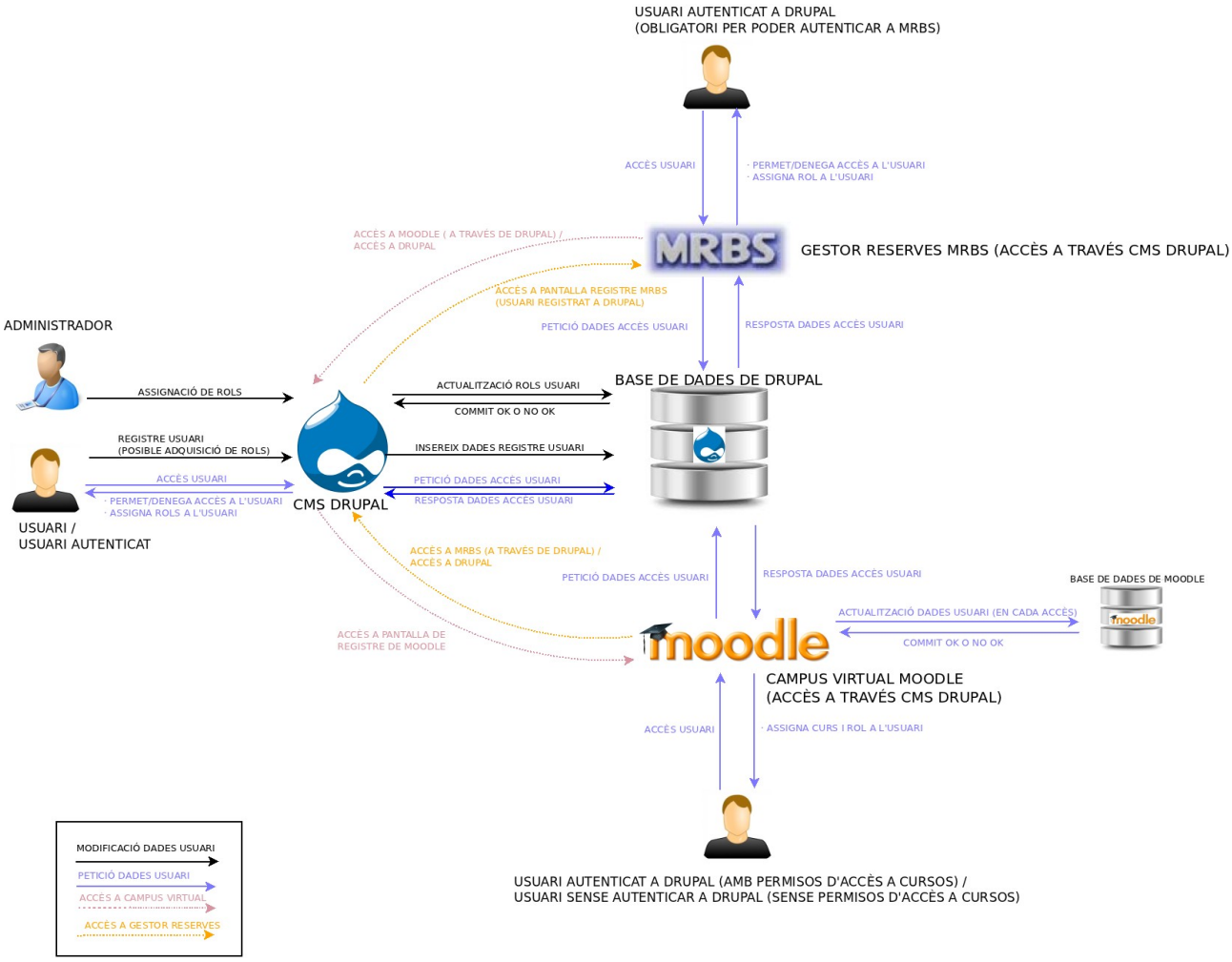

Figura 1: Integració Sistemes (1)

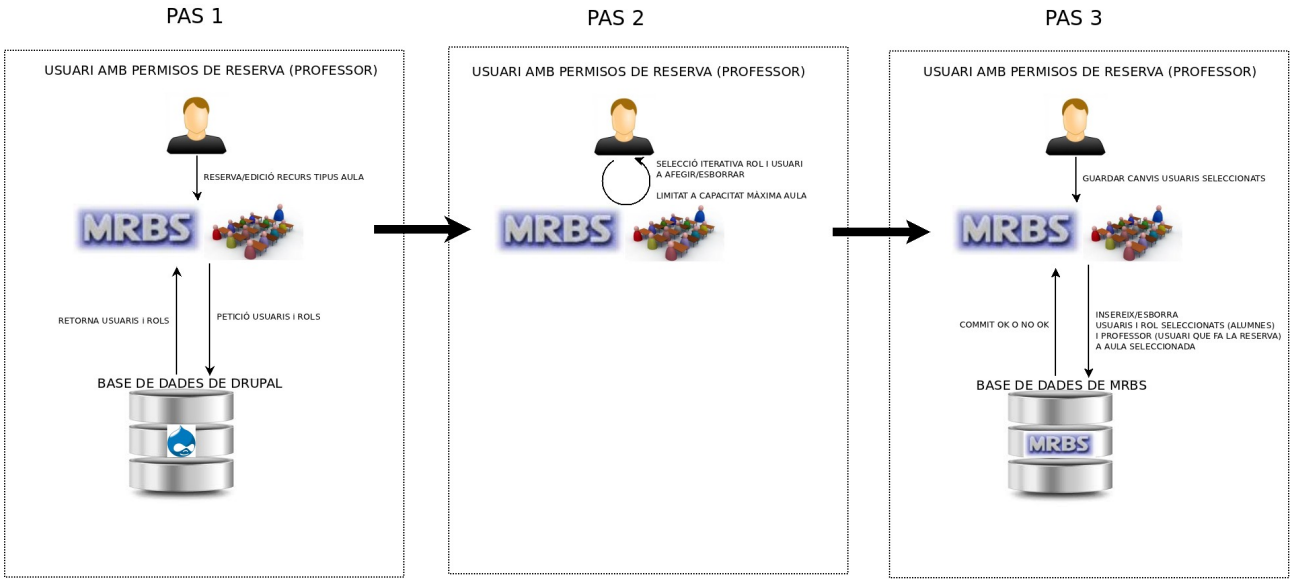

Figura 2: Procés de reserva d'aules (1)

El projecte parteix de la creació d'una cooperativa de formació, la qual compta amb Drupal com a CMS de presentació de continguts de la cooperativa a través d'Internet, així com de Botiga virtual per a la compra de cursos i de diverses funcionalitats requerides per la mateixa. Drupal serà la interfície d'accés de tots els alumnes de la cooperativa a través d'Internet, amb la qual cosa, s'utilitzarà com a sistema d'accés i de nexe entre MRBS i Moodle (també tenint en compte els avantatges funcionals i tècnics d'aquest disseny esmentats en els anteriors paràgrafs). La cooperativa compta amb Moodle com a LMS per a implementar el Campus Virtual i amb MRBS com a sistema gestor de reserves amb la qual cosa una bona integració és un punt essencial per al correcte funcionament de la mateixa.

La realització del projecte, a més, permetrà posar en mans de la comunitat de cada projecte els mòduls o blocs creats que compleixin els estàndards específics de cada projecte.

Paraules clau (Keywords): gestor reserves, Drupal, Moodle, MRBS, programari lliure, integració, mòdul.

1.2 Objectius principals

L'objectiu de la realització del projecte d'integració i millora del Sistema Gestor de Reserves MRBS per a Moodle i Drupal, és el de disposar d'un sistema que permeti controlar els recursos disponibles d'una escola, principalment aules, ja siguin presencials o virtuals, d'una forma eficient i amb la màxima flexibilitat possible.

Per a dur a terme la integració i modelatge del sistema gestor s'hauran d'acomplir les següents fites/objectius:

- Permetre registre únic entre Drupal i MRBS. Donat que la via d'entrada d'usuaris serà a través de la interfície de Drupal, s'haurà de desenvolupar un mòdul que permeti Registre únic entre Drupal i MRBS, de manera que un usuari es pugui moure entre els dos sistemes amb un únic registre, i que el control d'usuaris i rols es centralitzi a través de la interfície de Drupal.

- Modelar el sistema Gestor de Reserves per a què permeti que dins d'un recurs reservable o aula es puguin incorporar els usuaris disponibles a la llista d'usuaris de la base de dades que tinguin un rol determinat, fins al màxim permès per a la capacitat del recurs que s'està reservant.

- Permetre registre únic entre Drupal i Moodle. Donat que la via d'entrada d'usuaris serà a través de la interfície de Drupal, s'haurà de desenvolupar o integrar un mòdul que permeti Registre únic entre Drupal i Moodle, de manera que un usuari es pugui moure entre els dos sistemes amb un únic registre, i que el control d'usuaris i accés als cursos es centralitzi a través de la interfície de Drupal.

- Permetre definir els rols que, tant des de Drupal com des de Moodle permetran accedir als sistema gestor de reserves com a administradors, editors o visors del gestor de reserves.

- Crear un bloc per accedir al sistema Gestor de Reserves (passant per la interfície de Drupal), des del campus virtual de Moodle.

- Crear un bloc per accedir al sistema Gestor de Reserves des de la interfície de captació d'usuaris de Drupal, disponible per als usuaris que tinguin permisos per accedir al sistema Gestor de Reserves.

1.3 Beneficis

La realització del projecte permetrà disposar d'un sistema Gestor de Reserves que compleix amb la gran majoria de requisits que es poden necessitar per a gestionar les reserves d'aules i/o recursos d'un centre educatiu, a través del seu campus virtual i/o de la interfície de captació d'usuaris.

A més, si el desenvolupament del codi necessari per a acomplir els requisits del projecte compleix els estàndards de desenvolupament de cada projecte (Drupal, Moodle i MRBS) es podrà posar en mans de la comunitat, amb l'objectiu de que es disposi d'una alternativa en els sistemes gestors de Reserva i que podrà ser millorat per qualsevol altre usuari. Tot i això els canvis implementats sobre MRBS seran difícilment distribuïts, ja que aquest projecte no utilitza mòduls, sinó que permet els canvis sobre el propi nucli per a modelar el sistema segons els requeriments de cada usuari.

1.4 Motivació

Tal i com s'ha comentat en l'apartat de l'abast del projecte, la realització d'aquest TFG s'engloba com a una petita part, però necessària, de la creació d'una cooperativa de formació, en la qual l'estudiant que realitza el projecte en forma part.

La finalitat és la de posar eines TIC a les mans dels estudiants per a millorar l'experiència de l'aprenentatge.

Amb la creació del projecte es podrà millorar l'experiència d'utilització de Moodle i Drupal envers el gestor de reserves MRBS, tant per part de l'alumne que crea el

projecte, com de les comunitats de programari lliure, sempre que el codi generat segueixi els estàndards de cada projecte.

Capítol 2: Estat de l'art

2.1 Sistemes CMS

Els sistemes de gestió de continguts (en Anglès, Content Management System, abreviat CMS) són programes que permeten crear una estructura de suport (frameWork) per la creació i administració de continguts, principalment en pàgines web, per part dels administradors, editors, i participants.

Consisteixen en una interfície que controla una o vàries bases de dades on s'allotgen els continguts i el disseny. El sistema sempre funciona al servidor web en el qual està allotjat.

Trobem diverses classificacions de Sistemes CMS:

1. Segons característiques.
2. Segons llicència.
3. Segons el llenguatge de programació utilitzat
4. Segons el seu ús i funcionalitats.

En aquest projecte ens centrarem en els CMS distribuïts sota llicència de programari lliure, que són els que actualment copsen la major part del mercat. Dins dels sistemes CMS de programari lliure ens centrarem en els de caràcter general i els d'entorn educatiu.

Analitzant des del punt de vista de les funcionalitats, els sistemes de gestió de continguts de programari lliure que trobem avui en dia al mercat són múltiples i, tot i que els mòduls creats per tercers permeten incorporar funcionalitats extres, els CMS es poden categoritzar segons la funcionalitat original per la qual van ser creats. Les funcionalitats més importants són:

- Blogs: Són els que s'han desenvolupat originalment per a gestionar el contingut de pàgines personals o blogs. En són exemple b2evolution, Movable Type,

Nucleus CMS, Simple PHP Blog, Textpattern o Wordpress, que és un dels més utilitzats dins la seva categoria.

- Fòrums: Són els que s'han desenvolupat originalment per a gestionar el contingut de pàgines on s'escriuen missatges, com poden ser: preguntes, respostes, opinions, reflexions, etc.. En són exemple miniBB, MyBB, phpBB, punBB, Simple Machines Forum.
- Galeries d'Imatges: Són els que s'han desenvolupat originalment per a gestionar pàgines on el contingut principal són imatges. En són exemple Coppermine, Gallery i Gallery 2.
- Wikis: Són els que s'han desenvolupat originalment per a gestionar pàgines on es realitza contingut col·laboratiu. En són exemple Dokuwiki, MediaWiki, TiddlyWiki, WikkaWiki i Pmwiki.
- Entorns Educatius: Són els que s'han desenvolupat originalment per a gestionar pàgines de continguts d'ensenyament online. En són exemple Blackboard Basic, Mahara, Claroline, Dokeos, JoomlaLMS i Moodle, que és el més utilitzat dins la seva categoria.

  Aquests CMS, tenen un nou terme: LMS. També es poden trobar com CMS, però on C no es refereix a "Content", sinó a "Course". Tanmateix, també es poden trobar com LCMS, on la C pot ser tant "Course" com "Content". Fins i tot, alguns parlen de TEL (Technology Enhanced Learning) i VLE (Virtual Learning Enviroment.
- Portals Web o Caràcter general: Són els que s'han desenvolupat originalment per a gestionar el contingut de pàgines de propòsit general i que a través dels mòduls contribuïts poden incorporar funcionalitats de la majoria de les categories anteriors. En són exemple Apache Lenya, Jaws, Joomla, Envolution, myphpnuke, PHP-Nuke, TiddlyWiki, Xaraya, xWiki, Cletu, Drupal i Joomla. Els dos últims són els més utilitzats en la seva categoria.

La majoria de Sistemes CMS de programari lliure es desenvolupen per a operar correctament amb el paquet de programari LAMP, format per Linux com a sistema operatiu del servidor, Apache com a servidor web, MySQL com a gestor de Bases de Dades (tot i que l'adquisició del programa per part d'una empresa que utilitza programari propietari ha fet que la majoria de CMS també operin correctament amb

d'altres bases de dades de programari lliure, tals com PostgreSql o MariaDB) i PhP, Perl o Python com a llenguatges de programació.

Aquest fet fa que l'elecció de LAMP per a ser utilitzat en un CMS de programari lliure sigui la més adequada.

Tal i com s'ha comentat en el punt anterior, en aquest projecte ens centrarem en sistemes CMS de caràcter general i d'entorn educatiu.

2.1.1 CMS de caràcter general basats en programari lliure. Comparativa

Drupal és un dels 3 CMS més utilitzats i populars a dia d'avui com a CMS de programari lliure a Internet. Els seus grans competidors són Joomla! i WordPress. La gran comunitat d'usuaris que utilitzen, mantenen i milloren aquests 3 projectes fa que l'elecció de qualsevol dels 3 sistemes com a CMS de caràcter general sigui adequada.

Tal i com es pot comprovar a la figura 3, extreta de DeviousMedia (http://news.deviousmedia.com/which-open-source-backend- platform-suits-your), WordPress és el CMS més utilitzat, segurament per la seva facilitat d'us, seguit de Joomla! i de Drupal a una gran distància.

Tal i com s'ha comentat anteriorment, qualsevol dels tres CMS presentats a la comparativa es pot considerar un bon CMS. Els tres disposen de mòduls que permeten acomplir la gran majoria de funcionalitats necessàries per qualsevol gestor de continguts web, així com de multitud de temes que permeten personalitzar la presentació del portal. Els tres projectes utilitzen una API pròpia que permet facilitar la creació de nous mòduls, així com d'una documentació molt extensa, que facilita la integració de nous usuaris.

D'aquesta manera, podríem considerar que l'elecció del CMS vindrà donada per les característiques de l'usuari que s'encarregarà de realitzar el desenvolupament del portal web, sent Drupal el més adequat per els usuaris més avançats, i Wordpress el més adequat per a usuaris menys avançats. En el cas d'aquest projecte l'elecció del Sistema CMS és la de Drupal ja que l'usuari que el porta a terme té experiència amb

aquest sistema, i tot i ser el més complex, disposa de majors funcionalitats que la resta de CMS, i la seva complexitat tampoc és molt gran. A més, disposa d'API's de programació que facilitaran la creació de nous mòduls necessaris per a interaccionar amb la resta de sistemes necessaris, tot i que com s'ha comentat anteriorment qualsevol dels tres sistemes hagués estat vàlid.

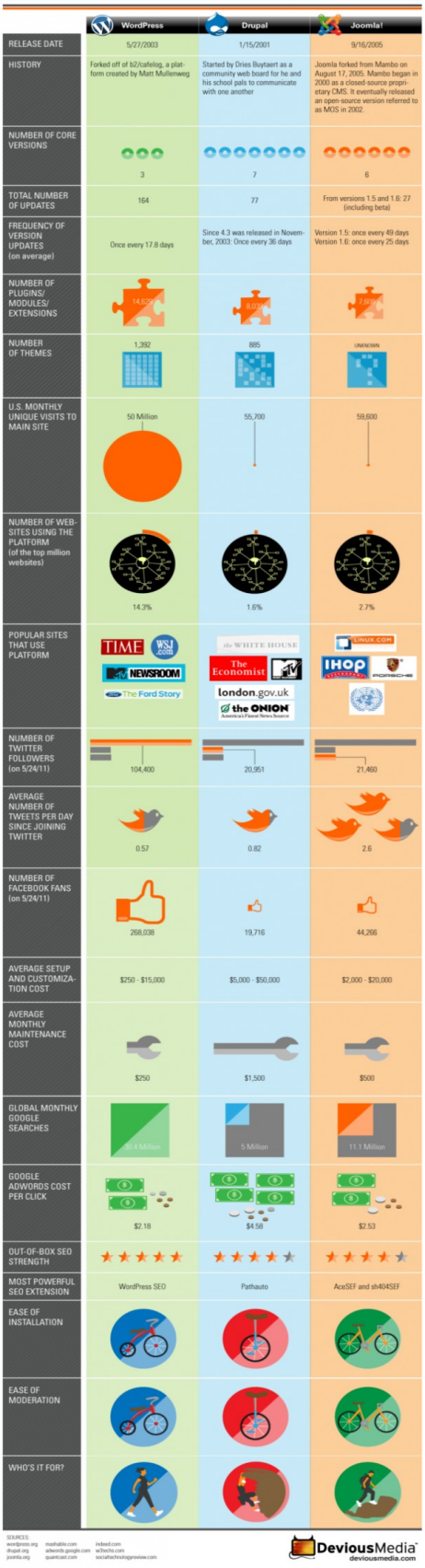

Figura 3: Comparativa CMS Programari Lliure de Caràcter General.

2.1.1.1 Drupal com a Sistema CMS de gestió de Portals Web

Drupal és una plataforma i sistema de gestió de continguts basat en programari lliure que utilitza PHP com a llenguatge de programació i que disposa d'una estructura optimitzada per a ser utilitzat en un servidor LAMP.

Drupal té una capa bàsica, o nucli, que incorpora les funcionalitats bàsiques del sistema gestor de continguts i que s'encarrega de donar estabilitat i seguretat al sistema. El nucli també dóna suport a mòduls que permeten comportaments addicionals. Els mòduls disponibles per a Drupal proporcionen un assortiment ampli de noves característiques, incloent-hi sistemes de comerç electrònic (tals com el mòdul d'Ubercart[3] o Drupal Commerce[4]), flux de treball, galeries d'imatges, o taxonomia/classificació de Drupal (permet que qualsevol contingut sigui classificat amb un sistema d'etiquetat flexible), entre molts d'altres.

El nucli de Drupal (i la majoria dels mòduls contribuïts) disposa de diversos grups de desenvolupament que van incorporant noves funcionalitats a mesura que es van iniciant noves versions. A la data de la redacció d'aquest projecte, la versió estable recomanada per a utilitzar és la 7.12, tot i que se segueix mantenint i desenvolupant la versió del nucli 6.x. Finalment, comentar que ja s'està treballant en el desenvolupament de la versió 8.x

Figura 4: Logo de Drupal.

3 http://drupal.org/project/ubercart
4 http://drupal.org/project/commerce

2.1.2 CMS de caràcter educatiu (LMS) basats en Programari Lliure. Comparativa

Els sistemes CMS de caràcter educatiu, anomenats LMS (Learning Management System), permeten gestionar pàgines de continguts d'ensenyament online.

Moodle és l'LMS més utilitzat i és el projecte que compte amb la major comunitat de suport i desenvolupament a nivell mundial a la data de la redacció d'aquest projecte. El projecte compte amb una comunitats regionals (tals com l'espanyola i la catalana) on es poden copsar amb major detall les necessitats especifiques de cada regió. La majoria d'universitats, centres educatius i acadèmics de l'estat espanyol utilitzen Moodle com a plataforma LMS per al seu campus virtual, com per exemple la UAB [5], la UPC [6] o la UCM [7] entre moltes d'altres.

La gran comunitat de desenvolupadors que hi ha en el projecte Moodle fa que les funcionalitats base del nucli siguin millorades constantment, cosa que en garanteix la seva estabilitat, i que incorpori noves funcionalitats, en forma de blocs, que cobreixen la gran majoria de necessitats de l'aprenentatge en entorns virtuals.

Aquests factors fan que Moodle sigui una de les millors eleccions per a ser usat com a sistema LMS.

El mercat de LMS disposa de diversos projectes alternatius a Moodle, tant de programari propietari com de programari lliure, i que igualment disposen de les funcionalitats necessàries per a poder implementar un bon sistema LMS. Una de les alternatives es BlackBoard, que està basat en programari propietari, i que juntament amb Moodle es un dels LMS més utilitzats.

El fet de què Moodle sigui un projecte distribuït sota llicència de programari lliure, és una de les principals avantatges respecte BlackBoard, ja que a part de no tenir costos de llicència, disposa d'una gran comunitat de desenvolupadors, usuaris i experts en pedagogia, especialment activa per tractar-se d'un entorn educatiu, que permeten

5 http://moo.uab.cat/
6 https://atenea.upc.edu/Moodle/login/index.php
7 https://cv2.sim.ucm.es/Moodle/login/index.php

incorporar ràpidament qualsevol necessitat detectada en els sistemes de gestió de continguts de caràcter educatiu.

En diversos articles s'ha realitzat una comparativa detallada entre les diverses funcionalitats de Blackboard i Moodle. Dos exemples en son [1] i [2].

2.1.2.1 Moodle com a Sistema LMS de gestió de Portals Web en entorns educatius

Tal i com s'ha comentat en el punt anterior, Moodle és un sistema LMS (Learning Management System) de programari lliure. Moodle utilitza PHP com a llenguatge de programació i disposa d'una estructura optimitzada per a ser utilitzat en un servidor LAMP.

El disseny i desenvolupament de Moodle es basen en la idea de la pedagogia constructivista social, tal i com declara Martin Dougiamas [3], creador de Moodle. Aquesta filosofia radica en la cooperació en grup i en la idea que el coneixement es pot crear o construir activament per l'estudiant.

A la data de la redacció d'aquest projecte, Moodle compta amb més de 66.000 llocs web de més de 200 països que utilitzen el LMS[8].

Moodle és la primera comunitat de Programari lliure no formada íntegrament per a informàtics, sinó que també hi participen pedagogs i gent relacionada amb l'ensenyament[9].

De la mateixa manera que la majoria dels CMS de programari lliure, Moodle disposa d'un nucli que incorpora les funcionalitats base del sistema, s'encarrega de la seguretat i controla els diversos blocs contribuïts que es realitzen per part de la comunitat i que permeten millorar o personalitzar les característiques i el disseny del sistema.

8 http://Moodle.org/stats/
9 http://ca.wikipedia.org/wiki/Moodle

Els principals blocs per defecte que venen amb el nucli de Moodle són:

- Bloc de tasques: Permet enviar i qualificar les tasques realitzades pels estudiants.
- Bloc de consulta: Permet realitzar votacions sobre temes proposats.
- Bloc de fòrum: Permet crear fòrums on els usuaris poden participar enviant comentaris sobre temes proposats.
- Bloc diari: Permet enviar informació privada entre estudiant i professor.
- Bloc qüestionari: Permet crear bases de dades de preguntes que es qualifiquen automàticament.
- Bloc recurs: Permet presentar contingut digital (Word, PowerPoint, Flash,..)
- Bloc enquesta: Permet realitzar enquestes sobre les classes realitzades.
- Bloc wiki: Permet que els usuaris treballin en grup sobre un mateix document.

Figura 5: Logo de Moodle.

2.2 Sistemes Gestors de Reserves

Els sistemes gestors de reserves són aplicacions web o locals, que mitjançant una interficie gràfica permeten realitzar reserves de sales, habitacions o d'altres recursos.

Les característiques habituals que presenta un sistema gestor de reserves són:

- Basats en Web/Intranet. Permeten connexió des de qualsevol cercador de qualsevol component.
- Simplicitat d'administració i gestió.
- Autenticació d'usuaris amb diversos permisos.

- Resolució de conflictes en les reserves.
- Disponibilitat de reserves iteratives.
- Vistes seleccionables per dia/mes/any.
- Integració amb els principals sistemes gestors de bases de dades.
- Multi-llenguatge.

En aquest projecte ens centrarem en els sistemes gestors de reserves distribuïts sota llicència de programari lliure, que són els que actualment copsen la major part del mercat. Dins dels sistemes gestors de reserves de programari lliure ens centrarem en els que permeten integració amb Moodle i amb Drupal, que són el CMS i LMS seleccionats per a dur a terme el projecte.

2.2.1 Sistemes Gestors de Reserves per a Moodle 2.x

Una de les funcionalitats que pot requerir un centre de formació, és la de gestionar la reserva de recursos, principalment les aules, ja siguin virtuals o presencials.

Aquest fet fa necessària la disponibilitat d'un sistema gestor de reserves que centralitzi les reserves, tant de les aules virtuals, com de les aules presencials, aixi com d'altres elements o materials dels quals disposi el centre de formació (com poden ser materials esportius, projectors,...).

En el cas de Moodle, trobem dues maneres de crear o utilitzar sistemes gestors de reserves:

- Sistemes gestors de reserves creats ad hoc per a Moodle.
- Sistemes gestors de reserves existents integrats al framework de Moodle.

2.2.1.1 Sistemes gestors de reserves creats ad hoc per a Moodle

A la data de redacció d'aquest projecte trobem un bloc gestor de reserves per a integrar al sistema base de Moodle 2.x, en forma de bloc, el bloc Reservation (veure

figura 6). Aquest mòdul es pot trobar a la pagina de blocs de Moodle i té les següents característiques:

- Nom: Reservation
- Categoria: Activitats.
- Versions per a descarrega: 2.0 i 2.1 (funciona correctament amb el nucli 2.2).
- Lloc del projecte: http://Moodle.org/plugins/view.php?plugin=mod_reservation
- Responsable: Roberto Pinna.
- Descripció: L'objectiu del bloc Reservation és la de programar sessions de Laboratori i exàmens. També permet realitzar reserves de diferents recursos.
  El professor pot definir el nombre màxim d'alumnes per reserva, la data d'obertura de l'event, i la data de tancament entre d'altres.
- Avantatges del bloc per la integració al nucli de Moodle 2.2:
  - S'integra correctament al nucli de Moodle 2.2.
  - Integra els usuaris de la base de dades d'usuaris de Moodle.
  - Permet definir un nombre màxim d'alumnes per recurs o aula.
  - Permet definir un temps de disponibilitat del recurs o aula.
- Desavantatges del bloc per la integració al nucli de Moodle 2.2 / raons per les quals es requereix la utilització d'un altre mètode gestor de recursos:
  - Un recurs es pot reservar més d'una vegada, en un mateix espai temporal.
  - Els recursos es defineixen com a camps de text, no com a elements o objectes definits en la base de dades del sistema.
  - Cada reserva està associada a un curs. No es pot accedir a un recurs reservat en un curs diferent en el que ha estat definit, el que impossibilita declarar aules de forma global.

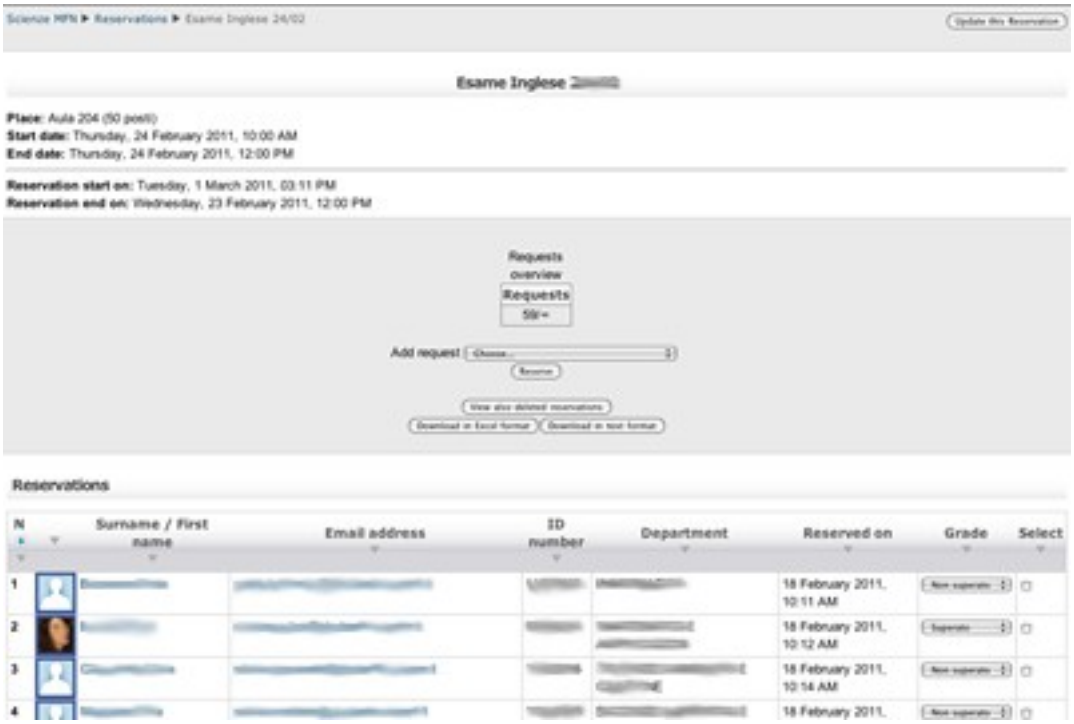

Figura 6: Bloc Reservation per Moodle

2.2.1.2 Sistemes gestors de reserves existents integrats al framework de Moodle

A la data de redacció d'aquest projecte trobem un bloc que integra un sistema gestor de reserves extern amb Moodle 2.2. Aquest és el bloc MRBS Integration (veure Figura 7), que integra el Sistema gestor de Reserves MRBS (Meeting Room Booking System), dins de Moodle en forma de bloc. Aquest mòdul el podem trobar a la categoria de *Old mòdules and plugins* del projecte i té les següents característiques:

- Nom: MRBS
- Categoria: Altres.
- Versions per a descarrega: 1.8, 1.9 i 2.0 (funciona correctament amb el nucli 2.2).
- Lloc del projecte: http://Moodle.org/mod/data/view.php?d=13&rid=734
- Responsable: Anthony Borrow.

- Descripció: El bloc MRBS integra el projecte MRBS utilitzant les característiques definides al fitxer config.php de Moodle per autenticar contra el fitxer mdl_user com a base de dades Externa i permetre autenticació única i compartició de Rols entre Moodle i RMBS.
- Avantatges del bloc per la integració al nucli de Moodle 2.2:
  - S'integra correctament al nucli de Moodle 2.2.
  - Integra els usuaris i rols de la base de dades d'usuaris de Moodle (fent autenticació en base de dades externa).
  - Realitza la integració de MRBS en mode d'Iframe a Moodle.
  - Permet definir un nombre màxim d'alumnes per recurs o aula.
  - Permet definir un temps de disponibilitat del recurs o aula.
  - Permet realitzar reserves recursives.
  - SSO entre Moodle i MRBS.
- Desavantatges del bloc per la integració al nucli de Moodle 2.2 / raons per les quals es requereix la utilització d'un altre mètode gestor de recursos:
  - Un recurs només pot ser reservat per un usuari (no es poden assignar alumnes a una aula).
  - La versió 2.0 te diversos problemes de seguretat i es distribueix en mode *development*.
  - Tal i com succeïa amb Reservation, cada reserva està associada a un curs. No es pot accedir a un recurs reservat en un curs diferent en el que ha estat definit.
  - Es requereix un altre bloc per a integrar MRBS i Moodle amb d'altres CMS (tals com Drupal).

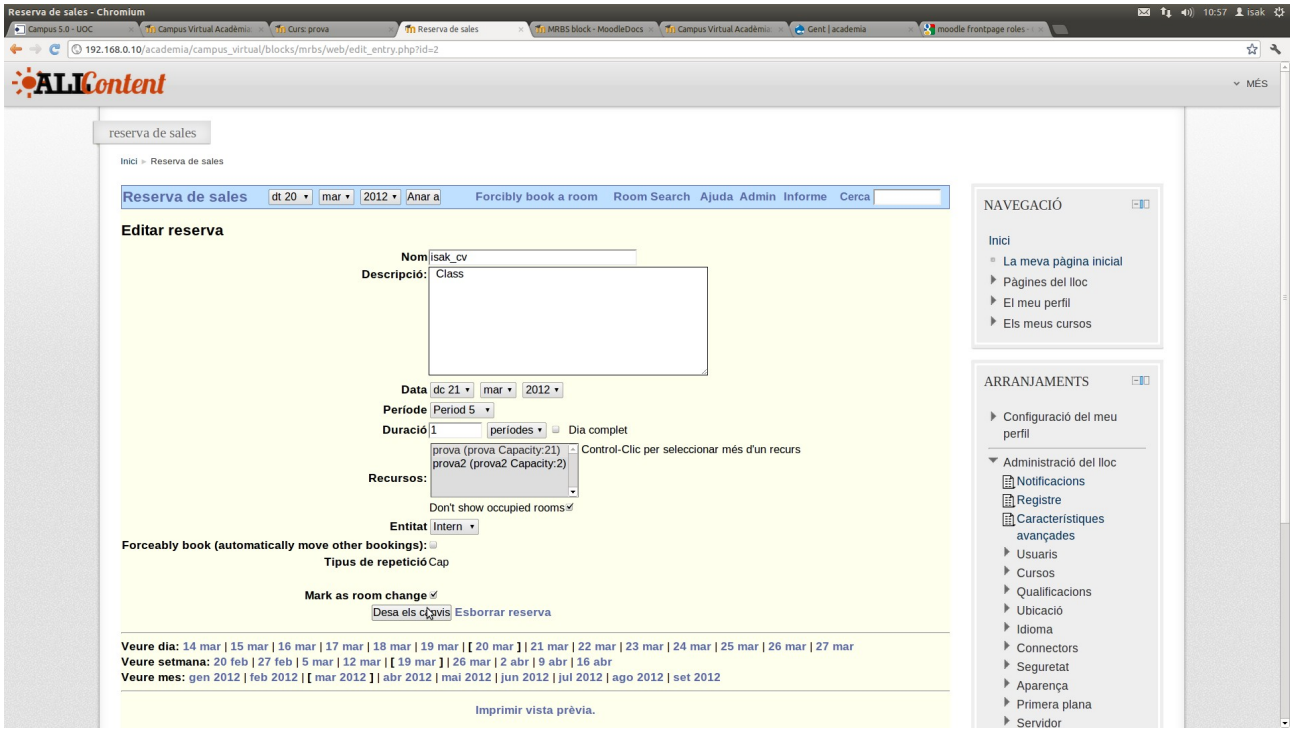

Figura 7: Bloc MRBS Integration per Moodle

2.2.2 Sistemes gestors de reserves per a Drupal 7

Una alternativa per a gestionar les reserves de recursos en centres de formació que disposen d'un CMS no orientat a educació, com potser Drupal, és utilitzar el CMS per a realitzar les reserves de les sales i recursos, deixant les funcions educatives per a Moodle i funcions de captació de clients, gestió de reserves, per al sistema CMS no orientat a educació.

En el cas de Drupal, trobem dues maneres de crear o utilitzar sistemes gestors de reserves:

- Sistemes gestors de reserves creats ad hoc per a Drupal.
- Sistemes gestors de reserves existents integrats al framework de Drupal.

2.2.2.1 Sistemes gestors de reserves creats ad hoc per a Drupal

A la data de redacció d'aquest projecte trobem diversos Mòduls gestors de reserves de recursos en fraccions d'hora per a integrar al sistema base de Drupal 7 (també en trobem d'altres que serveixen per realitzar reserves en fraccions de dia, destinats a hotels que no s'adeqüen als requeriments del projecte). Aquests mòduls es poden trobar a la pàgina de *mòduls* de Drupal. Els principals són Reservations API i Simple Reservation.

El Mòdul Reservations Api (del qual encara no es pot prendre captura de pantalla per no estar finalitzat el desenvolupament de la versió 7.x), té les següents característiques:

- Nom: Reservations API
- Categoria: Community, E-commerce.
- Versions per a descarrega: 7.x i 6.x
- Lloc del projecte: http://Drupal.org/project/reservations
- Responsables: kreynen, Dane Powell i caktux.
- Descripció: En la seva versió 7 permet reservar qualsevol tipus de contingut basat en una gran varietat de característiques. Disposa d'un procés de Reserva, confirmació, checkout, i checkin integrats.
- Avantatges del mòdul per la integració al nucli de Drupal 7:
  - Integra els usuaris i rols de la base de dades d'usuaris de Drupal.
  - Permet realitzar reserves en fraccions horàries.
  - Permet la integració amb d'altres mòduls que en faciliten i amplien les característiques.
  - Permet reservar qualsevol tipus de contingut com a entitat.
- Desavantatges del mòdul per la integració al nucli de Drupal 7 / raons per les quals es requereix la utilització d'un altre mètode gestor de recursos:
  - No permet definir un temps de disponibilitat del recurs o aula.
  - No permet realitzar reserves recursives, és a dir no es pot realitzar una reserva d'una sala en una franja horària per a més d'un dia de forma automàtica.

- No permet definir un nombre màxim d'usuaris per recurs.
- La versió 7 no està finalitzada, es distribueix en mode *development* i no és del tot operativa.
- Tot i permetre reserves en franges parcials d'hores està enfocat a reserves d'elements amb cost associat (com per exemple habitacions d'hotel,...)

El Mòdul Simple Reservations (veure Figura 8), té les següents característiques:

- Nom: Simple Reservation
- Categoria: n/d.
- Versions per a descarrega: 7.x i 6.x
- Lloc del projecte: http://Drupal.org/project/simple_reservation
- Mantenidor: Jochen Wendebaum i salvis.
- Descripció: El mòdul proveeix una manera fàcil de realitzar reserves d'ítems que poden ser creats per l'administrador. Permet indicar el nombre màxim de reserves que pot fer un usuari.
- Avantatges del mòdul per la integració al nucli de Drupal 7:
  - Integra els usuaris i rols de la base de dades d'usuaris de Drupal.
  - Permet definir un temps de disponibilitat del recurs o aula.
  - Permet crear contingut reservable únicament una vegada per cada franja horària.
- Desavantatges del mòdul per la integració al nucli de Drupal 7 / raons per les quals es requereix la utilització d'un altre mètode gestor de recursos:
  - La versió 7 no està finalitzada, es distribueix en mode *development* i no es del tot operativa.
  - Interfície de reserva senzilla i amb poques funcionalitats.
  - No permet definir un nombre màxim d'alumnes per recurs o aula.
  - No permet realitzar reserves recursives, és a dir no es pot realitzar una reserva d'una sala en una franja horària per a més d'un dia de forma automàtica.
  - Un recurs només pot ser reservat per un usuari (no es poden assignar alumnes a una aula).

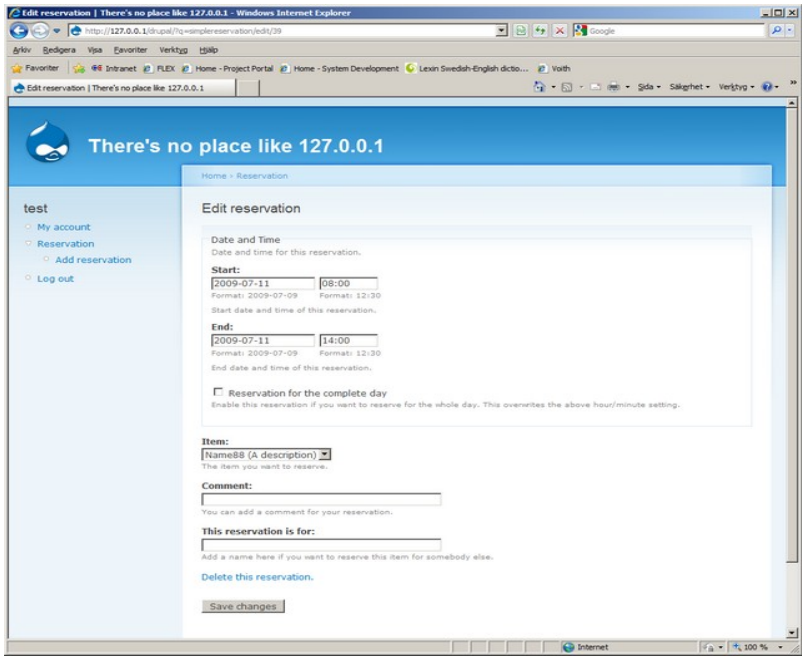

Figura 8: Bloc Simple Reservation per Drupal

2.2.2.2 Sistemes Gestors de reserves existents integrats al framework de Drupal

A la data de redacció d'aquest projecte trobem un mòdul que integra un sistema gestor de Reserves Extern amb Drupal. Aquest mòdul és l'MRBS (veure Figura 9), i integra Drupal (en la seva versió 6) amb el Sistema gestor de Reserves MRBS (Meeting Room Booking System). El mòdul el podem trobar a la pàgina de *mòduls* de Drupal i té les següents característiques:

- Nom: MRBS
- Categoria: Third-party Integration, Utility.
- Versions per a descarrega: 6.
- Lloc del projecte: http://Drupal.org/project/MRBS
- Mantenidor: wesku.

- Descripció:mòdul que ofereix Single Sign-On entre MRBS i Drupal, utilitzant els comptes, rols i sessions de Drupal a MRBS .
- Avantatges del mòdul per la integració al nucli de Drupal 7:
  - Integra els usuaris i rols de la base de dades d'usuaris de Drupal (fent autenticació en base de dades externa).
  - Realitza la integració de MRBS amb Drupal.
  - Permet definir un nombre màxim d'alumnes per recurs o aula.
  - Permet definir un temps de disponibilitat del recurs o aula.
  - Permet realitzar reserves recursives.
  - SSO entre Drupal i MRBS.
- Desavantatges del mòdul per la integració al nucli de Drupal 7 / raons per les quals es requereix la utilització d'un altre mètode gestor de recursos:
  - No disposa de cap versió per al nucli 7.
  - No és compatible amb la versió 1.4.8 de MRBS.
  - No permet integrar més d'un usuari en una reserva.

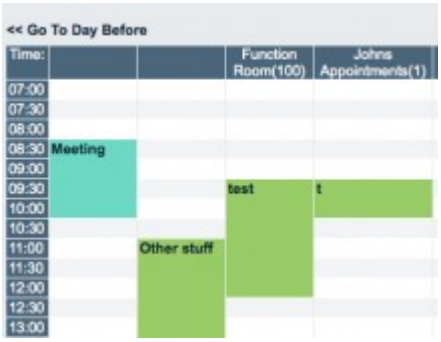

Figura 9: Bloc MRBS per Drupal

2.2.3 MRBS com a Sistema Gestor de Reserves per a Drupal i Moodle

Observades totes les avantatges i desavantatges dels mòduls existents podem veure com a la data de redacció d'aquest projecte no hi ha cap mòdul existent que complexi els requisits d'implementar un mòdul gestor de reserves que permeti la reserva de sales per part de professors amb la integració d'alumnes en concordança amb la capacitat de l'aula ni per Drupal7 ni per Moodle2.

Analitzant els mòduls existents i les tasques que s'haurien de dur a terme per a integrar un sistema gestor de reserves en concordança amb els requisits del projecte, trobem que el mòdul o bloc MRBS, tant per a Moodle com per a Drupal és el que més s'hi acosta. El fet de què MRBS sigui un sistema gestor extern fa que les funcionalitats siguin més avançades a qualsevol sistema gestor integrat a Drupal o Moodle.

En ambdós casos les tasques necessàries per a adequar el mòdul són l'adaptació a les versions de Drupal o Moodle utilitzades en el projecte i la integració d'alumnes en les reserves.

La integració amb Drupal permet més flexibilitat, ja que es poden realitzar reserves amb independència del curs, fent que un recurs o aula sigui disponible per a tots els cursos. Aquesta independència en la reserva de les aules no es permet amb la creació del bloc de Moodle.

Tenint en compte el comentat en l'anterior paràgraf i el fet de què la interfície de captació d'usuaris que utilitzarà el projecte serà Drupal, fa que aquest sigui el sistema seleccionat per a que MRBS obtingui les dades i permisos dels usuaris.

En el cas del projecte realitzat la interfície serà la de Drupal7, amb la qual cosa serà necessària la creació del mòdul de MRBS per a Drupal7 i l'adaptació del codi font d'MRBS per a operar amb les dades de Drupal7, ja que tal i com s'ha comentat en el punt 2.2.2.2 el Mòdul MRBS existent és per a Drupal 6 i MRBS 1.4.0.

A més per a què la integració sigui completa, també farà falta la integració dels usuaris de Drupal7 i Moodle2, fent que un usuari registrat a Drupal7 sigui el mateix a MRBS i Moodle2.

Per tant aquest projecte intenta solucionar la mancança d'un gestor de reserves d'aules amb professors i alumnes per a Drupal7 i Moodle2, fent us d'MRBS, en la seva versió 1.4.8.

Capítol 3: Disseny

3.1 Descripció general de la integració

El disseny del projecte es basa en la integració de Drupal, Moodle i MRBS, per tal que un usuari que accedeix a la interfície de Drupal (que proporciona el sistema de gestió i captació d'usuaris) o Moodle (que proporciona el campus virtual), tingui accés al sistema gestor de reserves i als cursos que proporciona Moodle, realitzant un únic registre i en concordança amb els rols que té assignats, a Drupal.

L'usuari amb rols d'administració o reserves (MRBS admin o MRBS Bookings) podrà realitzar reserves de recursos a MRBS. En el cas de la família de recursos *Aula*, l'usuari que realitza la reserva, podrà integrar els usuaris que tenen rols d'alumne. La reserva no permetrà integrar més usuaris a l'aula dels que permet la capacitat màxima de l'aula. També hi haurà un rol per a usuaris que únicament pugui accedir a veure les reserves realitzades MRBS (MRBS view).

D'aquesta manera, un dels primers requisits per a poder integrar correctament els diferents sistemes que s'interconnecten, ja sigui integrat en la interfície del sistema principal, o de forma externa, és el registre únic d'usuari o *Single-Sign On*. El registre únic permetrà que l'usuari es mogui pels diferents sistemes a través d'enllaços integrats en els mateixos sense haver de registrar-se en cada sistema que composa la interfície general.

A continuació és mostra i descriu l'esquema gràfic de la integració dels sistemes que formen part del projecte i la interacció dels mateixos amb l'usuari: anònim, registrat o administrador, que ja s'havia presentat en el punt 1.1 de la introducció:

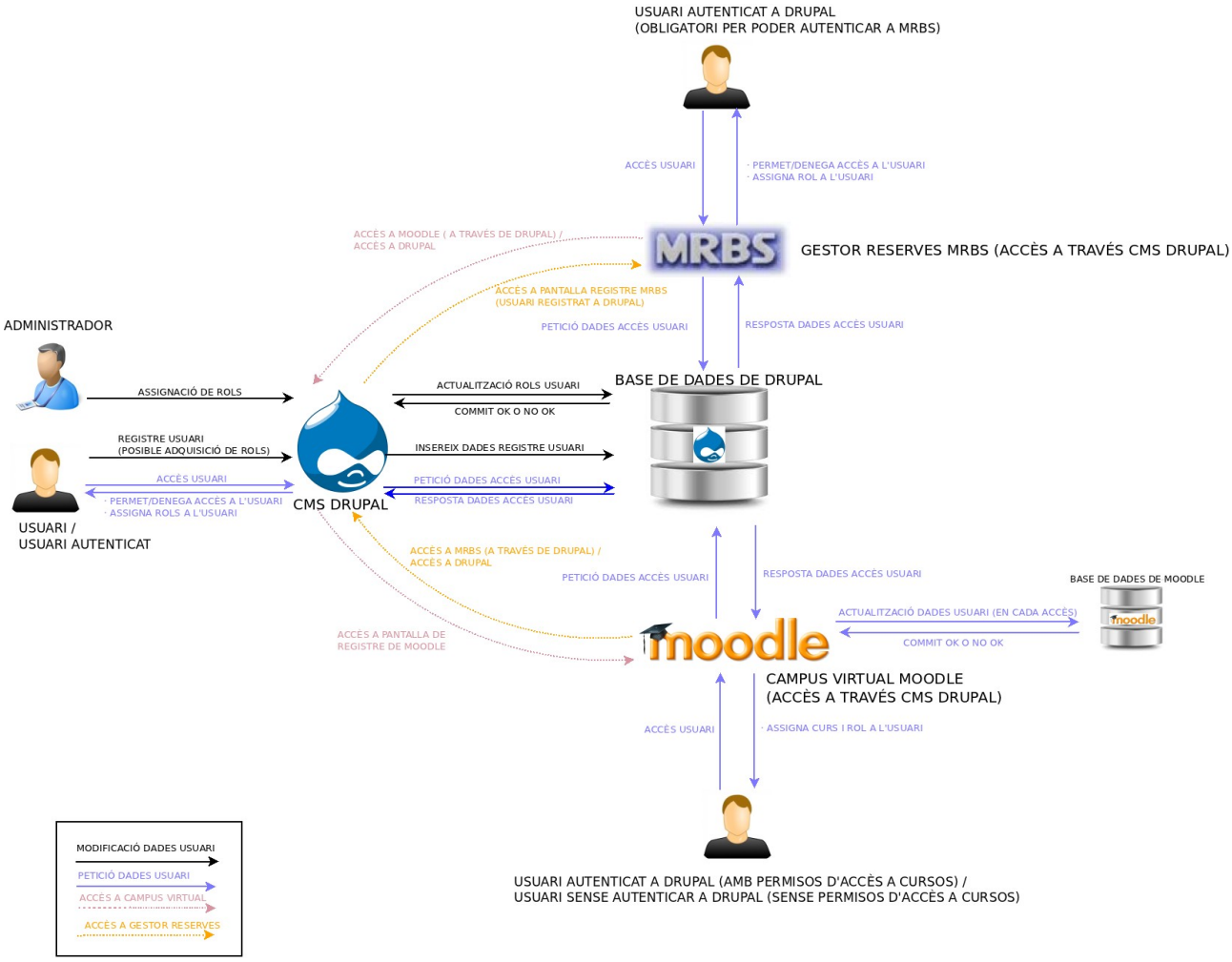

Figura 10: Integració de sistemes (2)

Tal i com es pot observar al gràfic, la interfície que permet i gestiona el registre d'usuaris es la de Drupal, des d'on es pot assignar un rol a un usuari, ja sigui per part de l'administrador, o per una adquisició del propi usuari. Tant les dades del registre de l'usuari (nom, contrasenya, rols) com les posteriors actualitzacions, quedaran desades a la base de dades de Drupal. Els sistemes Moodle i MRBS accediran a la base de dades de Drupal per a verificar les dades d'accés de cada usuari, així com els rols dels qual disposa cada usuari.

Un usuari podrà accedir al campus virtual de Moodle a través de la interfície de Drupal, si l'usuari està registrat i disposa d'algun rol que doni accés a algun curs disposarà d'accés al mateix. Si l'usuari no està registrat o no disposa de rols que donin accés als cursos només podrà accedir als cursos oberts a tots els usuaris. Les dades

d'accés de l'usuari es comproven a la base de dades de Drupal, i es desen i actualitzen a la base de dades de Moodle per a què el sistema pugui realitzar posteriors operacions amb l'usuari registrat (com per exemple guardar camps d'usuari que només queden registrats a Moodle).

En el cas d'MRBS únicament els usuaris registrats i amb rols d'accés, reserva o administració d'MRBS podran accedir al sistema i realitzar les accions en concordança amb el rol que se'ls ha assignat. En el cas d'MRBS les dades no queden registrades a la base de dades d'MRBS, ja que al ser un sistema menys complex que Moodle, no es realitzen operacions addicionals amb les dades de l'usuari. D'aquesta manera, en cas que un usuari vulgui accedir a MRBS desde Moodle, haurà de passar per la interfície d'accés de Drupal.

Tot i que les dades d'accés s'obtinguin i gestionin a través de la base de dades de Drupal, cada sistema disposa de la seva pròpia variable de sessió que s'encarregarà de mantenir i controlar l'accés de cada usuari segons les regles de cada sistema, amb el que la interfície de Drupal serà el punt únic de registre i modificació de les dades bàsiques de l'usuari, però l'usuari s'haurà d'autenticar en els tres sistemes.

Si el disseny requerís de Moodle com a sistema gestor d'usuaris i rols, sense disposar de l'interfície de Drupal, es podria modificar l'accés a la base de dades de Drupal per l'accés a la base de dades de Moodle per a què MRBS integrès alumnes prenent Moodle com a sistema gestor d'alumnes, de forma senzilla.

El segon requisit que presenta la integració és la de permetre que els usuaris amb permisos de reserva o administració d'MRBS puguin integrar alumnes als recursos de tipus aula, ja que el sistema MRBS en la versió d'aquest projecte, la 1.4.8, no permet integrar usuaris a la reserva. La interfície d'integració d'alumnes ha de ser amigable per l'usuari, poden filtrar per tipus d'alumne i poden editar reserves simples i reserves múltiples creades anteriorment.

A continuació és mostra i descriu l'esquema gràfic del procés de reserva d'aules, que ja s'havia presentat en el punt 1.1 de la introducció:

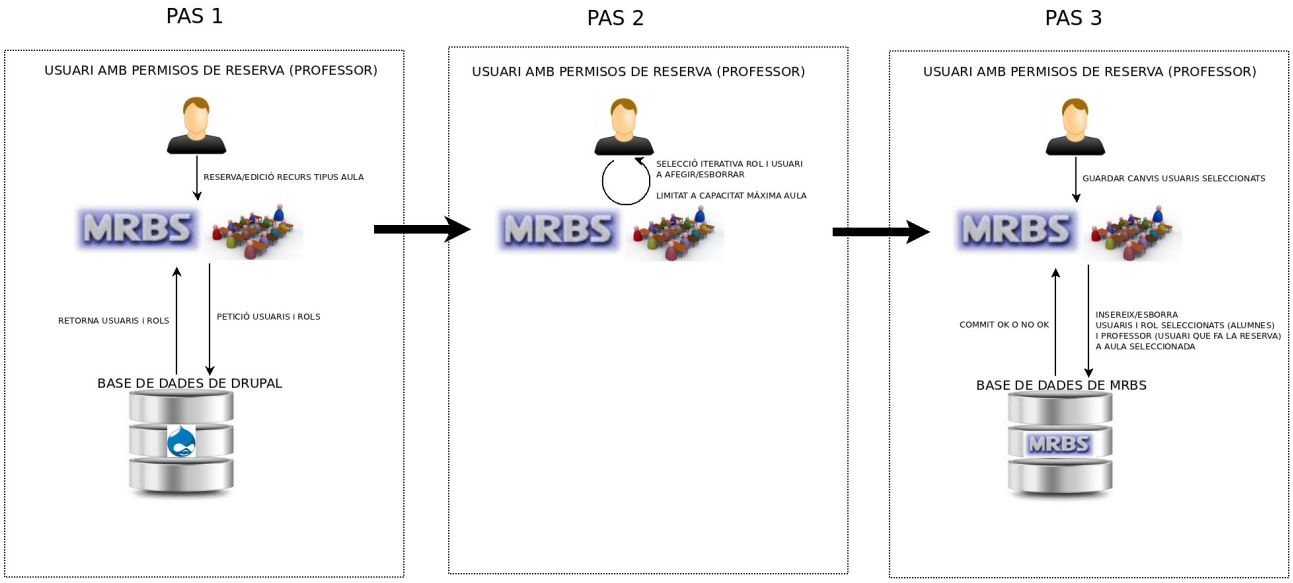

Figura 11: Procés de reserva d'aules (2)

Quan un usuari que disposa de permisos de reserva a MRBS accedeix a realitzar o editar una reserva d'un recurs de la família aula, el sistema sol·licita les dades dels usuaris alumnes i rols que tenen assignats a la base de dades de Drupal. Una vegada les rep les presenta a l'usuari, i aquest afegeix els alumnes que desitja, de forma temporal, amb la limitació d'un màxim d'alumnes en concordança amb la capacitat màxima d'alumnes de la aula. Una vegada finalitzada la selecció l'usuari desa la reserva de la aula, amb els usuaris associats a la reserva a la base de dades d'MRBS, des d'on es prendran les dades per a poder consultar o editar la reserva.

L'usuari que realitza la reserva quedarà assignat com a professor, per tant el rol d'administració o reserves d'MRBS sempre serà assignat a un professor.

D'aquesta manera podem concloure que per a dur a terme el projecte serà necessària la creació o integració de tres mòduls:

- Mòdul per a permetre el *Single-sign On* entre Drupal i Moodle.

- Mòdul per a permetre el *Single-sign On* entre Drupal i MRBS.

- Mòdul (o modificació del codi font) per a permetre la integració d'usuaris a les reserves per a MRBS.

A més serà necessària la creació dels següents blocs que permeten l'accés entre sistemes:

- Bloc per enllaçar Drupal amb Moodle.

- Bloc per enllaçar Drupal amb MRBS.

En els següents punts s'estudia el disseny de cada mòdul o funcionalitat necessària.

3.1.1 Creació de mòdul per a permetre el Single-sign On entre Drupal i Moodle

L'autenticació o registre únics, *Single sign-on* (SSO) en anglès, és un mètode que permet a un/a usuari/ària haver de procedir només a una autenticació o registre per accedir a diverses aplicacions informàtiques o llocs web.

Els objectius de l'autenticació única són diversos:

• Simplificar per a la persona usuària la gestió d'identificadors i contrasenyes.

• Guanyar en gestió eficient de la seguretat dels sistemes informàtics ja que com més contrasenyes hagi de gestionar un/a usuari/ària, més tendència tindrà a fer servir contrasenyes similars i/o senzilles de memoritzar, amb la qual cosa es va rebaixant el nivell de seguretat de les contrasenyes.

• Simplificar la gestió de les dades personals hostatjades pels diversos serveis en línia.

• Simplificar la definició i la posada en marxa de polítiques de seguretat informàtica.

Els sistemes d'autenticació única que trobem al mercat, també es poden categoritzar segons la llicència de distribució:

- Programari propietari, com poden ser SiteMinder o OneSign
- Programari lliure com pot ser OpenID.

En els següents subapartats passem a veure els sistemes d'autenticació única existents entre Drupal i Moodle, així com l'elecció del més adequat per al projecte.

3.1.1.1 Mòduls d'Autenticació Única per a Drupal - Moodle

Als projectes de Drupal i Moodle trobem diversos mòduls contribuïts que permeten realitzar autenticació única (ja sigui a nivell general o entre ambdós CMS), mitjançant diversos protocols o sistemes d'identificació digital descentralitzats, com poden ser OpenID, OpenID Provider, Single Sign-on, LDAP Single Sign-On, Moodle Single Signon, Moodle_Course entre molts d'altres en el cas de Drupal. En el cas de Moodle trobem els blocs Drupal SSO, Drupal Signle-Domain SSO o Moodle LDAP SSO Authentication entre molts d'altres.

A continuació, analitzarem els principals mòduls que permeten realitzar Signle Sign-On entre Moodle - Drupal, així com les seves característiques principals i les raons per les quals són vàlids o no per a la integració en el projecte.

3.1.1.1.1 Principals Mòduls d'Autenticació Única per a Drupal

A la data de redacció d'aquest projecte trobem diversos Mòduls que permeten realitzar Autenticació única entre Drupal i Moodle, tals com Moodle Single Signon, OpenID i OpenID Provider (veure Figura 12) o Moodle_Course. Aquests mòduls els podem trobar a la pàgina de Drupal.

A continuació es presenten les principals característiques i les avantatges i inconvenients de la integració de Moodle Single Sign-on en el projecte:

- Nom: Moodle Single Signon
- Categoria: Third-party Integration, Utility.
- Versions per a descarrega: 5.
- Lloc del projecte: http://Drupal.org/project/Moodlesso
- Responsable: Chris Johnson.
- Descripció: mòdul de Drupal que permet activar una cookie compartida i mantenir una base de dades d'usuaris que pot llegir Moodle. Utilitze el mode de creació de comptes de Moodle "lazy account creation".
- Avantatges del mòdul per la integració al nucli de Drupal 7:
  - Possible base per a desenvolupar la integració de Moodle 2 amb el nucli 7 de Drupal.
- Desavantatges del mòdul per la integració al nucli de Drupal 7:
  - No disposa de cap versió per al nucli 7.
  - El mòdul està en un estat de desenvolupament i manteniment desconegut.

A continuació es presenten les principals característiques i les avantatges i inconvenients de la integració de OpenID i OpenID Provider en el projecte:

- Nom: OpenID i OpenID Provider.
- Categoria: Third-party Integration, User Access & Authentication.
- Versions per a descarrega: 5,6,7,8 (Incluit al nucli).
- Lloc del projecte: http://Drupal.org/project/openid i http://Drupal.org/project/openid_provider
- Responsable: walkah.
- Descripció: mòdul del nucli de Drupal que permet l'autenticació dels usuaris utilitzant el protocol OpenID en mode client o servidor en funcio del mòdul integrat.
- Avantatges del mòdul per la integració al nucli de Drupal 7:
  - Permet autenticació única entre diversos llocs web o CMS integrats al mateix servidor.
- Desavantatges del mòdul per la integració al nucli de Drupal 7:

- Requereix de servidor OpenID que s'encarregui de l'autenticació.
- Requereix que tots els usuaris disposin d'un compte amb un servidor OpenID.
- En el cas del client, complica la operativa en la utilització de les dades dels usuaris versus mètodes d'autenticació en bases de dades internes.
- En el cas del servidor, fa que el lloc web es converteixi en proveïdor de comptes OpenID.
- Requereix que la resta de CMS integrats al lloc principal disposin d'autenticació amb OpenID (No disponible per a Moodle 2.2 ni per a RMBS a la data de la redacció d'aquest projecte).

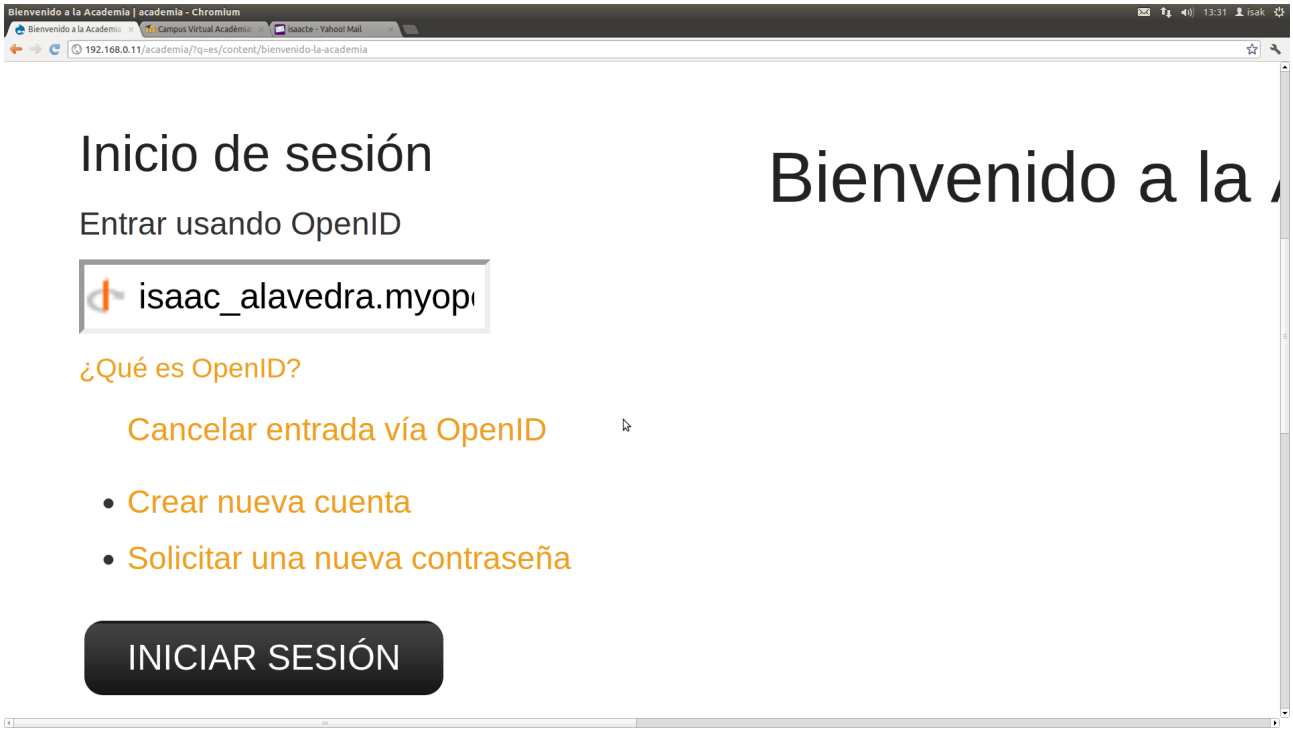

Figura 12: Mòdul OpenID (Client) per a Drupal

A continuació es presenten les principals característiques i les avantatges i inconvenients de la integració de Moodle_Course en el projecte:

-
- Nom: Moodle_Course.
- Categoria: Pendent de definir.
- Versions per a descarrega: 6,7 (Ambdós previstes).

- Lloc del projecte: http://Drupal.org/node/1472078
- Responsable: gkom.
- Descripció: mòdul de Drupal que permetrà la integració de Moodle en forma d'Iframe. Permetrà Single Sign-On entre Drupal i Moodle, permet presentar cursos de Moodle a Drupal entre d'altres característiques d'integració.
- Avantatges del mòdul per la integració al nucli de Drupal 7:
  - Permetrà autenticació única entre Drupal i Moodle.
  - Funcionalitats afegides d'integració entre Drupal i Moodle.
- Desavantatges del mòdul per la integració al nucli de Drupal 7:
  - Encara no hi ha disponible cap versió (Ni de desenvolupament). Primera versió de desenvolupament prevista per a finals de Juny de 2012.
  - No s'assegura la disponibilitat ver a la versió 2.x de Moodle (parteix del projecte Course que integra la versió 1.9 de Moodle).

3.1.1.1.2 Principals Blocs d'Autenticació Única per a Moodle

A la data de redacció d'aquest projecte trobem diversos Blocs que permeten realitzar Autenticació única entre Moodle i Drupal, tals com **Drupal_SSO, Drupal Single-Domain SSO i Moodle OpenID** (veure Figura 13) o **Moodle_Course.** Aquests blocs els podem trobar a la pàgina de Moodle.

A continuació es presenten les principals característiques i les avantatges i inconvenients de la integració de Drupal_SSO en el projecte:

- Nom: Drupal_SSO.
- Categoria: Authentication Method.
- Versions per a descarrega: 1.9 i 2.0.
- Lloc del projecte: http://Moodle.org/mod/data/view.php?d=13&rid=4240&filter=1
- Responsable: Scott Schaffter and Arsham Skrenes.
- Descripció: mòdul que habilitat el Single Sign-On entre Moodle 1.9 o 2.0 i Drupal 6, mitjançant dues bases de dades (una per CMS).
- Avantatges del mòdul per la integració al nucli de Moodle 2.x:
  - Permet Single Sign-On entre Moodle i Drupal 6.

- Desavantatges del mòdul per la integració al nucli de Moodle 2.x:
  - No permet Single Sign-On entre Moodle 2 i Drupal 7 ni hi ha prevista cap migració.

A continuació es presenten les principals característiques i les avantatges i inconvenients de la integració de Drupal Single-Domain SSO en el projecte:

- Nom: Drupal Single-Domain SSO.
- Categoria: Authentication Method.
- Versions per a descarrega: 1.9 i 2.0.
- Lloc del projecte: http://Moodle.org/mod/data/view.php?d=13&rid=2941&filter=1
- Responsable: Federico Heinz.
- Descripció: mòdul que habilita el Single Sign-On entre Moodle 1.9 o 2.0 i Drupal 6, mitjançant una única base de Dades.
- Avantatges del mòdul per la integració al nucli de Moodle 2.x:
  - Permet Single Sign-On entre Moodle i Drupal 6.
- Desavantatges del mòdul per la integració al nucli de Moodle 2.x:
  - No permet Single Sign-On entre Moodle 2 i Drupal 7 ni hi ha prevista cap migració.

A continuació es presenten les principals característiques i les avantatges i inconvenients de la integració de Moodle OpenID en el projecte:

- Nom: Moodle OpenID.
- Categoria: Authentication Method.
- Versions per a descarrega: 1.9.
- Lloc del projecte: https://launchpad.net/Moodle-openid
- Mantenidor: Stuart Metcalfe.
- Descripció: mòdul contribuït de Moodle que permet l'autenticació dels usuaris utilitzant el protocol OpenID en mode client.
- Avantatges del mòdul per la integració al nucli de Moodle 2.x:
  - n/a
- Desavantatges del mòdul per la integració al nucli de Moodle 2.x:

- No funciona correctament amb Moodle 2.2.

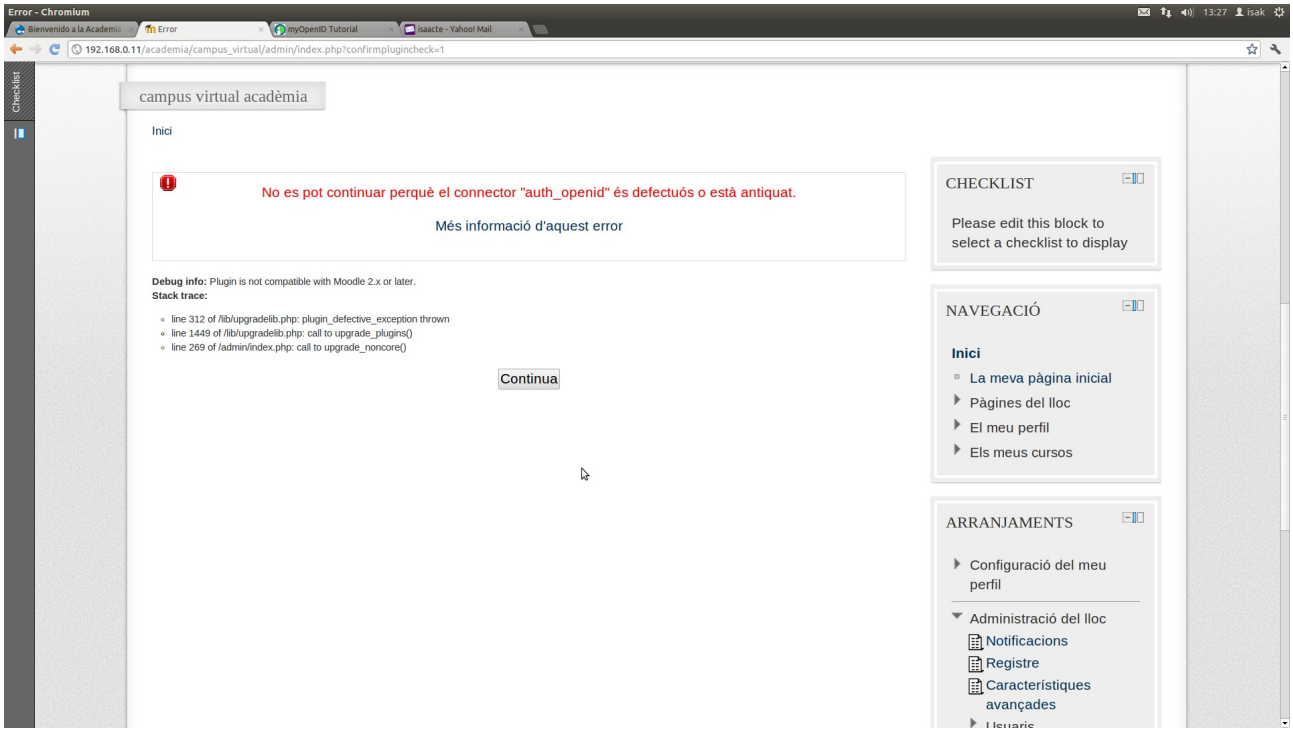

Figura 13: Bloc Moodle OpenID per a Moodle.

3.1.1.2 Elecció del mòdul per al Single-Sign On entre Drupal i Moodle

Tal i com s'ha pogut comprovar en els punts 3.1.1.1.1 i 3.1.1.1.2, a la data de creació d'aquest projecte, no es disposa de cap mòdul o bloc que permeti realitzar *single-sign on* entre Drupal 7 i Moodle 2.2.

El nucli de Moodle disposa d'un bloc que permet prendre les dades d'accés en una base de dades externa (el bloc autenticació en base de dades externa) i d'un bloc que permet comprovar els cursos i rols de curs en que està inscrit un usuari en una base de dades externa (el bloc inscripcions en base de dades externa).

Mitjançant aquests blocs es poden obtenir les dades d'accés, rols, i inscripció a cursos de l'usuari que hi ha a la base de dades de Drupal, permetent registre únic i controlant l'accés als cursos del campus virtual, a través de Drupal.

Per a dur a terme aquesta tasca, s'ha de generar una nova taula a la base de dades de Drupal que obtingui valors actualitzats de rols de cada usuari, i que assigni els rols de Professor de Moodle a certs rols de Drupal, ja que les taules existents no tenen les dades requerides per els blocs d'autenticació única i inscripcions de Moodle.

La correspondència de valors és la següent:

- Curs ID de Moodle <-> Rol ID de Drupal.
- name de Moodle <-> username de Drupal.
- rolename de Moodle <-> estudiant per defecte, fent que sigui professor o manager per a certs rols de Drupal.

D'aquesta manera, un usuari tindrà accés a un curs en funció del rol assignat a Drupal. Per exemple, un usuari amb el rol primària a Drupal, tindrà accés al curs primari a Moodle.

El rol de l'usuari als cursos de Moodle també depenent del rol assignat a Drupal. Per defecte es el d'estudiant, ara bé, si l'usuari té el rol de professor de Drupal passa a tenir el rol de professor als cursos en que te permís d'accés a Moodle. Per exemple un usuari que té el rol de professor i primària a Drupal, tindrà accés al curs de primària amb rol de professor. A la resta de cursos no tindrà accés si no se li assigna el rol pertinent.

Aquesta disseny és vàlid per a llocs on el nombre de cursos es reduït, com n'és el cas de la cooperativa en la qual es desenvolupa el codi del projecte, ja que sinó el nombre de rols necessaris seria molt gran.

D'altra banda, el projecte Moodle_Course de Drupal està treballant amb la creació d'un Mòdul que en breu està previst que s'alliberi a la comunitat. Aquest mòdul pot millorar

alguna de les característiques de la integració que es dissenya en aquest projecte, pel que podria ser una bona opció per a la integració futura de Drupal i Moodle.

Tenint en compte el paràgraf anterior, s'opta per a utilitzar els blocs de autenticació en base de dades externa i inscripcions en base de dades externa i la creació d'un mòdul per a generar les dades per a les inscripcions i autenticació en base de dades externa a Drupal, i es deixa la migració al mòdul Moodle_Course com a una possible millora futura.

La opció de disseny seleccionada, permet el registre únic però requereix d'un accés independent ja que cada sistema disposa de la seva variable de sessió, on guarda les dades d'accés.

Per a realitzar el modul s'utilitzarà l'API de programació de Drupal, que es basa en el llenguatge PhP, seguint les directrius per a la creació de mòduls per a Drupal. També s'utilitzarà MySQL per a crear la Taula i es configuraran els mòduls d'autenticació i inscripcions en base de dades externa de Moodle.

3.1.2 Creació de mòdul per a permetre Single-sign On entre Drupal i MRBS

El projecte MRBS no disposa de cap mòdul contribuïts que permeti Single Sign-On ja que el projecte no incorpora mòduls.

En el cas de Drupal, trobem el mòdul MRBS que permeten SSO entre Drupal i MRBS i que s'ha presentat en el punt 2.2.2.2.

Aquest Mòdul permet realitzar Single-Sign On entre Drupal i MRBS, i per a dur a terme aquesta funcionalitat és totalment vàlid, ja que ens permet una millor integració i més fàcil a nivell de manteniment que si utilitzem algun dels mòduls per a SSO no específics presentats en l'apartat 3.1.1.1.1.

Serà necessària, però, la migració del mòdul a la versió 7 de Drupal, ja que la versió que hi ha disponible es per a la versió 6 de Drupal, i la migració del codi d'MRBS a la versió 1.4.8, ja que el modul que hi ha disponible treballa amb la versió 1.4.0 d'MRBS.

Per a realitzar el modul s'utilitzarà l'API de programació de Drupal, que es basa en el llenguatge PhP, seguint les directrius per a la creació de mòduls per a Drupal.

3.1.3 Modificació codi font MRBS per la integració d'usuaris a les reserves

Per tal de dur a terme la tercera funcionalitat que ha d'implementar el projecte, serà necessària la modificació del codi font d'MRBS per tal que les pantalles de creació i d'edició de reserves permetin la inserció i l'esborrat d'usuaris de la reserva, ja que MRBS no permet la creació de mòduls contribuïts, tal i com s'ha descrit en la figura 11 de l'apartat 3.1.

Al punt 4.1.3 del capítol de la implementació es detallen els arxius i codi implicat en la integració d'usuaris en les reserves.

D'una forma senzilla es podria modificar l'accés a la base de dades de Drupal per l'accés a la base de dades de Moodle per a que MRBS integres alumnes prenent Moodle com a sistema gestor d'alumnes, canviant la base de dades a la qual s'ha d'accedir i els noms de les taules que contenen els usuaris i rols. En aquest projecte no es realitza ja que totes les dades de l'usuari es centralitzen a Drupal.

Per a canviar el codi font d'MRBS s'utilitzarà el llenguatge de programació PhP, juntament amb la seva API, javascript per a executar les funcions en local, i MySQL per a la connexió amb la base de dades, seguint els llenguatges utilitzats per a implementar MRBS.

3.1.4 Creació de Blocs per l'accés entre Drupal – Moodle i Drupal - MRBS

L'ultima fase a tenir en compte en el disseny de la integració consta de la creació dels blocs necessaris per a enllaçar Drupal, Moodle i MRBS.

Tenint en consideració que el sistema que fa de nexe entre Moodle i MRBS és Drupal, i que els tres sistemes s'executen en finestres independents, serà necessari crear els següents blocs per a realitzar els enllaços:

- Nou bloc a Moodle que permet enllaçar amb Drupal.

- Creació de dos enllaços al menú d'usuari de Drupal que permeten enllaçar amb Moodle i MRBS.

Per a la creació del bloc de Moodle i dels enllaços al menú d'usuari de Drupal no serà necessari cap llenguatge de programació, ja que es realitzarà mitjançant mòduls d'administració dels nuclis d'ambdós sistemes.

3.2 Funcionalitats i Serveis de la Integració de Sistemes

Les principals funcionalitats i serveis que es duran a terme amb el projecte d'integració i millora dels sistemes Drupal 7, Moodle 2.2 i MRBS 1.4.8 s'enumeren a continuació:

- Migració del Mòdul MRBS a Drupal 7 que permetrà el registre únic i la integració entre Drupal 7 i MRBS 1.4.8, tant a nivell d'interfície com a nivell de permisos d'accès al sistema gestor de reserves.

  Aquest Mòdul es posarà a disposició de la comunitat a través de la pàgina del projecte MRBS de Drupal (drupal.org/project/mrbs).

- Creació d'un Mòdul de Drupal 7 que permeti l'actualització de dades en una taula de la base de dades de Drupal des d'on Moodle obtindrà les dades d'inscripció als cursos. Aquest mòdul no s'alliberarà a la comunitat, ja que és especific pel projecte.

Per a permetre la integració i registre únic també es realitzarà la configuració dels blocs d'autenticació i inscripcions en base de dades externa de Moodle.

- Integració d'usuaris amb rol d'alumne en les reserves de recursos tipus aula en el sistema MRBS, mitjançant una interfície amigable per a l'usuari.

- Creació de blocs que continguin enllaços per a l'accés bidireccional entre les interfícies de Drupal - MRBS i Drupal - MRBS. L'accés entre Moodle - MRBS es realitza a través de la interfície de Drupal.

- Per a dur a terme la integració es realitzarà la configuració d'un servidor web Apache2, un gestor de base de dades MySQL, el CMS Drupal 7.12, l'LMS Moodle 2.2 i el sistema gestor de reserves MRBS 1.4.8.

3.3 Recursos necessaris: infraestructura, maquinari, programari, serveis i compres externs

3.3.1 Infraestructura

Per a dur a terme el projecte no es necessària infraestructura addicional.

3.3.2 Maquinari

Per a dur a terme el projecte no és necessària l'adquisició de Maquinari addicional. Qualsevol ordinador que disposi d'un sistema operatiu que permeti la integració d'Apache com a Servidor (amb la integració de les llibreries per a executar PhP), i que permeti l'execució de MySQL com a Motor de Base de Dades és vàlid per a dur a terme el projecte.

3.3.3 Programari

Per a dur a terme el projecte, serà necessari el següent programari:

- El projecte utilitzarà la distribució GNU/Linux Ubuntu 11.10 com a sistema operatiu base per a integrar el servidor Apache.
- El servidor web que s'utilitzarà per a executar la interfície web i integrar els diferents sistemes web necessaris per a dur a terme el projecte, serà Apache 2.
- El servidor de Base de Dades serà MySQL.
- El llenguatge servidor per a l'execució dels diversos aplicatius serà PHP5.
- El Sistema de Reserves serà implementat per part d'MRBS 1.4.8.
- El Campus Virtual serà implementat per a Moodle 2.2.
- El gestor de Continguts Web serà implementat per Drupal 7.12.
- El client Web serà FireFox o Google Chrome.

3.3.4 Serveis i Compres Externs

Per a dur a terme el projecte serà necessaris els serveis de l'estudiant, durant un temps de 3 mesos aproximats, i amb una dedicació de 10 hores la setmana. No serà necessària cap compra externa.

3.4 Llicencies dels Sistemes i Aplicatius necessaris

Tal i com s'ha comentat en punts anteriors, tots els components que formen el servidor de comunicacions unificades estan llicenciats sota alguna de les llicencies lliures o open source que trobem al mercat. D'aquesta forma totes compleixen les quatre llibertats del programari lliure:

- Llibertat per a executar el programa en qualsevol lloc, amb qualsevol propòsit i per sempre.
- Llibertat per a estudiar-lo i adaptar-lo a les nostres necessitats. Això exigeix l'accés al codi font.
- Llibertat de redistribució, de manera que se'ns permeti col·laborar amb veïns i amics.

- Llibertat per a millorar el programa i publicar-ne les millores. Això també exigeix el codi font.

A continuació es descriu la llicència de cadascun dels component principals, així com una breu descripció de les principals característiques de la mateixa:

- Drupal: La versió 7 de Drupal utilitza la GNU GPL, en la seva versió 2. La llicencia es pot trobar a **http://api.drupal.org/api/drupal/LICENSE.txt/7**. Aquesta és la mateixa llicencia que s'utilitza en els mòduls que s'han creat per aquest projecte amb el que la seva compatibilitat esta assegurada.
- Moodle: La versió 2 de Moodle utilitza la GNU GPL, en la seva versió 3. La llicencia es pot trobar a **http://docs.moodle.org/dev/License**.
- MRBS: La versió 1.4.8 de MRBS utilitza la GNU GPL, en la seva versió 3 i la PHP License. Es poden trobar a la pàgina que el projecte té a sourceforge.
- Apache: La versió 2 d' Apache utilitza la llicencia pròpia apache2. Aquesta llicència es pot trobar a **http://www.apache.org/licenses/LICENSE-2.0**.

Tal i com podem veure els tres sistemes que formen el projecte, Moodle, Drupal i MRBS tenen la llicencia GNU GPL en les seves versions 2 i 3, amb el que no hi ha problema en la seva compatibilitat.

Capítol 4: Implementació

4.1 Plantejament general de la implantació.

El desenvolupament del projecte constarà de cinc fases:

- Fase 1: Instal·lació dels sistemes necessaris per a dur a terme el projecte
- Fase 2: Creació del mòdul per a permetre el Single-sign On entre Drupal i MRBS.
- Fase 3: Modificació del codi font d'MRBS per a permetre la integració d'usuaris a les reserves de la família aules.
- Fase 4: Creació del mòdul per a permetre el Single-sign On entre Drupal i Moodle i integració de blocs d'autenticació i inscripcions de Moodle.
- Fase 5: Creació de blocs per a l'accés entre els diferents sistemes.

Tal i com s'ha anat comentant en la fase de disseny, el llenguatge seleccionat per a implementar el codi del projecte és PHP per a la creació dels dos mòduls necessaris per a Drupal, ja que és el llenguatge que s'utilitza en la creació de mòduls de Drupal (utilitzant la seva API), i PHP i javascript en el cas de la modificació del nucli de MRBS ja que també són els dos llenguatges utilitzats en aquest sistema.

A continuació es presenta una breu descripció del codi que forma part de cada Fase, el qual es pot trobar en l'apartat d'annexos.

4.1.1 Fase 1: Instal·lació dels sistemes necessaris per a dur a terme el projecte

Per a dur a terme el projecte s'ha instal·lat els següents sistemes i/o aplicatius i en aquest ordre:

1. Primerament, s'instal·la un sistema base, que en el cas del projecte es tracta d'Ubuntu 11.10, que ja implementa les funcions bàsiques del sistema així com de l'escriptori i aplicatius base necessaris.
2. Posteriorment s'instal·la el servidor Apache2 amb els paquets per a poder executar les funcionalitats bàsiques de seguretat i PhP5.
3. En tercer lloc, s'instal·la el sistema gestor de Bases de Dades MySQL, juntament amb el gestor PhPMyAdmin.
4. Una vegada s'instal·la el sistema base, el servidor i el gestor de base de dades ja es poden instal·lar els sistemes que formen part del projecte, Drupal, Moodle i MRBS.

En aquesta memòria no es descriuran les accions necessàries per a instal·lar cadascun dels sistemes presentats, sinó que ens centrarem en les accions necessàries per a permetre la integració de Moodle 2.2 i Drupal 7.x amb MRBS 1.4.8. A Internet es disposa de múltiples manuals per a instal·lar Moodle [4], Drupal [5] i MRBS [6] .

4.1.2 Fase 2: Creació de Mòdul per a permetre el *Single-sign On* entre Drupal i MRBS

El mòdul que permet la integració de Drupal i MRBS disposa dels arxius necessaris per a Drupal i els arxius necessaris per a MRBS. El llenguatge utilitzat en aquest mòdul es PhP i s'han utilitzat algunes funcions de la API de Drupal per a poder implementar-lo.

Per a MRBS s'han integrat al mòdul els següents arxius:

- `mrbs.info`, que conte la informació dels arxius i del propi mòdul.

- LICENSE.txt, que mostra la llicencia del mòdul.

- mrbs.module, que no conté codi però que és necessari per a tots els mòduls de Drupal.

- mrbs.install, que implementa els hooks [10] que creen els rols d'administració, reserva i vista per a MRBS en la seva instal·lació i els elimina en la seva desinstal·lació.

- README.txt, aquest fitxer explica on s'han d'ubicar els arxius específics d'MRBS i els canvis a realitzar a l'arxiu de configuració d'MRBS (config.inc.php).

- La carpeta mrbs_inc que conté els arxius session_drupal.inc i auth_drupal.inc. El primer fitxer s'encarrega de controlar la sessió d'MRBS, mantenint una variable de sessió pròpia per MRBS i realitzant accions per a millorar la integració entre sistemes, a més d'implementar les funcions *printLoginForm*, *authGet,* i *PrintLogonBox* que s'encarreguen de realitzar i comprovar nivells d'accés de l'usuari al sistema.

  El segon fitxer, auth_drupal.inc, implementa les funcions *authValidateUser* i *authGetUserLevel* que s'encarreguen de que MRBS comprovi les dades d'accés i els rols d'usuari a la base de Dades de Drupal, utilitzant algunes funcions pròpies de Drupal.

A l'annex 1 es pot comprovar el codi del mòdul amb les explicacions detallades de cada arxiu i funció integrades al codi. Les explicacions s'han realitzat en anglès per a permetre l'alliberament del mòdul a la comunitat.

El mòdul es pot trobar a **http://drupal.org/project/mrbs**, sent la última versió que s'ha implementat durant la creació del projecte la que es pot trobar a **http://drupal.org/node/1591964**. A les tres setmanes de quedar alliberat 8 llocs web reporten la utilització d'aquest mòdul.

10 http://api.drupal.org/api/drupal/includes!module.inc/group/hooks/7

4.1.3 Fase 3: Modificació del codi font d'MRBS per a permetre la integració d'usuaris a les reserves de la família aules

Per a permetre integrar usuaris en les reserves de MRBS s'han modificat arxius del nucli d'MRBS ja que, com s'ha comentat anteriorment, MRBS no disposa de mòduls contribuïts per a ampliar-ne les funcionalitats.

Els arxius que s'han modificat són els següents:

- edit_entry.php, que s'utilitza per a editar reserves.
- view_entry.php, que s'utilitza per a veure reserves ja realitzades.
- functions_table.inc, que inclou algunes funcions que s'utilitzen a edit_entry.php i view_entry.php
- functions_view.inc, que inclou algunes funcions que s'utilitzen a edit_entry.php i view_entry.php
- També s'ha modificat alguna plantilla de disseny css i els arxius de traducció al català i castellà i el fitxer de configuració d'MRBS config.inc.php.

A part dels arxius modificats s'han creat dos nous arxius que inclouen noves funcions per a inserir i eliminar usuaris de les reserves a la base de dades, l'*afegeix_usuari_reserva.php* i l'*elimina_usuari_reserva.php*.

Per a desar els alumnes que formaran part de cada aula, s'han modificat les taules mrbs_entry i mrbs_repeat que contenen les dades de les reserves (simples o múltiples).

Si bé és cert que la estructura més adequada per a permetre incloure una multitud de nous registres a les reserves, com es el cas dels alumnes, que poden arribar a ocupar 70 o 80 camps per reserva, és la de crear una nova taula que associi el parell reserva

– alumne. El fet de què la complexitat i el temps de modificació del codi font per a permetre integrar aquesta nova taula sigui massa alt per aquest projecte (principalment degut a les reserves múltiples), ha fet que s'opti per implementar l'opció que MRBS permet per a modificar els camps de les reserves; afegir camps addicionals a les taules mrbs_entry i mrbs_repeat per a que les reserves dels recursos de la família aula incloguin els alumnes. El màxim d'alumnes per reserva be donat pel nombre de camps nous que s'hagi destinat per alumnes, que en el cas del projecte és de 40.

Aquests camps són de tipus Varchar i incorporen el nom i el rol de l'alumne, que n'identifica el nivell i tipologia d'estudis.

Els llenguatges utilitzats per a modificar i crear els arxius del nucli de MRBS són php i javascript, a més d'utilitzar l'SQL per a la modificació de les taules, a través de l'administrador de MySQL PhPMyAdmin.

A l'Annex 2 es pot observar el detall del codi modificat així com l'explicació del mateix, sent aquest en llenguatge anglès, ja que tot i que no s'allibera per la comunitat, segueix l'idioma en que estan realitzats els comentaris del codi original.

Val a dir que, tot i que el codi implementat permet disposar de les noves funcionalitats, aquest es podria optimitzar, cosa que no s'ha dut a terme en aquest projecte per manca de temps.

4.1.4 Fase 4: Creació de Mòdul per a permetre el Single-sign On entre Drupal i Moodle

Per a permetre el registre únic entre Drupal i Moodle es configura el bloc d'autenticació en base de dades externa de Moodle. Les dades d'accés a Moodle es prenen de la taula d'usuaris de Drupal. En cada autenticacio a Moodle, les dades s' actualitzen a la taula d'usuaris de Moodle. Les dades de configuració del bloc són les que associen cada camp de la taula d'usuaris de Drupal amb els camps d'usuari de Moodle (per exemple user_name de Moodle amb name de Drupal,...).

Posteriorment es configura el bloc d'inscripcions en base de dades externa de Moodle. Aquest bloc pren les dades del nom d'usuari, curs i rol de l'usuari en el curs, d'una nova taula que es crea per aquest projecte, *user_roles_courses_moodle,* i s'actualitza a partir d'un nou mòdul que s'implementa per a Drupal, *moodle_actions*. La taula *user_roles_courses_moodle* es crea a la base de dades de Drupal.

El mòdul *moodle_actions*, aprofita les funcions del mòdul *actions*, del nucli de Drupal, per a generar una nova acció, que s'executa en cada entrada o vista del perfil d'usuari a Drupal, i que actualitza els camps de la taula *user_roles_courses_moodle* (veure Figura 14). Els camps d'aquesta taula són:

- rid: L'identificador únic de rol de l'usuari. Aquest camp es d'on Moodle pren el curs d'usuari, de tal forma que un usuari amb el rol de *primària* a Drupal pot accedir al curs de *primària* de Moodle.

- username: El nom únic que identifica l'usuari tant a Drupal com a Moodle.

- rolename: El rolename de l'usuari en el curs de Moodle. Per defecte es el d'estudiant, i s'assigna el de *editingteacher* i *manager*, en el cas de disposar del rol de *professor* o *manager* a Drupal.

- uid: L'identificador d'usuari a Drupal. Necessari per a la funció que actualitza els rols.

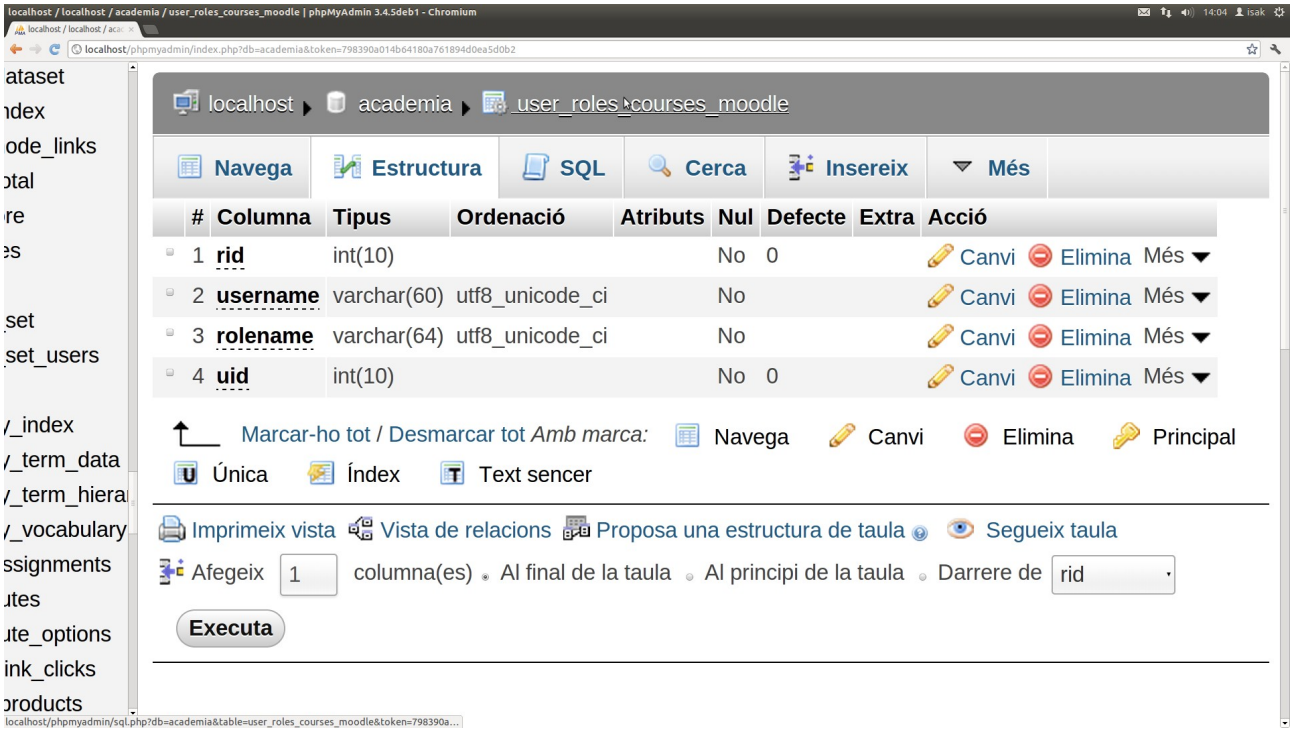

Figura 14: Vista taula user_roles_courses_moodle de Drupal

El mòdul *moodle_actions* consta dels següents arxius:

- moodle_actions.info, que conté la informació dels arxius i del propi mòdul.
- LICENSE.txt, que mostra la llicència del mòdul.
- moodle_actions.module, que implementa les accions que s'encarreguen d'actualitzar la taula user_roles_courses_moodle.

El codi detallat amb explicacions del nou mòdul de Drupal que s'encarrega d'actualitzar les dades a la taula, *moodle_actions*, es pot trobar a l'annex 3 i utilitza PhP com a llenguatge de programació seguint l'estructuració i API's de Drupal.

El mòdul de Drupal creat en aquesta fase no s'allibera a la comunitat, ja que es considera especific per a aquest projecte. Tot i així, segueix tota la normativa d'estructuració de mòduls de Drupal.

Cal esmentar que el projecte *Moodle_Course* de Drupal està treballant amb la creació d'un Mòdul per a la integració de Moodle i Drupal que en breu esta previst que s'alliberi a la comunitat. Aquest mòdul té prevista, entre d'altres funcionalitats, l'accés a Drupal i Moodle mantenint una única variable de sessió per ambdós sistemes, fet que podria millorar la integració que per manca de temps no s'ha pogut dur a terme en aquest projecte.

4.1.5 Fase 5: Creació de Blocs o Enllaços per a l'accés entre els diferents sistemes

L'ultima fase consta de la creació dels blocs necessaris per a enllaçar els Drupal, Moodle i MRBS.

Tenint en consideració que el sistema que fa de nexe entre Moodle i MRBS és Drupal, i que els tres sistemes s'executen en finestres independents, s'implementa els següents blocs en cada sistema, per a realitzar els enllaços:

- Nou bloc a Moodle que permet enllaçar amb Drupal. Aquest bloc es crea mitjançant l'opció inserir un nou bloc HTML, i es configura per aparèixer en totes les pàgines del campus virtual. Aquest bloc és un simple enllaç que fa retornar a l'usuari a la pàgina principal de Drupal sense tancar la sessió de Moodle, cosa que s'ha de realitzar manualment per part de l'usuari.

- Creació de dos enllaços al menú d'usuari de Drupal que permeten enllaçar amb Moodle i MRBS. L'enllaç per enllaçar amb MRBS només apareix pels usuaris que tenen rols a MRBS.

En el cas d'MRBS l'arxiu session_drupal.inc ja incorpora les funcions que gestionen l'accés a la interfície de Drupal, i que fan que un usuari s'hagi de des-autenticar abans de retornar a Drupal. D'aquesta manera la sessió queda tancada.

4.2 Calendari: etapes i fases, fites, seguiment del projecte, calendari

Al tractar-se d'un projecte on s'utilitza programari lliure es creu convenient utilitzar el mètode PERT mitjançant el qual s'estima el temps par cercar el camí critic. Adequant la implementació als recursos i temps disponibles.

Temps esperat=(duració_optimista + 4 x duració_probable + duració_pessimista) / 6
Anàlisi =(7 +7*4+7)/6= 7 dies.
Desenvolupament =(70+80*4+90)/6= 80 dies.
Implantació = A estimar, si s'esdevé.

Temps total projecte (anàlisi, desenvolupament) ~ 3 mesos (El temps d'implantació no es computa, ja que serà en funció de si s'esdevé o no).

El temps total previst per la formació serà en funció de la tipologia dels clients i del nombre de clients als quals s'hagi d'implantar el sistema.

Les fites de cada fase del projecte s'han considerat com a dependents, ja que tot i que de la fase 2 a la fase 4 es poden implementar de forma independent, sense assolir cadascuna de les fases el projecte no es pot completar al 100%.

A continuació es presenten el diagrama de gantt i de PERT representatius on es poden observar les etapes, fases, fites i seguiment del projecte:

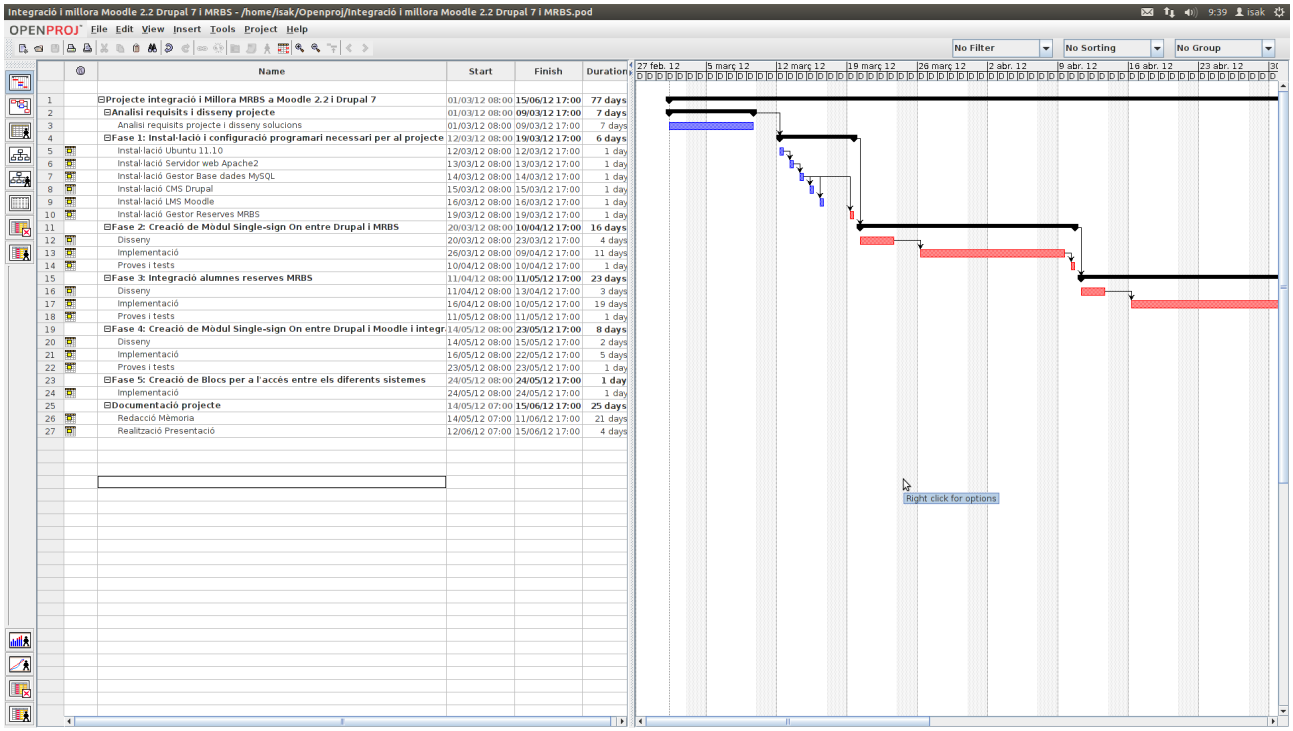

Figura 15: Diagrama gannt del Projecte 1

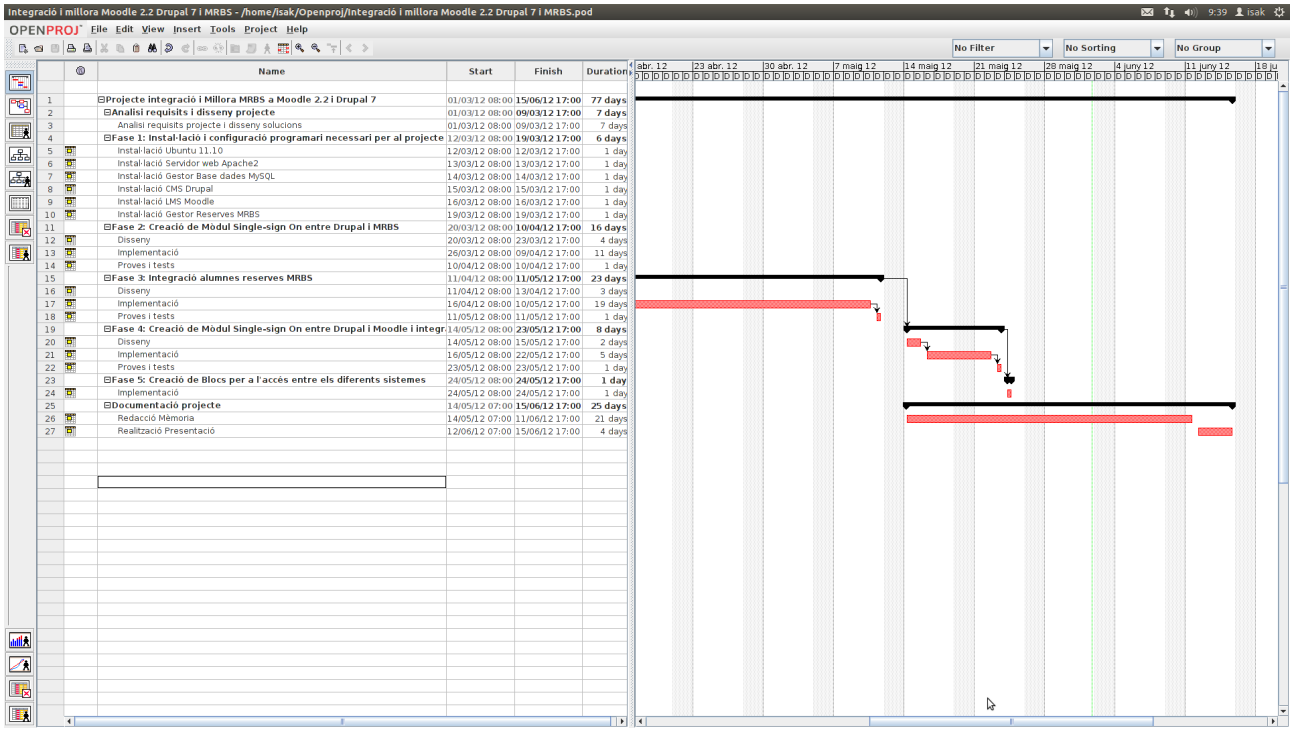

Figura 16: Diagrama gannt del Projecte (2)

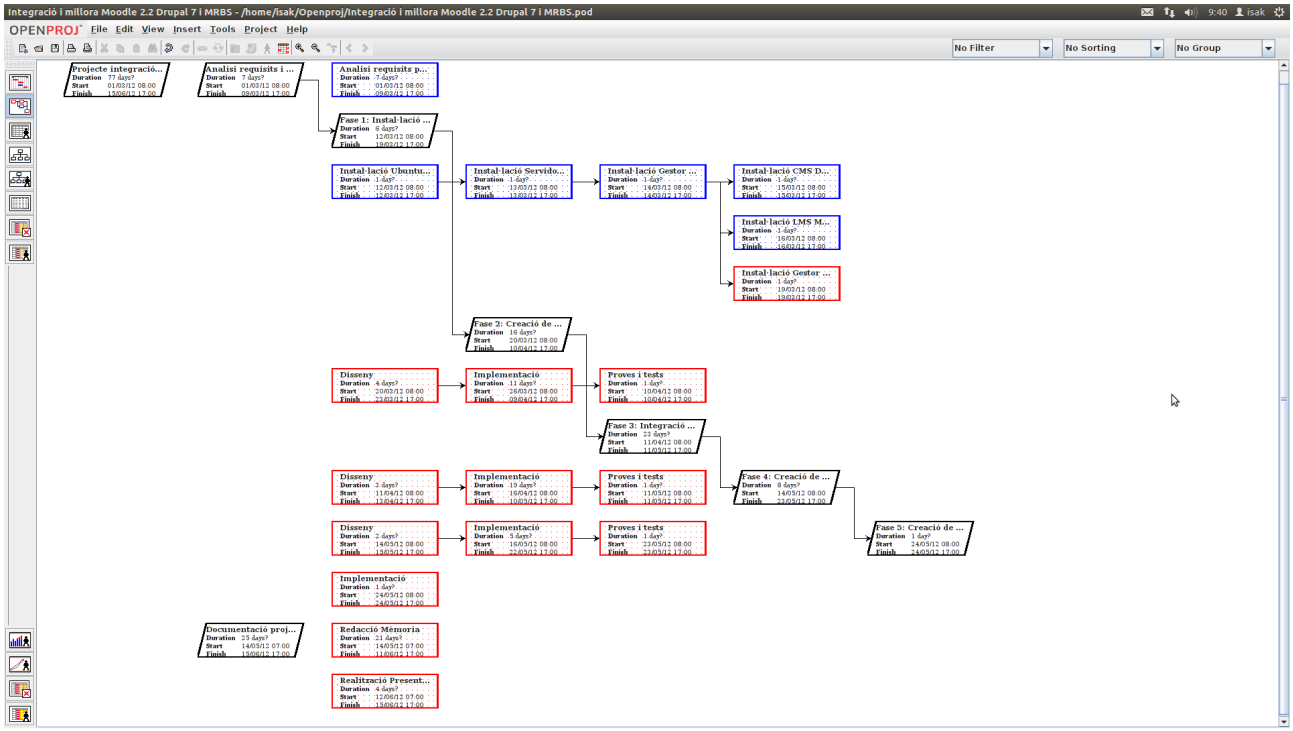

Figura 17: Diagrama PERT del projecte

4.3 Pressupost econòmic

El cost estimat per a la realització del projecte es de 13.600 Euros. Tot i així al tractar-se d'un projecte final de grau no implica cap cost real. Els costos consideren el treball d'una setmana a 40 hores (en realitat es dediquen 25 hores de mitjana).

Els costos anuals en manteniment i recanvis no s'avaluen ja que no hi ha maquinari implicat, i únicament s'hauria de considerar un petit cost pel manteniment i actualitzacions del programari implicat en concepte de manteniment. El manteniment del servidor no es considera.

A continuació es desglossen els costos estimats:

Costos estimats Projecte Integració i millora del Sistema gestor de Reserves MRBS 1.4.8 amb Moodle 2.2 i Drupal 7.x				
Concepte	Homes/Dia	Cost Homes/Dia	Cost Unitari en Euros	Notes
Gestió	90	50	4.500	Gestió del projecte durant 3 Mesos per una persona
Desenvolupament Integració Sistemes	90	100	9.000	Desenvolupament del projecte durant 3 mesos per una persona
Formació			0	Es pot utilitzar el material realitzat durant el desenvolupament del projecte
Instal·lació			100	Instal·lació codi desenvolupat en un Servidor
Maquinari			0	Es considera implementació sobre maquinari existent
Llicencies Software			0	Totes les llicencies son lliures sense cost associat
TOTAL			13.600	

Taula 1: Estimació de costos del projecte.

4.4 Gestió de Riscs i Propostes de solució

S'han identificat els següents riscs:

Riscs Temporals. El fet de que el cap de projecte té poca experiència prèvia en l'entorn de desenvolupament i aplicatius utilitzats fa que la valoració temporal pugui no ser exacte, amb el que es podria arribar al final del projecte i no acomplir els objectius. Aquest risc es podria solucionar allargant el temps del projecte o assignant un altre desenvolupador per a finalitzar-lo.

Riscs de Programari. El fet de que la integració es realitzi per a una versió concreta del gestor de reserves, fa que les posteriors actualitzacions no es pugin realitzar de forma automàtica, amb el que el gestor pot quedar desfasat en poc temps. La solució seria

mirar d'integrar funcionalitats noves al gestor MRBS en forma de mòdul de forma que es pogués independitzar el codi font de les noves funcionalitats afegides.

4.5 Integrants del projecte

El projecte està organitzat i estructurat per a que sigui dut a terme per part d'un únic component, que s'encarrega de realitzar les fases que componen el projecte, i de la gestió del mateix.

Capítol 5: Demostració del funcionament de la integració i millora del sistema gestor reserves

Per tal de demostrar el funcionament del sistema gestor de reserves utilitzarem una interfície web que incorpora els mòduls, configuracions i codi implementat per aquest projecte i ho compararem amb una altra interfície web, amb els mateixos sistemes del projecte, però que no n'incorpora el codi implementat.

A les figures de les demostracions que es presentaran a continuació, podem observar la interfície 1 amb el codi del projecte a l'esquerra de la imatge i la interfície 2 que no incorpora el codi a la dreta de la imatge.

Iniciarem la demostració creant un usuari de prova amb la contrasenya 1234 en ambdues interfícies tal i com podem observar a la Figura 18, assignant els rols de *primària* i *mrbs bookings* a l'usuari de la interfície que incorpora el codi desenvolupat pel projecte.

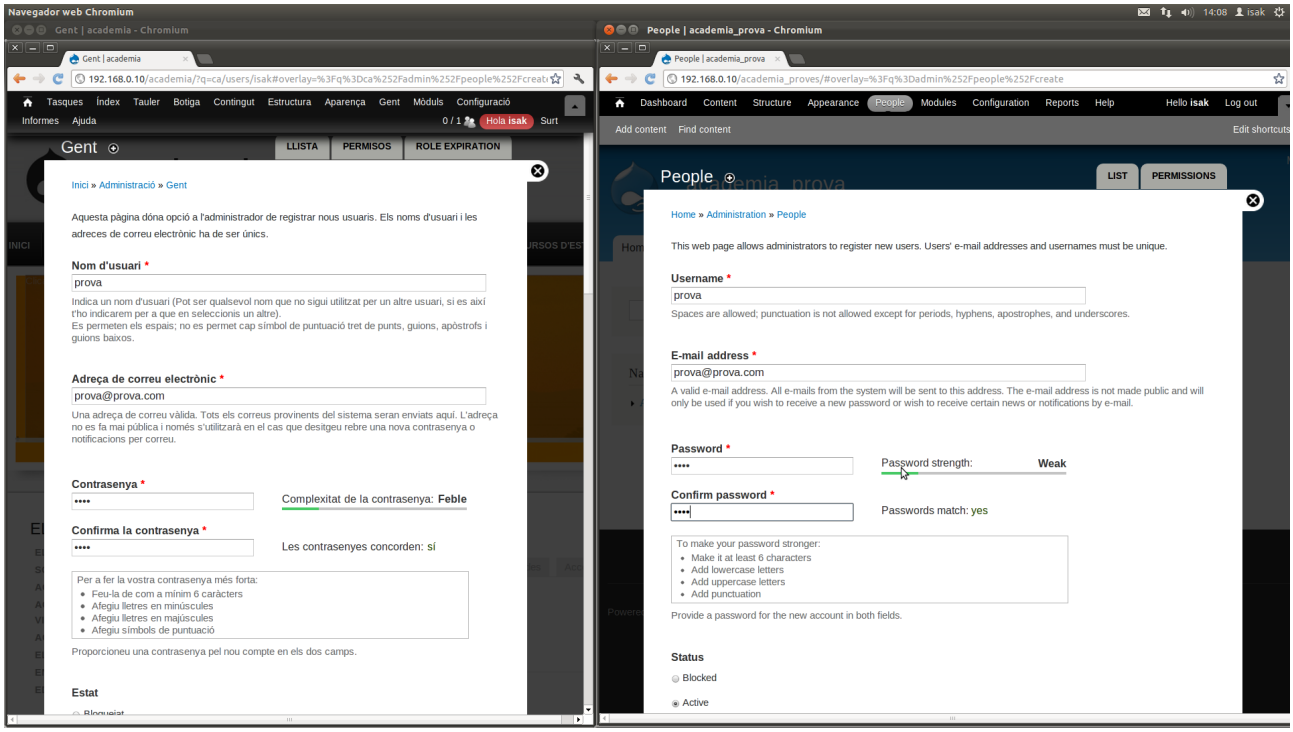

Figura 18: Demostració. Creació nou usuari

Posteriorment farem l'accés al sistema amb l'usuari de prova i accedirem al sistema gestor de reserves, mitjançant l'enllaç del menú personal de l'usuari, en el cas de la interfície 1. Per a la interfície 2 l'accés s'ha de realitzar de forma manual a través de l'explorador.

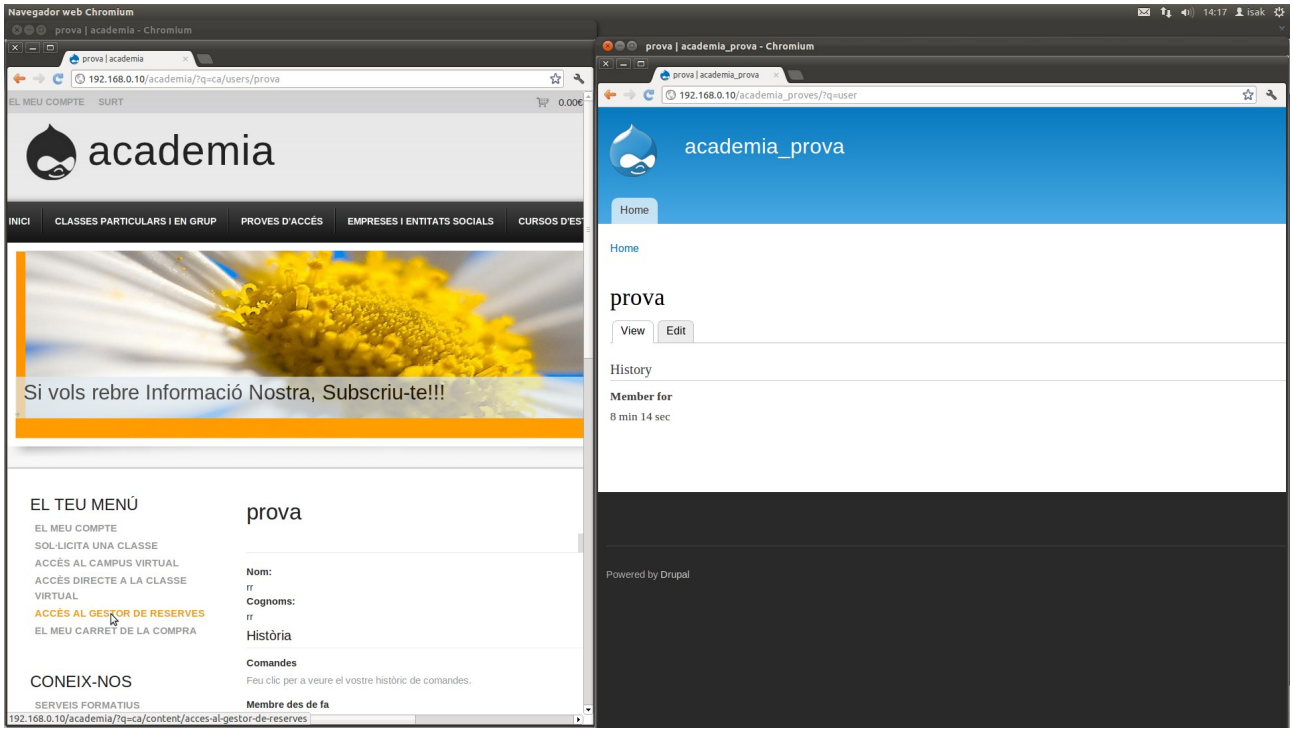

Figura 19: Demostració. Accés interfície Drupal

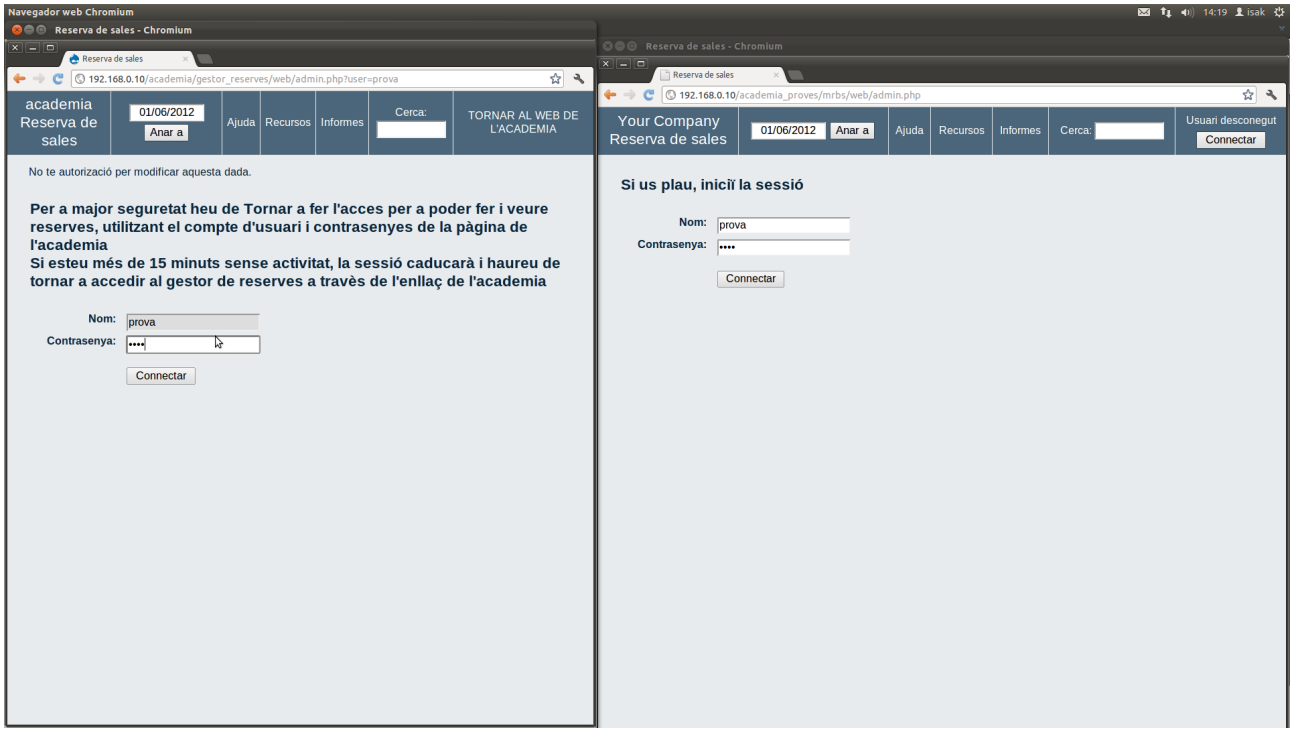

Figura 20: Demostració. Accés interfície MRBS desde Drupal

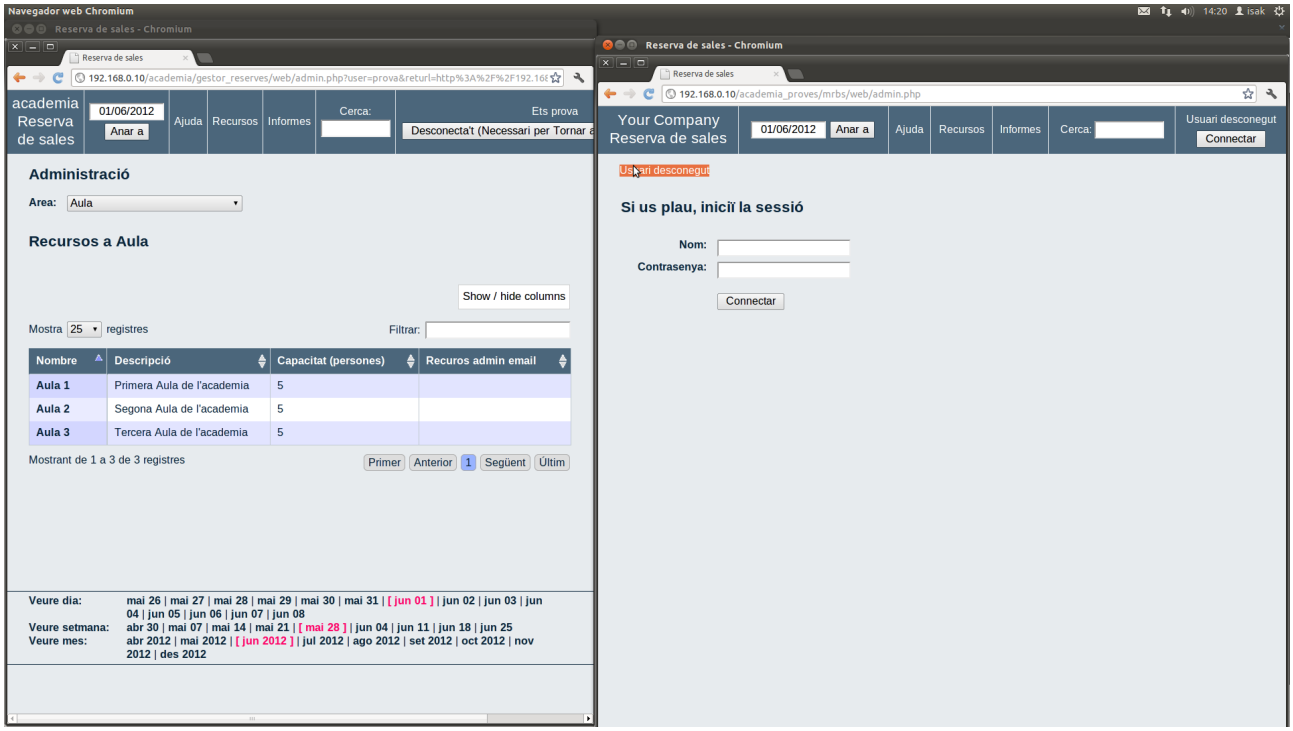

Figura 21: Demostració. Accés interfície MRBS acceptat/denegat

Tal i com podem observar a la Figura 21 l'accés amb l'usuari que acabem de crear a la interfície 2 es correcte, en canvi en la 2 no es reconeix, ja que pren les dades de la base de dades pròpia de MRBS.

Posteriorment creem un usuari manualment a la interfície 2 i accedim a realitzar una reserva senzilla per a comprovar com a la interfície 1 es poden afegir alumnes a la reserva i a la que no n'incorpora no s'hi poden afegir. Intentem afegir l'alumne que acabem de crear, i que disposa del rol de *primària*, i veiem com a la figura 22, el sistema ens indica que no ens hem d'afegir a la reserva, ja que l'usuari que realitza la reserva s'insereix en la mateixa com a professor automàticament, sent els professors els únics que tenen rols de reserves a MRBS.

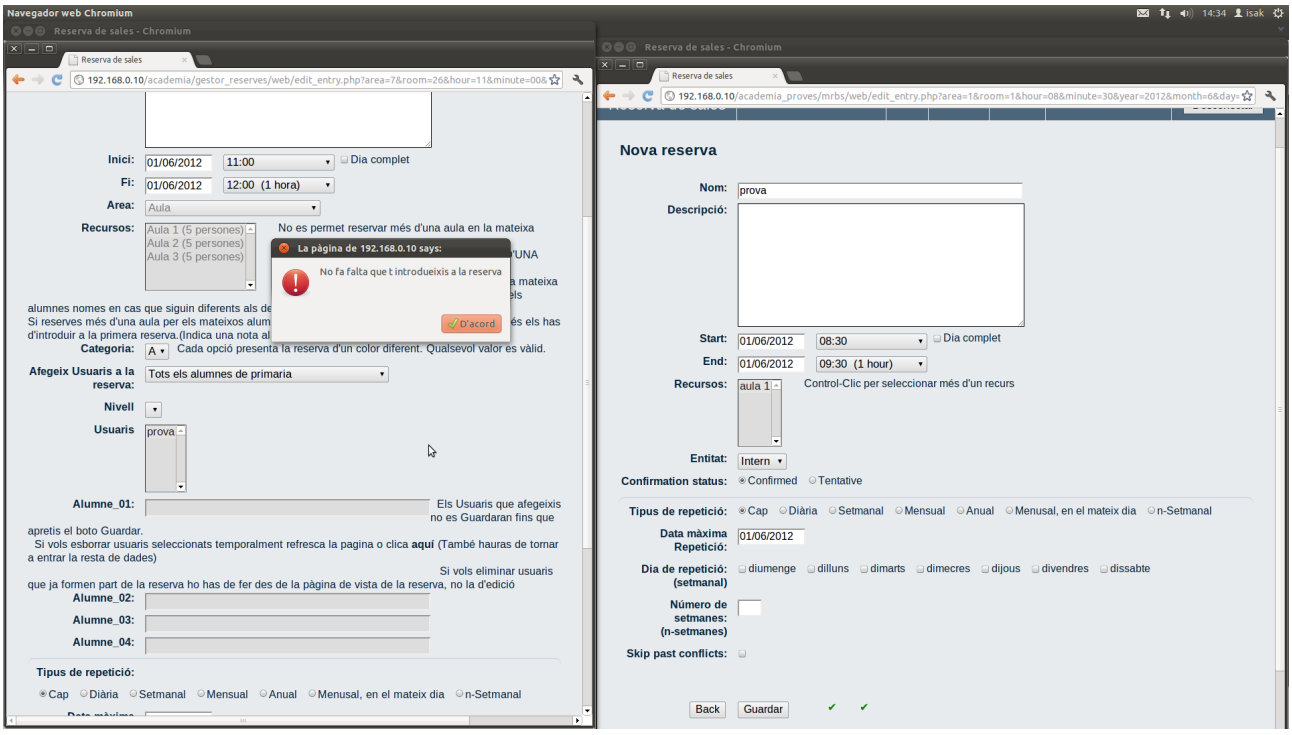

Figura 22: Demostració. Reserva d'aula amb capacitat per a 5 persones (Professor + 4 Alumnes)

Posteriorment, creem la reserva amb un altre alumne i comparem ambdues vistes de reserva.

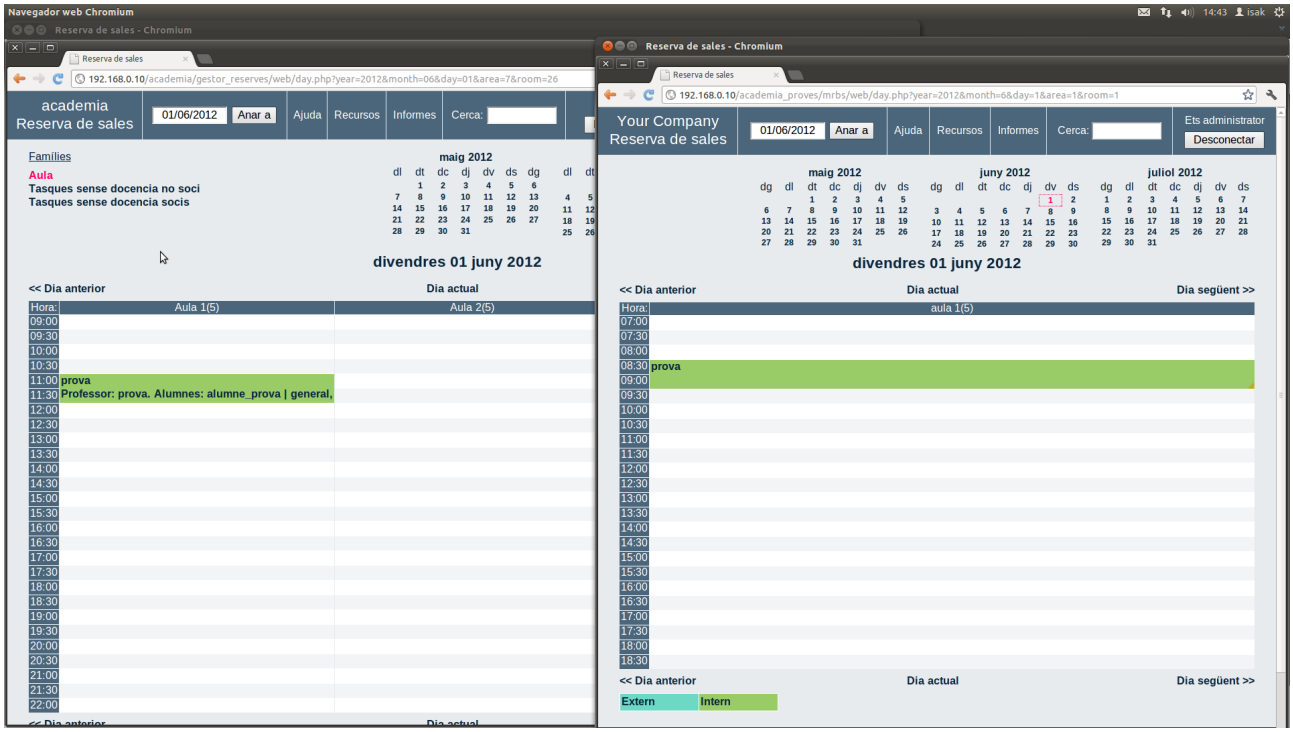

Figura 23: Demostració. Vista de reserva d'aula

Si accedim a editar la reserva podem observar com a la interfície 1 es pot eliminar l'usuari de la reserva o afegir-ne de nous ,tant des del mode vista com des del d'edició.

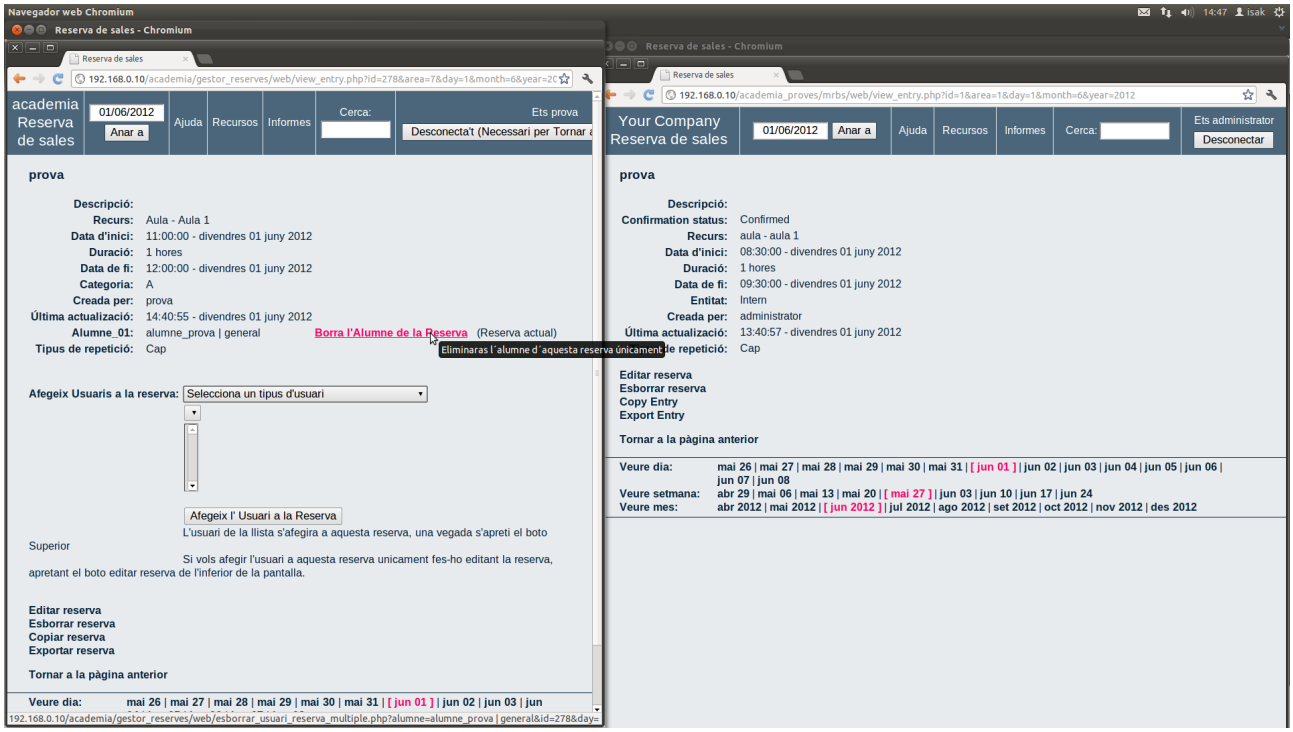

Figura 24: Demostració. Vista de reserva d'aula

Posteriorment observem com en el cas de les reserves múltiples el proces es similar, poden eliminar o afegir usuaris a les reserves actual i futures de la reserva múltiple o únicament a la reserva actual.

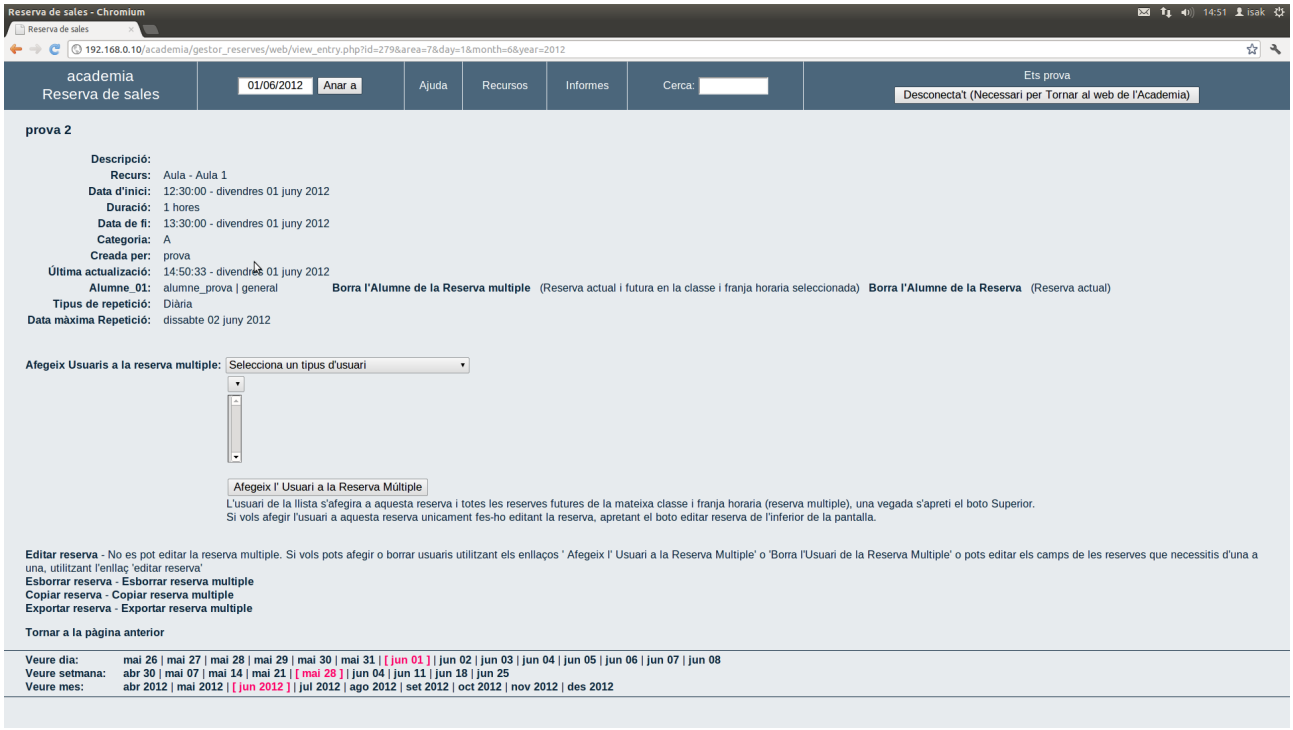

Figura 25: Demostració. Vista de reserva múltiple d'aula amb codi projecte

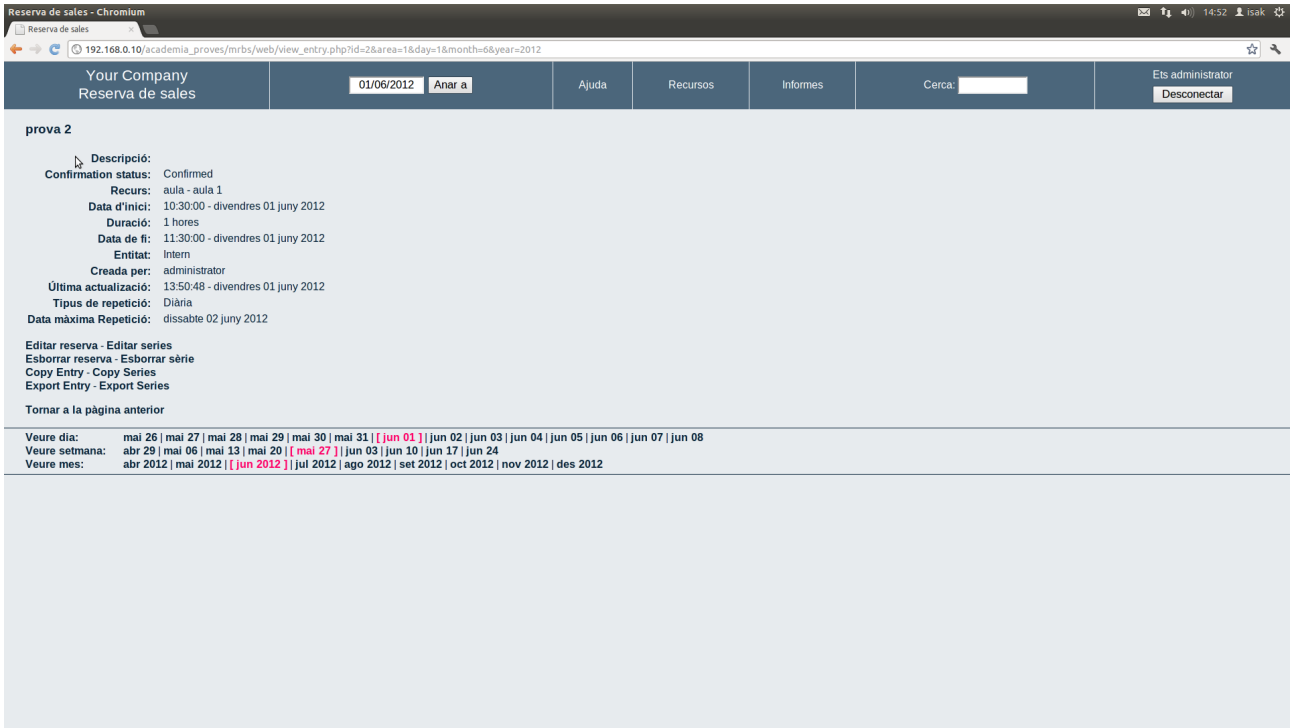

Figura 26: Demostració. Vista de reserva múltiple d'aula amb codi no projecte

Si, una vegada desconnectats, volem accedir al gestor de reserves de la interfície 1, inserint la direcció del gestor o amb un usuari que no té permisos d'accés veiem com el gestor no ens permet ni fer l'accés, una de les diverses funcionalitats de seguretat que s'han implementat.

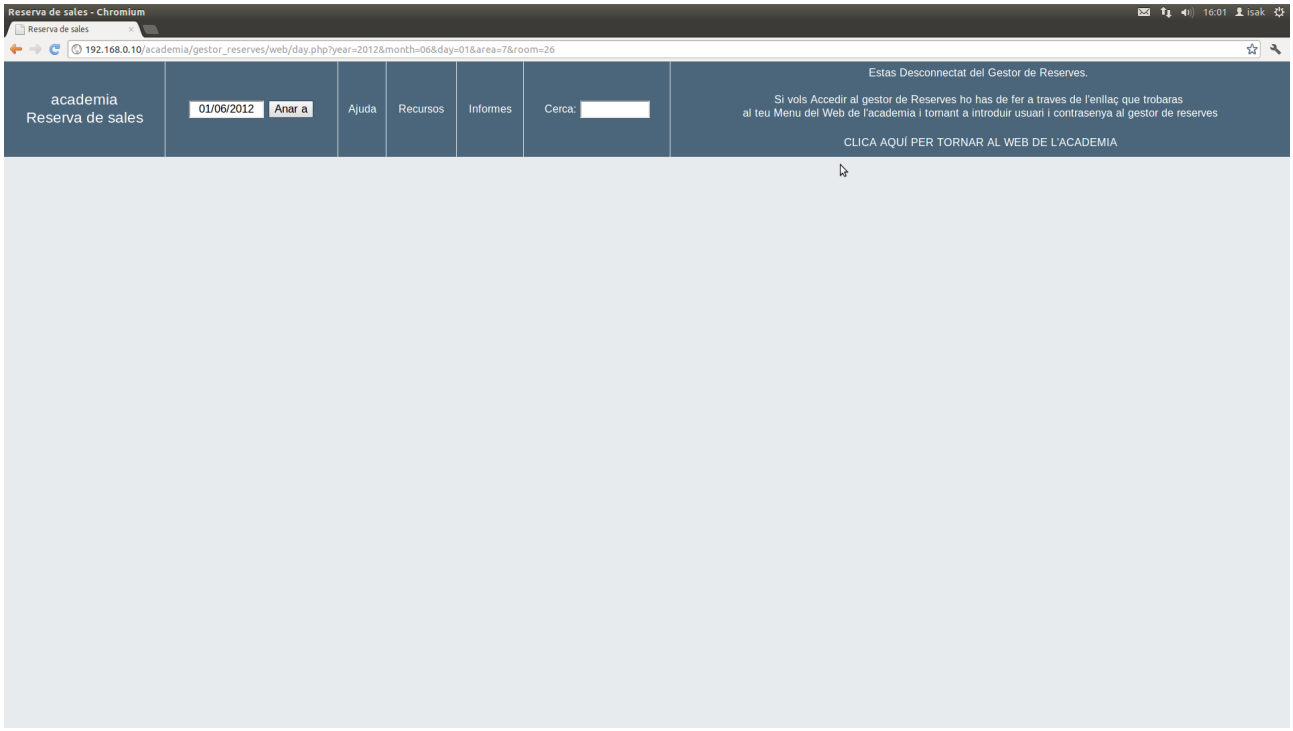

Figura 27: Demostració. Accés gestor reserves directe des de l'explorador.

Retornant a la pàgina de la interfíce de Drupal podem veure com des del menú d'usuari hi ha un enllaç al campus virtual, des d'on l'usuari de prova, tindrà accés al curs de *primària*, tal i com indica el seu rol. En el cas de la interfície 2 l'accés es realitza manualment des de l'explorador i veiem com l'usuari de prova no ha estat creat, amb el que s'accedeix utilitzant un usuari d'administració que hem creat per a realitzar la prova.

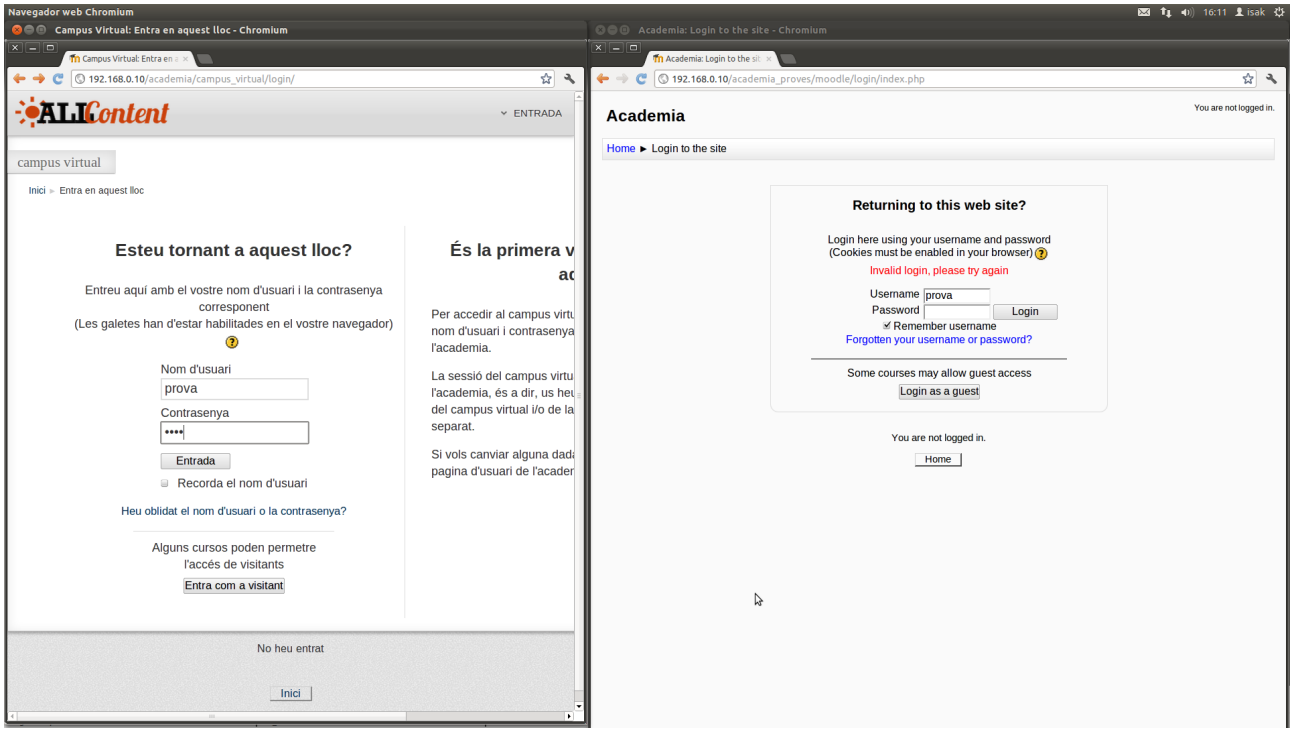

Figura 28: Demostració. Accés Campus Virtual.

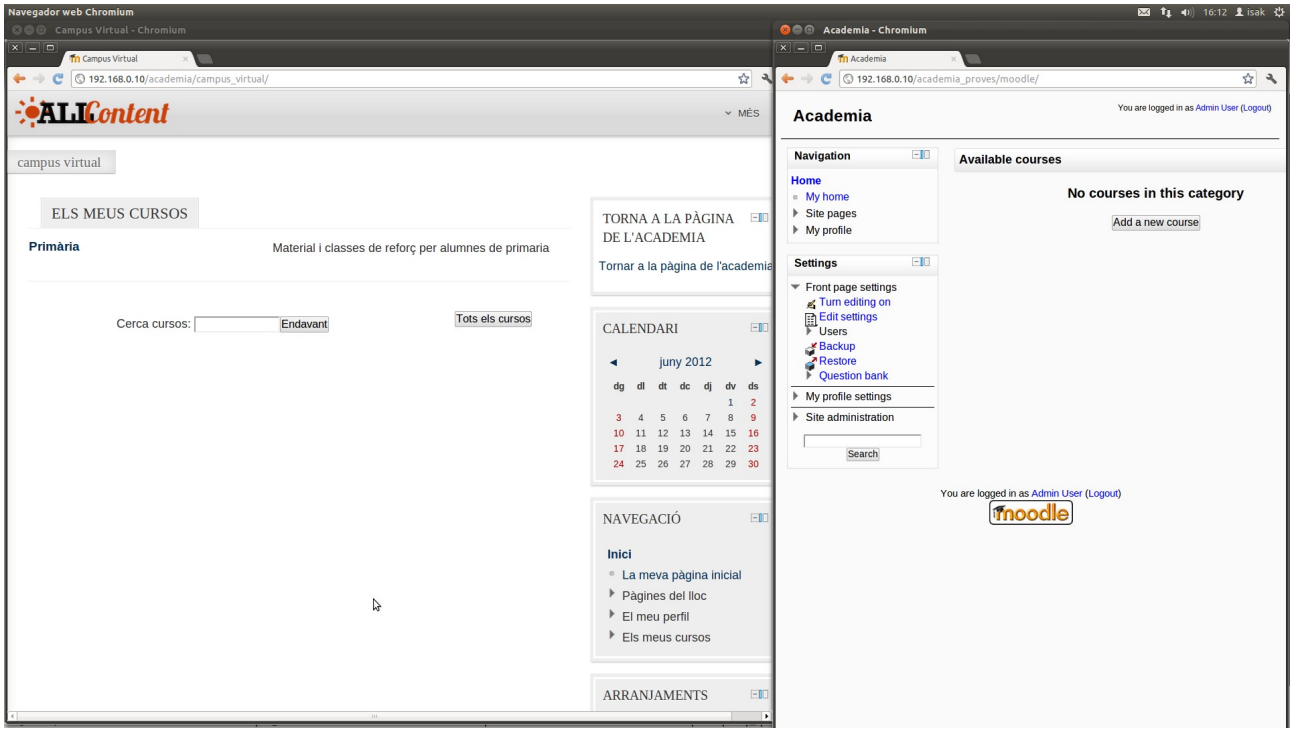

Figura 29: Demostració. Accés Campus Virtual (2).

Finalment tornem a la interfície de Drupal a través de l'enllaç del bloc superior del campus Virtual, des d'on podem tornar al gestor de reserves i realitzar les reserves que requerim.

Capítol 6: Conclusions i línees de futur

6.1 Validesa general del projecte i punts clau de la viabilitat

El projecte es pot considerar viable, partint de què el desenvolupament de Mòduls i blocs per a ser integrats en Moodle i Drupal per part d'altres usuaris i empreses del sector han obtingut bons resultats. La integració d'MRBS a Moodle i Drupal com a sistema gestor de Reserves també s'ha realitzat per a versions anteriors d'ambdós sistemes amb èxit. D'altra banda, tot i que MRBS no permet la inclusió de mòduls o blocs que operin amb el codi font, aquest es distribueix amb facilitats per a la customització per part de l'usuari.

Un altre punt a tenir en compte és que la integració es realitza en tres sistemes existents basats en un programari fiable i segur. La solució global per al servei requerit, inclosos els costos, amb els recursos disponibles és coherent. Pel que fa al plaç temporal, en cas de no poder finalitzar el projecte, o que aquest s'implementés de forma que es pogués optimitzar fàcilment (com podria ser el cas d'alguna de les parts del projecte), es podria allargar el temps del projecte, contractar un altre desenvolupador o posar-lo en mans de la comunitat, per a finalitzar i/o millorar el projecte (per exemple millorar l'estructuració de codi).

A continuació s'indiquen els factors claus de la viabilitat:

- Desenvolupament i integració de tres sistemes, Moodle, Drupal i MRBS implementat en empreses del sector i que ha obtingut bons resultats.

- Les funcionalitats de Moodle, i sobretot de Drupal s'integren de forma separada al nucli, de manera que es poden detectar i corregir errors fàcilment. En el cas d'MRBS, tot i incloure's directament sobre el nucli aquest disposa de facilitats per a customitzar el codi.

- Detecció i correcció d'errors del projecte amb ajuda de la comunitat.

- Utilització de programari lliure; fiable, estable i amb un ampli entorn de desenvolupament i suport.

- Facilitat en la formació dels usuaris (desenvolupadors futurs o usuaris genèrics) gràcies a la documentació especifica generada per aquest projecte i la documentació associada que ja existeix en els projectes de Drupal, Moodle i MRBS.

6.2 Conclusions Finals

A la conclusió d'aquest projecte s'ha comprovat com tot i que les hores dedicades per a dur-lo a terme són moltes més de les que està planificat en el pla d'estudis de l'assignatura, les sinergies que el projecte té amb la creació d'una cooperativa de formació (comentat a l'apartat de motivacions), ha fet que aquest es pugui dur a terme amb èxit dins la planificació temporal estimada inicialment. Val a dir però, que la majoria del codi desenvolupat es podria optimitzar, en cas de disposar de major temps, per tal que els processos i el propi codi fos menor.

També s'ha pogut observar com, tot i que el disseny del projecte es basa en la integració d'un sistema gestor de reserves sobre dos sistemes, Drupal i Moodle, sent Drupal el sistema que s'encarrega de mantenir i gestionar els usuaris, aquesta integració es podria realitzar fàcilment sobre Moodle (o algun altre sistema) fent petites modificacions, gràcies a la separació de funcionalitats, i a la estructuració similar (tant a nivell de dades com de funcions) que tenen els sistemes web basats en programari lliure.

De la mateixa manera es pot concloure que la creació de mòduls o blocs per a sistemes de programari lliure, es pot dur a terme sense disposar d'un gran coneixement de programació, com es el cas de l'estudiant que ha dut a terme el projecte, tot i que això impliqui desenvolupar el codi d'una forma no del tot òptima, cosa que es pot solucionar fàcilment posant-lo en mans de la comunitat d'experts.

Finalment valorar que el projecte ha servit per a permetre la integració del sistema gestor de reserves MRBS i el campus Virtual de Moodle a la pagina web creada mitjançant Drupal per a la cooperativa de formació de la qual l'estudiant en forma part, i de l'alliberació de part del codi a la comunitat, com es el cas del modul MRBS per a Drupal, on a la data de la redacció d'aquestes conclusions ja hi ha 8 llocs web que l'estan utilitzant.

Annexos

Annex 1: Mòdul MRBS de Drupal

Fitxer mrbs.info:

```
; $Id$
name = MRBS
description = Creates permissions for MRBS 1.4.8 integration and MRBS inc files for SSO
package = MRBS
dependencies[] = user
core = 7.x
files[] = mrbs.install
files[] = mrbs.module
```

Fitxer mrbs.module:

```
<?php

/**
 * @file
 * This file is required for all modules
 */
```

Fitxer mrbs.install:

```
<?php
/**
 * @file
 * MRBS install file, add just MRBS Roles
 */

/**
 * Implements hook_install to add 3 MRBS Roles()
 */
function mrbs_install() {
        // create any roles we will be using

        _add_role('mrbs admin');
        _add_role('mrbs bookings');
        _add_role('mrbs view');
}
/**
```

```
 * Implements hook_uninstall()
 */
function mrbs_uninstall() {
	// remove any roles we created
	user_role_delete('mrbs admin');
	user_role_delete('mrbs bookings');
	user_role_delete('mrbs view');
}

function _add_role($machine_name, $weight = 0)  {
	$role = new stdClass();
	$role->name = $machine_name;
	$role->weight = $weight;
	user_role_save($role);
}
```

Fitxer README.txt:

```
// $Id: README.txt,v 1.1 2009/02/14 16:18:32 wesku Exp $

Drupal - MRBS integration

This very simple module offers single sign-on integration with MRBS ( http://mrbs.sourceforge.net/ ) and Drupal. The module uses Drupal user accounts and permissions with MRBS. Currently the module has only been tested when installed directly in a subdirectory on Drupal root directory, but it should be quite easy to modify it for other installations.

Installation
- Install MRBS in a subdirectory on your Drupal installation
- Install Drupal module and enable it
- Copy auth_drupal.inc from mrbs_inc subdirectory to your mrbs install directory
- Change config.inc.php:
  - $auth["type"] = "drupal";
- Add your Drupal settings in config.inc.php (the same as Drupal settings.php) not MRBS settings
  - $auth['drupal']['db_system'] = 'drupal_db_system';
  - $auth['drupal']['db_host'] = 'drupal_db_host';
  - $auth['drupal']['db_username'] = 'drupal_db_username';
  - $auth['drupal']['db_password'] = 'drupal_db_password';
  - $auth['drupal']['db_name'] = 'drupal_db_name';

That's it, MRBS users and should now be managed by Drupal.

OPTIONAL
----------
- Copy session_drupal.inc from mrbs_inc subdirectory to your mrbs install directory
- Change config.inc.php:
  - $auth["session"] = "drupal";
```

- Enable Drupal Menu Token Module.
- Set [current-user:name] as a token of the menu that links to MRBS in Drupal to send the name of the user as a user GET variable.
- Links to MRBS admin.php passing the current user name as the user GET variable (for example your_drupal_site/mrbs/web/admin.php?user=[current-user:name]).

The user log in MRBS is still needed.
With this the session is still managed by MRBS php schema, but with more secure and better drupal integration

Fitxer LICENSCE.txt:

El contingut d'aquest fitxer es pot trobar a http://www.gnu.org/licenses/gpl-2.0.html

Fitxer auth_drupal.inc:

```
<?php
/**************************************************************************
*
*   File name       auth_drupal.inc
*
*   Description     Authenticate users from Drupal7.
*
*   Notes           To use this authentication scheme, set in config.inc.php:
*         $auth["type"]  = "drupal";
*
*   History
*    Available in the source control system
*
**************************************************************************/
// $Id$

/* authValidateUser($user, $pass)
 *
 * Checks if the specified username/password pair are valid
 *
 * $user  - The user name
 * $pass  - The password
 *
 * Returns:
 *   0        - The pair are invalid or do not exist
 *   non-zero - The pair are valid
 */

//for drupal7 method autentication functions password.inc and bootstrap.inc are needed. The route we use is when we install MRBS as a Drupal subdirectory (drupal/mrbs/web)
```

```
include ("../../includes/password.inc");
include ("../../includes/bootstrap.inc");

function authValidateUser($user, $pass)
{
 global $auth;

 $retval = 0;

 $user = strtolower($user);

 if (empty($auth['drupal']['db_system']))
 {
  $auth['drupal']['db_system'] = 'mysql';
 }

 $conn = sql_connect($auth['drupal']['db_system'],
                     $auth['drupal']['db_host'],
                     $auth['drupal']['db_username'],
                     $auth['drupal']['db_password'],
                     $auth['drupal']['db_name']);

 $user = addslashes($user);
 //we use default D7 table names here
 $query = "SELECT pass FROM users WHERE name ='$user'";

 $r = sql_query($query, $conn);

 if ($r && (sql_count($r, $conn) == 1)) // force a unique match
 {
  $row = sql_row($r, 0, $conn);
  $password_to_compare->pass = $row[0];
  if (user_check_password($pass,$password_to_compare))
   {
    $retval = 1;
   }
  }
 return $retval;
}

/* authGetUserLevel($user)
 *
 * Determines the users access level
 *
 * $user - The user name
 *
 * Returns:
```

```
 *   The users access level
 */
function authGetUserLevel($user)
{
  global $auth;

  $conn = sql_connect($auth['drupal']['db_system'],
                      $auth['drupal']['db_host'],
                      $auth['drupal']['db_username'],
                      $auth['drupal']['db_password'],
                      $auth['drupal']['db_name']);
  $user = addslashes($user);
  //we use default D7 table names here to take rid of every mrbs role
  $query = "SELECT rid FROM role WHERE name = 'mrbs admin'";
  $r = sql_query($query, $conn);
  if ($r && (sql_count($r, $conn) == 1)) // force a unique match
  {
    $row = sql_row($r, 0, $conn);
    $mrbs_admin_rid = $row[0];
  }
  $query_2 = "SELECT rid FROM role WHERE name = 'mrbs bookings'";
  $r_2 = sql_query($query_2, $conn);
  if ($r_2 && (sql_count($r_2, $conn) == 1)) // force a unique match
  {
    $row_2 = sql_row($r_2, 0, $conn);
    $mrbs_bookings_rid = $row_2[0];
  }
  $query_3 = "SELECT rid FROM role WHERE name = 'mrbs view'";
  $r_3 = sql_query($query_3, $conn);
  if ($r_3 && (sql_count($r_3, $conn) == 1)) // force a unique match
  {
    $row_3 = sql_row($r_3, 0, $conn);
    $mrbs_view_rid = $row_3[0];
  }
  //we use default D7 table names here to take uid of the user
  $query_4 = "SELECT uid FROM users WHERE name = '$user'";
  $r_4 = sql_query($query_4, $conn);
  $row_4 = sql_row($r_4, 0, $conn);
  $user_uid = $row_4[0];

  //we use default D7 table names here to take all the rid of the user in array
  $query_5 = "SELECT rid FROM users_roles WHERE uid = '$user_uid'";
  $r_5 = sql_query($query_5, $conn);
  for ($i = 0; ($row_5 = sql_row($r_5, $i, $conn)); $i++)
  {
    $rid_array[] = $row_5[0];
  }
  //we compare the rid of the user to check if have mrbs roles
```

```
 if(in_array($mrbs_admin_rid, array_values($rid_array))){
 //admin permisions
 return 2;
 }
 elseif(in_array($mrbs_bookings_rid, array_values($rid_array))){
 //booking permisions
 return 1;
 }
 elseif(in_array($mrbs_view_rid, array_values($rid_array))){
 //view permisions
 return 0;
 }
 else{
 //can't access to MRBS
 return -1;
 }
}

?>
```

Fitxer session_drupal.inc:

```
<?php
/**************************************************************************\
*                                                                          *
*   File name      session_drupal.inc                                      *
*                                                                          *
*   Description    Use PHP built-in sessions handling with some D7 checkings *
*                                                                          *
*   Notes          To use this session mechanism, set in config.inc.php:   *
*                  $auth["session"]  = "drupal";                           *
*                                                                          *
*   History                                                                *
*                  2009/02/14 Vesa Palmu Created this file                 *
*                  Remaining history in ChangeLog and CVS logs             *
*                                                                          *
\**************************************************************************/

// $Id$

global $PHP_SELF;

// Get non-standard form variables
$Action = get_form_var('Action', 'string');
$NewUserName = get_form_var('NewUserName', 'string');
$NewUserPassword = get_form_var('NewUserPassword', 'string');
$TargetURL = get_form_var('TargetURL', 'string');
```

```
$returl = get_form_var('returl', 'string');

if (isset($cookie_path_override))
{
  $cookie_path = $cookie_path_override;
}
else
{
  $cookie_path = $PHP_SELF;
  // Strip off everything after the last '/' in $PHP_SELF
  $cookie_path = preg_replace('/[^\/]*$/', '', $cookie_path);
}

global $auth;

if (!isset($auth["session_php"]["session_expire_time"]))
{
  // Default to the behaviour of previous versions of MRBS, use only
  // session cookies - no persistent cookie.
  $auth["session_php"]["session_expire_time"] = 0;
}

session_name("MRBS_SESSID");  // call before session_set_cookie_params() - see PHP manual
session_set_cookie_params($auth["session_php"]["session_expire_time"],
                          $cookie_path);
session_start();

/*
  Target of the form with sets the URL argument "Action=SetName".
  Will eventually return to URL argument "TargetURL=whatever".
*/
if (isset($Action) && ($Action == "SetName"))
{
  /* First make sure the password is valid */
  if ($NewUserName == "")
  {

    // Unset the session variables
    if (isset($_SESSION))
    {
      $_SESSION = array();
    }
    else
    {
      global $HTTP_SESSION_VARS;
      $HTTP_SESSION_VARS = array();
    }
  }
```

```
  else
  {
   if (!authValidateUser($NewUserName, $NewUserPassword))
   {
    print_header(0, 0, 0, 0, "");
    echo "<p>".get_vocab('unknown_user')."</p>\n";
    printLoginForm($TargetURL);
    exit();
   }

   if (isset($_SESSION))
   {
    $_SESSION["UserName"] = $NewUserName;
   }
   else
   {
    global $HTTP_SESSION_VARS;
    $HTTP_SESSION_VARS["UserName"] = $NewUserName;
   }
  }
  // preserve the original $HTTP_REFERER by sending it as a GET parameter
  if (!empty($returl))
  {
   // check to see whether there's a query string already
   if (strpos($TargetURL, '?') === false)
   {
    $TargetURL .= "?returl=" . urlencode($returl);
   }
   else
   {
    $TargetURL .= "&returl=" . urlencode($returl);
   }
  }
  header ("Location: $TargetURL"); /* Redirect browser to initial page */
  /* Note HTTP 1.1 mandates an absolute URL. Most modern browsers support relative URLs,
    which allows to work around problems with DNS inconsistencies in the server name.
    Anyway, if the browser cannot redirect automatically, the manual link below will work. */
  print_header(0, 0, 0, 0, "");
  echo "<br>\n";
  echo "<p>Please click <a href=\"".htmlspecialchars($TargetURL)."\">here</a> if you're not redirected automatically
to the page you requested.</p>\n";

  // Print footer and exit
  print_footer(TRUE);
}

/*
  Display the login form. Used by two routines below.
```

```
  Will eventually return to $TargetURL.
*/
function printLoginForm($TargetURL)
{
  global $PHP_SELF, $HTTP_REFERER;
  global $returl;
?>
<form class="form_general" id="logon" method="post" action="<?php echo htmlspecialchars(basename($PHP_SELF))
?>">
  <fieldset>
  <legend><?php echo get_vocab("please_login") ?></legend>
    <div>
      <label for="NewUserName"><?php echo get_vocab("users.name") ?>:</label>
      <input type="text" id="NewUserName" name="NewUserName">
    </div>
    <div>
      <label for="NewUserPassword"><?php echo get_vocab("users.password") ?>:</label>
      <input type="password" id="NewUserPassword" name="NewUserPassword">
    </div>
    <?php
    // We need to preserve the original calling page, so that it's there when we eventually get
    // to the TargetURL (especially if that's edit_entry.php).  If this is the first time through then $HTTP_REFERER holds
    // the original caller.    If this is the second time through we will have stored it in $returl.
    if (!isset($returl))
    {
      $returl = isset($HTTP_REFERER) ? $HTTP_REFERER : "";
    }
    echo "<input type=\"hidden\" name=\"returl\" value=\"" . htmlspecialchars($returl) . "\">\n";
    ?>
    <input type="hidden" name="TargetURL" value="<?php echo htmlspecialchars($TargetURL) ?>">
    <input type="hidden" name="Action" value="SetName">
    <div id="logon_submit">
      <input class="submit" type="submit" value=" <?php echo get_vocab('login') ?> ">
    </div>
  </fieldset>
</form>
<?php
  echo "</div>";  // Close of the contents div
//modified
?>
 <script language="JavaScript">
 var drupal_user_name = ["<?php echo $_GET["user"]; ?>"];
 document.getElementById('NewUserName').readOnly = true;
 document.getElementById('NewUserName').value = drupal_user_name;
 </script>
<?php

  // Print footer and exit
```

```
  print_footer(TRUE);
}

/*
  Target of the form with sets the URL argument "Action=QueryName".
  Will eventually return to URL argument "TargetURL=whatever".
*/
if (isset($Action) && ($Action == "QueryName"))
{
  print_header(0, 0, 0, 0, "");
  printLoginForm($TargetURL);
  exit();
}

/* authGet()
 *
 * Request the user name/password
 *
 * Returns: Nothing
 */
function authGet()
{
  global $PHP_SELF, $QUERY_STRING;

  print_header(0, 0, 0, 0, "");

  echo "<p>".get_vocab("norights")."</p>\n";

  $TargetURL = basename($PHP_SELF);
  if (isset($QUERY_STRING))
  {
    $TargetURL = $TargetURL . "?" . $QUERY_STRING;
  }
  printLoginForm($TargetURL);

  exit();
}

function getUserName()
{
  if (isset($_SESSION) && isset($_SESSION["UserName"]) && ($_SESSION["UserName"] != ""))
  {
    return $_SESSION["UserName"];
  }
  else
  {
    global $HTTP_SESSION_VARS;
    if (isset($HTTP_SESSION_VARS["UserName"]) && ($HTTP_SESSION_VARS["UserName"] != ""))
```

```
    {
      return $HTTP_SESSION_VARS["UserName"];
    }
  }
}

// Print the logon entry on the top banner.
function PrintLogonBox()
{

  global $PHP_SELF, $QUERY_STRING, $user_list_link, $day, $month, $year;

  $TargetURL = basename($PHP_SELF);
  if (isset($url_base) && ($url_base != ""))
  {
    $TargetURL = $url_base . '/' . $TargetURL;
  }
  if (isset($QUERY_STRING))
  {
    $TargetURL = $TargetURL . "?" . $QUERY_STRING;
  }
  $user=getUserName();
  if (isset($user))
  {
    //check if this user have Drupal7 permisions to view or edit MRBS. If not disable username vars to avoid connect with
not desired user
    if (authGetUserLevel($user) == -1){
      ?> Acces Denegat. <br> <a href="../..">Tornar a l'academia</a>
      <?php
     $user="";
     $_SESSION["UserName"]="";
     $HTTP_SESSION_VARS["UserName"]="";
     exit;
    }
    // words 'you are xxxx' becomes a link to the
    // report page with only entries created by xxx. Past entries are not
    // displayed but this can be changed
    $search_string = "report.php?from_day=$day&amp;from_month=$month&amp;".
      "from_year=$year&amp;to_day=1&amp;to_month=12&amp;to_year=2030&amp;areamatch=&amp;".
      "roommatch=&amp;namematch=&amp;descrmatch=&amp;summarize=1&amp;sortby=r&amp;display=d&amp;".
      "sumby=d&amp;creatormatch=".urlencode($user); ?>

            <a href="<?php echo "$search_string\" title=\""
        . get_vocab('show_my_entries') . "\">" . get_vocab('you_are')." "
        . htmlspecialchars($user) ?></a>
            <form method="post" action="admin.php">
              <div>
                <input type="hidden" name="TargetURL" value="<?php echo htmlspecialchars($TargetURL) ?>">
```

```
                <input type="hidden" name="Action" value="SetName">
                <input type="hidden" name="NewUserName" value="">
                <input type="hidden" name="NewUserPassword" value="">
                <input type="submit" value=" <?php echo get_vocab('logoff') ?> ">
              </div>
            </form>
<?php
  }
  else
  {
          //check if this user have Drupal7 permisions to view or edit MRBS, not get the permisions. To get the permisions must log in MRBS with the same drupal account
          if(isset($_GET['user'])){
          $user = $_GET['user'];
          }
          if (authGetUserLevel($user) == -1){
            ?> Estas Desconnectat del Gestor de Reserves. <br><br> Si vols Accedir al gestor de Reserves ho has de fer a través de l'enllaç que trobaras <br> al teu Menu del Web de l'academia i tornant a introduir usuari i contrasenya al gestor de reserves<br><br><a href="../..">CLICA AQUÍ PER TORNAR AL WEB DE L'ACADEMIA</a>
        <?php
            exit;
      }
      else{
      ?> <a href="../..">TORNAR AL WEB DE L'ACADEMIA</a>
      <?php
      }
  }
  if (isset($user_list_link))
  {
    print "<a href=\"$user_list_link\">" . get_vocab('user_list') . "</a>\n";
  }
}
?>
```

Annex 2: Arxius modificats nucli MRBS

Fitxer edit_entry.php (El codi afegit es pot veure sota el tag //modified):

```
<?php
// $Id: edit_entry.php 2225 2011-12-31 13:24:49Z cimorrison $

// If you want to add some extra columns to the entry and repeat tables to
// record extra details about bookings then you can do so and this page should
// automatically recognise them and handle them.    NOTE: if you add a column to
```

```
// the entry table you must add an identical column to the repeat table.
//
// At the moment support is limited to the following column types:
//
// MySQL        PostgreSQL            Form input type
// -----        ----------            ---------------
// bigint       bigint                text
// int          integer               text
// mediumint                          text
// smallint     smallint              checkbox
// tinyint                            checkbox
// text         text                  textarea
// tinytext                           textarea
//              character varying     textarea
// varchar(n)   character varying(n)  text/textarea, depending on the value of n
//              character             text
// char(n)      character(n)          text/textarea, depending on the value of n
//
// NOTE 1: For char(n) and varchar(n) fields, a text input will be presented if
// n is less than or equal to $text_input_max, otherwise a textarea box will be
// presented.
//
// NOTE 2: PostgreSQL booleans are not supported, due to difficulties in
// handling the fields in a database independent way (a PostgreSQL boolean
// will return a PHP boolean type when read by a PHP query, whereas a MySQL
// tinyint returns an int).   In order to have a boolean field in the room
// table you should use a smallint in PostgreSQL or a smallint or a tinyint
// in MySQL.
//
// You can put a description of the column that will be used as the label in
// the form in the appropriate lang file(s) using the tag 'entry.[columnname]'.
// (Note that it is not necessary to add a 'repeat.[columnname]' tag.   The
// entry tag is sufficient.)
//
// For example if you want to add a column recording the number of participants
// you could add a column to the entry and repeat tables called 'participants'
// of type int.  Then in the appropriate lang file(s) you would add the line
//
// $vocab["entry.participants"] = "Participants";  // or appropriate translation
//
// If MRBS can't find an entry for the field in the lang file, then it will use
// the fieldname, eg 'coffee_machine'.


require_once "defaultincludes.inc";
require_once "mrbs_sql.inc";

// Generate a time or period selector starting with $first and ending with $last.
```

```
// $time is a full Unix timestamp and is the current value.  The selector returns
// the start time in seconds since the beginning of the day for the start of that slot.
// Note that these are nominal seconds and do not take account of any DST changes that
// may have happened earlier in the day.  (It's this way because we don't know what day
// it is as that's controlled by the date selector - and we can't assume that we have
// JavaScript enabled to go and read it)
//
// The $display_none parameter sets the display style of the <select> to "none"
// The $disabled parameter will disable the input and also generate a hidden input, provided
// that $display_none is FALSE.  (This prevents multiple inputs of the same name)
function genSlotSelector($area, $prefix, $first, $last, $time, $display_none=FALSE, $disabled=FALSE)
{
  global $periods;

  $html = '';
  // Get the settings for this area.   Note that the variables below are
  // local variables, not globals.
  $enable_periods = $area['enable_periods'];
  $resolution = ($enable_periods) ? 60 : $area['resolution'];
  // Check that $resolution is positive to avoid an infinite loop below.
  // (Shouldn't be possible, but just in case ...)
  if (empty($resolution) || ($resolution < 0))
  {
    fatal_error(FALSE, "Internal error - resolution is NULL or <= 0");
  }

  // Get the current hour and minute and convert it into nominal (ie ignoring any
  // DST effects) seconds since the start of the day
  $date = getdate($time);
  $current_t = (($date['hours'] * 60) + $date['minutes']) * 60;

  if ($enable_periods)
  {
    $base = 12*60*60;  // The start of the first period of the day
  }
  else
  {
    $format = hour_min_format();
  }
  $html .= "<select" .
          (($display_none) ? " style=\"display: none\"" : "") .
          // If $display_none or $disabled are set then we'll also disable the select so
          // that there is only one select passing through the variable to the handler
          (($display_none || $disabled) ? " disabled=\"disabled\"" : "") .
          // and if $disabled is set, give the element a class so that the JavaScript
          // knows to keep it disabled
          (($disabled) ? " class=\"keep_disabled\"" : "") .
```

```
        " id=\"${prefix}seconds${area['id']}\" name=\"${prefix}seconds\"
onChange=\"adjustSlotSelectors(this.form)\">\n";
  for ($t = $first; $t <= $last; $t = $t + $resolution)
  {
    // The date used below is completely arbitrary.   All that matters is that it
    // is a day that does not contain a DST boundary.   (We need a real date so that
    // we can use strftime to get an hour and minute formatted according to the locale)
    $timestamp = $t + mktime(0, 0, 0, 1, 1, 2000);
    $slot_string = ($enable_periods) ? $periods[intval(($t-$base)/60)] : utf8_strftime($format, $timestamp);
    $html .= "<option value=\"$t\"";
    $html .= ($t == $current_t) ? " selected=\"selected\"" : "";
    $html .= ">$slot_string</option>\n";
  }
  $html .= "</select>\n";
  // Add in a hidden input if the select is disabled but displayed
  if ($disabled && !$display_none)
  {
    $html .= "<input type=\"hidden\" name=\"${prefix}seconds\" value=\"$current_t\">\n";
  }

  echo $html;
}


function create_field_entry_name($disabled=FALSE)
{
  global $name, $select_options, $maxlength, $is_mandatory_field;

  echo "<div id=\"div_name\">\n";
  $label_text = get_vocab("namebooker") . ":";
  if (!empty($select_options['entry.name']))
  {
    generate_select($label_text, 'name', $name, $select_options['entry.name'],
                    $is_mandatory_field['entry.name'], $disabled);
  }
  else
  {
    generate_input($label_text, 'name', $name, $disabled, $maxlength['entry.name']);
  }
  echo "</div>\n";
}


function create_field_entry_description($disabled=FALSE)
{
  global $description, $select_options, $is_mandatory_field;

  echo "<div id=\"div_description\">\n";
```

```
  $label_text = get_vocab("fulldescription");
  if (!empty($select_options['entry.description']))
  {
    generate_select($label_text, 'description', $description, $select_options['entry.description'],
                    $is_mandatory_field['entry.description'], $disabled);
  }
  else
  {
    generate_textarea($label_text, 'description', $description, $disabled);
  }
  echo "</div>\n";
}


function create_field_entry_start_date($disabled=FALSE)
{
  global $start_time, $areas, $area_id, $periods, $default_duration_all_day, $id, $drag;
  global $periods, $is_admin;

  echo "<div id=\"div_start_date\">\n";
  echo "<label>" . get_vocab("start") . ":</label>\n";
  $date = getdate($start_time);
  gendateselector("start_", $date['mday'], $date['mon'], $date['year'], '', $disabled);
  // If we're using periods the booking model is slightly different:
  // you're allowed to specify the last period as your first period.
  // This is why we don't substract the resolution

  foreach ($areas as $a)
  {
    if ($a['enable_periods'])
    {
      $a['resolution'] = 60;
      $first = 12*60*60;
      // If we're using periods we just go to the beginning of the last slot
      $last = $first + ((count($periods) - 1) * $a['resolution']);
    }
    else
    {
      $first = (($a['morningstarts'] * 60) + $a['morningstarts_minutes']) * 60;
      $last = (($a['eveningends'] * 60) + $a['eveningends_minutes']) * 60;
      $last = $last + $a['resolution'];
    }
    $start_last = ($a['enable_periods']) ? $last : $last - $a['resolution'];
    $display_none = ($a['id'] != $area_id);
    genSlotSelector($a, "start_", $first, $start_last, $start_time, $display_none, $disabled);

    echo "<div class=\"group\">\n";
    echo "<div id=\"ad{$a['id']}\"".($display_none ? " style=\"display: none\" " : "") .">\n";
```

```
    // We don't show the all day checkbox if it's going to result in bookings that
    // contravene the policy - ie if max_duration is enabled and an all day booking
    // would be longer than the maximum duration allowed
    $show_all_day = $is_admin || !$a['max_duration_enabled'] ||
                    ( ($a['enable_periods'] && ($a['max_duration_periods'] >= count($periods))) ||
                      (!$a['enable_periods'] && ($a['max_duration_secs'] >= ($last - $first))) );
    echo "<input id=\"all_day{$a['id']}\" class=\"checkbox\"" .
        // If this is an existing booking that we are editing or copying, then we do
        // not want the default duration applied
        (($default_duration_all_day && !isset($id) && !$drag) ? " checked=\"checked\"" : "") .
        " name=\"all_day\" type=\"checkbox\" value=\"yes\" onclick=\"OnAllDayClick(this)\"".
        ($show_all_day? "" : " style=\"display: none;\" ").
        // If $display_none or $disabled are set then we'll also disable the select so
        // that there is only one select passing through the variable to the handler
        (($display_none || $disabled) ? " disabled=\"disabled\"" : "") .
        // and if $disabled is set, give the element a class so that the JavaScript
        // knows to keep it disabled
        (($disabled) ? " class=\"keep_disabled\"" : "") .
        ">\n";
    if($show_all_day)
    {
      echo "<label for=\"all_day\">" . get_vocab("all_day") . "</label>\n";
    }
    echo "</div>\n";
    echo "</div>\n";
  }
  echo "</div>\n";
}


function create_field_entry_end_date($disabled=FALSE)
{
  global $end_time, $areas, $area_id, $periods, $multiday_allowed;

  echo "<div id=\"div_end_date\">\n";
  echo "<label>" . get_vocab("end") . ":</label>\n";
  $date = getdate($end_time);
  // Don't show the end date selector if multiday is not allowed
  echo "<div" . (($multiday_allowed) ? '' : " style=\"visibility: hidden\"") . ">\n";
  gendateselector("end_", $date['mday'], $date['mon'], $date['year'], '', $disabled);
  echo "</div>\n";
  // If we're using periods the booking model is slightly different,
  // so subtract one period because the "end" period is actually the beginning
  // of the last period booked
  foreach ($areas as $a)
  {
    if ($a['enable_periods'])
    {
```

```
    $a['resolution'] = 60;
    $first = 12*60*60;
    // If we're using periods we just go to the beginning of the last slot
    $last = $first + ((count($periods) - 1) * $a['resolution']);
   }
   else
   {
    $first = (($a['morningstarts'] * 60) + $a['morningstarts_minutes']) * 60;
    $last = (($a['eveningends'] * 60) + $a['eveningends_minutes']) * 60;
    $last = $last + $a['resolution'];
   }
   $end_value = ($a['enable_periods']) ? $end_time - $a['resolution'] : $end_time;
   $display_none = ($a['id'] != $area_id);
   genSlotSelector($a, "end_", $first, $last, $end_value, $display_none, $disabled);
  }
  echo "<span id=\"end_time_error\" class=\"error\"></span>\n";
  echo "</div>\n";
}


function create_field_entry_areas($disabled=TRUE)
{
  global $areas, $area_id, $rooms;

  echo "<div id=\"div_areas\">\n";
  echo "</div>\n";
  // if there is more than one area then give the option
  // to choose areas.
  if (count($areas) > 1)
  {
    ?>
     <script type="text/javascript">
     //<![CDATA[

     var area = <?php echo $area_id ?>;

     function changeRooms( formObj )
     {
       areasObj = eval( "formObj.area" );

       area = areasObj[areasObj.selectedIndex].value;
       roomsObj = eval( "formObj.elements['rooms']" );

       // remove all entries
       roomsNum = roomsObj.length;
       for (i=(roomsNum-1); i >= 0; i--)
       {
         roomsObj.options[i] = null;
```

```
}
// add entries based on area selected
switch (area){
  <?php
  foreach ($areas as $a)
  {
    print "case \"" . $a['id'] . "\":\n";
    // get rooms for this area
    $i = 0;
    foreach ($rooms as $r)
    {
      if ($r['area_id'] == $a['id'])
      {
        print "roomsObj.options[$i] = new Option(\"" . escape_js($r['room_name']) . "\"," . $r['id'] . ");\n";
        $i++;
      }
    }
    // select the first entry by default to ensure
    // that one room is selected to begin with
    if ($i > 0)  // but only do this if there is a room
    {
      print "roomsObj.options[0].selected = true;\n";
    }
    print "break;\n";
  }
  ?>
} //switch

<?php
// Replace the start and end selectors with those for the new area
// (1) We set the display for the old elements to "none" and the new
// elements to "block".   (2) We also need to disable the old selectors and
// enable the new ones: they all have the same name, so we only want
// one passed through with the form.  (3) We take a note of the currently
// selected start and end values so that we can have a go at finding a
// similar time/period in the new area. (4) We also take a note of the old
// area id because we'll need that when trying to match up slots: it only
// makes sense to match up slots if both old and new area used the same
// mode (periods/times).

// For the "all day" checkbox, the process is slightly different.  This
// is because the checkboxes themselves are visible or not depending on
// the time restrictions for that particular area. (1) We set the display
// for the old *container* element to "none" and the new elements to
// "block".  (2) We disable the old checkboxes and enable the new ones for
// the same reasons as above.  (3) We copy the value of the old check box
// to the new check box
?>
```

```
  var oldStartId = "start_seconds" + currentArea;
  var oldEndId = "end_seconds" + currentArea;
  var newStartId = "start_seconds" + area;
  var newEndId = "end_seconds" + area;
  var oldAllDayId = "ad" + currentArea;
  var newAllDayId = "ad" + area;
  var oldAreaStartValue = formObj[oldStartId].options[formObj[oldStartId].selectedIndex].value;
  var oldAreaEndValue = formObj[oldEndId].options[formObj[oldEndId].selectedIndex].value;
  $("#" + oldStartId).hide()
                     .attr('disabled', 'disabled');
  $("#" + oldEndId).hide()
                   .attr('disabled', 'disabled');
  $("#" + newStartId).show()
                     .removeAttr('disabled');
  $("#" + newEndId).show()
                   .removeAttr('disabled');
                   +       $("#" + oldAllDayId).hide();
  $("#" + newAllDayId).show();
  if($("#all_day" + currentArea).attr('checked') == 'checked')
  {
    $("#all_day" + area).attr('checked', 'checked').removeAttr('disabled');
  }
  else
  {
    $("#all_day" + area).removeAttr('checked').removeAttr('disabled');
  }
  $("#all_day" + currentArea).removeAttr('disabled');
  var oldArea = currentArea;
  currentArea = area;
  prevStartValue = undefined;
  adjustSlotSelectors(formObj, oldArea, oldAreaStartValue, oldAreaEndValue);
}

// Create area selector, only if we have Javascript
var div_areas = document.getElementById('div_areas');
// First of all create a label and insert it into the <div>
var area_label = document.createElement('label');
var area_label_text = document.createTextNode('<?php echo get_vocab("area") ?>:');
area_label.appendChild(area_label_text);
area_label.setAttribute('for', 'area');
div_areas.appendChild(area_label);
// Now give it a select box
var area_select = document.createElement('select');
area_select.setAttribute('id', 'area');
area_select.setAttribute('name', 'area');
area_select.onchange = function(){changeRooms(this.form)}; // setAttribute doesn't work for onChange with IE6
// populated with options
var option;
```

```
    var option_text
    <?php
    // go through the areas and create the options
    foreach ($areas as $a)
    {
      ?>
      option = document.createElement('option');
      option.value = <?php echo $a['id'] ?>;
      option_text = document.createTextNode('<?php echo escape_js($a['area_name']) ?>');
      <?php
      if ($a['id'] == $area_id)
      {
        ?>
        option.selected = true;
        <?php
      }
      ?>
      option.appendChild(option_text);
      area_select.appendChild(option);
      <?php
    }
    ?>
    // insert the <select> which we've just assembled into the <div>
    div_areas.appendChild(area_select);

    <?php
    if ($disabled)
    {
      // If the field is disabled we need to disable the select box and
      // add in a hidden input containing the value
      ?>
      $('#area').attr('disabled', 'disabled');
      $('<input>').attr('type', 'hidden')
                  .attr('name', 'area')
                  .val('<?php echo $area_id ?>')
                  .appendTo('#div_areas');
      <?php
    }
    ?>

    //]]>
    </script>


    <?php
  } // if count($areas)
}
```

```
function create_field_entry_rooms($disabled=TRUE)
{
  global $rooms, $multiroom_allowed, $room_id, $area_id, $selected_rooms;

  echo "<div id=\"div_rooms\">\n";
  echo "<label for=\"rooms\">" . get_vocab("rooms") . ":</label>\n";
  echo "<div class=\"group\">\n";
  echo "<select id=\"rooms\" name=\"rooms[]\"" .
    (($multiroom_allowed) ? " multiple=\"multiple\"" : "") .
    (($disabled) ? " disabled=\"disabled\"" : "") .
    " size=\"5\">\n";
  // $selected_rooms will be populated if we've come from a drag selection
  if (empty($selected_rooms))
  {
    $selected_rooms = array($room_id);
  }
  foreach ($rooms as $r)
  {
    if ($r['area_id'] == $area_id)
    {
      $is_selected = in_array($r['id'], $selected_rooms);
      $selected = ($is_selected) ? "selected=\"selected\"" : "";
      echo "<option $selected value=\"" . $r['id'] . "\">" . htmlspecialchars($r['room_name']) . " (" .
htmlspecialchars($r['capacity']) . " persones) </option>\n"; //modified
    }
  }
  echo "</select>\n";
  // No point telling them how to select multiple rooms if the input
  // is disabled
  if ($multiroom_allowed && !$disabled)
  {
    echo "<span>" . get_vocab("ctrl_click") . "</span>\n";
  }
  echo "</div>\n";
  if ($disabled)
  {
    foreach ($selected_rooms as $selected_room)
    {
      echo "<input type=\"hidden\" name=\"rooms[]\" value=\"$selected_room\">\n";
    }
  }
  //modified
  if ($disabled==TRUE){
    echo "No es permet reservar més d'una aula en la mateixa reserva.<br>COM HO POTS FER PER RESERVAR MÉS
D'UNA AULA AL MATEIX DIA I MATEIXA HORA?<br>Si vols reservar més d'una aula el mateix dia i la mateixa franja
horaria fes una nova reserva i introdueix els alumnes nomes en cas que siguin diferents als de la primera
```

```
reserva.<br>Si reserves més d'una aula per els mateixos alumnes al mateix dia i la mateixa franja horaria, només els
has d'introduir a la primera reserva.(Indica una nota al camp de descripció de la segona reserva).";
  }
  echo "</div>\n";
}


function create_field_entry_type($disabled=FALSE)
{
  global $booking_types, $type;
  echo "<div id=\"div_type\">\n";
  echo "<label for=\"type\">" . get_vocab("type") . ":</label>\n";
  echo "<select id=\"type\" name=\"type\"" .
       (($disabled) ? " disabled=\"disabled\"" : "") .
       ">\n";
  foreach ($booking_types as $key)
  {
    echo "<option value=\"$key\"" . (($type == $key) ? " selected=\"selected\"" : "") .
">".get_type_vocab($key)."</option>\n";
  }
  echo "</select>\n     Cada opció presenta la reserva d'un color diferent. Qualsevol valor es
vàlid.";
  if ($disabled)
  {
    echo "<input type=\"hidden\" name=\"type\" value=\"$type\">\n";
  }
  echo "</div>\n";
}


function create_field_entry_confirmation_status($disabled=FALSE)
{
  global $confirmation_enabled, $confirmed;

  // Confirmation status
  if ($confirmation_enabled)
  {
    echo "<div id=\"div_confirmation_status\">\n";
    echo "<label>" . get_vocab("confirmation_status") . ":</label>\n";
    echo "<div class=\"group\">\n";
    echo "<label><input class=\"radio\" name=\"confirmed\" type=\"radio\" value=\"1\"" .
      (($confirmed) ? " checked=\"checked\"" : "") .
      (($disabled) ? " disabled=\"disabled\"" : "") .
      ">" . get_vocab("confirmed") . "</label>\n";
    echo "<label><input class=\"radio\" name=\"confirmed\" type=\"radio\" value=\"0\"" .
      (($confirmed) ? "" : " checked=\"checked\"") .
      (($disabled) ? " disabled=\"disabled\"" : "") .
      ">" . get_vocab("tentative") . "</label>\n";
```

```
    echo "</div>\n";
    if ($disabled)
    {
      echo "<input type=\"hidden\" name=\"confirmed\" value=\"" .
           (($confirmed) ? "1" : "0") .
           "\">\n";
    }
    echo "</div>\n";
  }
}


function create_field_entry_privacy_status($disabled=FALSE)
{
  global $private_enabled, $private, $private_mandatory;

  // Privacy status
  if ($private_enabled)
  {
    // No need to pass through a hidden variable if disabled because the handler will sort it out
    echo "<div id=\"div_privacy_status\">\n";
    echo "<label>" . get_vocab("privacy_status") . ":</label>\n";
    echo "<div class=\"group\">\n";
    echo "<label><input class=\"radio\" name=\"private\" type=\"radio\" value=\"0\"" .
      (($private) ? "" : " checked=\"checked\"") .
      (($private_mandatory || $disabled) ? " disabled=\"disabled\"" : "") .
      ">" . get_vocab("public") . "</label>\n";
    echo "<label><input class=\"radio\" name=\"private\" type=\"radio\" value=\"1\"" .
      (($private) ? " checked=\"checked\"" : "") .
      (($private_mandatory || $disabled) ? " disabled=\"disabled\"" : "") .
      ">" . get_vocab("private") . "</label>\n";
    echo "</div>\n";
    if ($disabled)
    {
      echo "<input type=\"hidden\" name=\"private\" value=\"" .
           (($private) ? "1" : "0") .
           "\">\n";
    }
    echo "</div>\n";
  }
}

function create_field_entry_custom_field($field, $key, $disabled=FALSE)
{
  global $custom_fields, $tbl_entry, $select_options;
  global $is_mandatory_field, $text_input_max;
  global $room_id, $rooms, $area_id;
```

```
foreach ($rooms as $r)
{
 if ($r['id'] == $room_id){
 $capacitat_aula= $r['capacity'];
 }
}
$var_name = VAR_PREFIX . $key;
$value = $custom_fields[$key];
$label_text = get_loc_field_name($tbl_entry, $key) . ":";
//modified
$user_number = substr($label_text, 7, 2);
$user_number = (int)$user_number;
if ($capacitat_aula > $user_number && $area_id == 7){
  echo "<div id=\"div_alumnes\">\n";
}
//no show more users than the capacity permits
else {
  echo "<div id=\"div_alumnes\" style=\"display:none\">\n";
}
// Output a checkbox if it's a boolean or integer <= 2 bytes (which we will
// assume are intended to be booleans)
if (($field['nature'] == 'boolean') ||
 (($field['nature'] == 'integer') && isset($field['length']) && ($field['length'] <= 2)) )
{
 echo "<label for=\"$var_name\">$label_text</label>\n";
 echo "<input type=\"checkbox\" class=\"checkbox\" " .
   "id=\"$var_name\" name=\"$var_name\" value=\"1\" " .
   ((!empty($value)) ? " checked=\"checked\"" : "") .
   (($disabled) ? " disabled=\"disabled\"" : "") .
   ">\n";
}
// Output a select box if they want one
elseif (!empty($select_options["entry.$key"]))
{
 $mandatory = (array_key_exists("entry.$key", $is_mandatory_field) &&
   $is_mandatory_field["entry.$key"]) ? true : false;
 generate_select($label_text, $var_name, $value,
   $select_options["entry.$key"], $mandatory, $disabled);
}
// Output a textarea if it's a character string longer than the limit for a
// text input
elseif (($field['nature'] == 'character') && isset($field['length']) && ($field['length'] > $text_input_max))
{
 generate_textarea($label_text, $var_name, $value, $disabled);
}
// Otherwise output a text input
else
{
```

```
    generate_input($label_text, $var_name, $value, $disabled);
  }
  if ($disabled)
  {
    echo "<input type=\"hidden\" name=\"$var_name\" value=\"$value\">\n";
  }
  if ($var_name == "f_Alumne_01"){echo "  Els Usuaris que afegeixis no es Guardaran fins que apretis el
boto Guardar.<br>  Si vols esborrar usuaris seleccionats temporalment refresca la pagina o clica <a
href=''>aquí</a> (També hauras de tornar a entrar la resta de
dades)<br>               &
nbsp;                &
nbsp;                &
nbsp;                &
nbsp;                &
nbsp;                &
nbsp;                &
nbsp;     Si vols eliminar usuaris que ja formen part de la reserva ho has de fer des de
la pàgina de vista, no la d'edició</div>\n";}
  else {echo "</div>\n";}
//modified
?>
 <script language="JavaScript">
 /*set the user inputs to textonly and remove function*/
 var var_name = ["<?php echo $var_name; ?>"];
 document.getElementById(var_name).readOnly = true;
 </script>
<?php
}

//modified
//function to add drupal users in the booking
function create_field_entry_add_drupal_users()
{
  global $name_all_users, $name_alumn_estiu, $name_alumn_matricula_grup, $name_alumn_matricula_particular,
$name_alumn_pack_particular, $name_alumn_pack_grup, $name_alumn_proves_acces_academia, $name_selectivitat,
$name_centre_client, $name_responsable_centre_client, $name_professor, $name_professor_soci,
$name_alumn_offline, $name_universitat, $name_primaria, $name_eso, $name_batxillerat;
//get all the drupal users name that have the roles that we want and put it in array

  global $auth, $area_id;

  if ($area_id == 7){

              $conn = sql_connect($auth['drupal']['db_system'],
                                              $auth['drupal']['db_host'],
                                              $auth['drupal']['db_username'],
                                              $auth['drupal']['db_password'],
                                              $auth['drupal']['db_name']);
```

```
//take all users name
    $query = "SELECT name FROM users WHERE uid != 0 ORDER BY name";
    $r = sql_query($query, $conn);
    for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
    {
    $name_all_users[] = $row[0];
    }
    ?>
    <script language="JavaScript">
    var name_all_users = ["<?php echo join("\", \"", $name_all_users); ?>"];
    </script>
    <?php

    //take the users that have the role id 12 (alumne estiu)
    $query = "SELECT uid FROM users_roles WHERE rid = '12'";
    $r = sql_query($query, $conn);
    for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
    {
    $query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
    $r_2 = sql_query($query_2, $conn);
    $row_2 = sql_row($r_2, 0, $conn);
    $name_alumn_estiu[] = $row_2[0];
    }
    sort($name_alumn_estiu);
    ?>
    <script language="JavaScript">
    var name_alumn_estiu = ["<?php echo join("\", \"", $name_alumn_estiu); ?>"];
    </script>
    <?php

    //take the users that have the role id 10 (alumne matricula grup)
    $query = "SELECT uid FROM users_roles WHERE rid = '10'";
    $r = sql_query($query, $conn);
    for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
    {
    $query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
    $r_2 = sql_query($query_2, $conn);
    $row_2 = sql_row($r_2, 0, $conn);
    $name_alumn_matricula_grup[] = $row_2[0];
    }
    sort($name_alumn_matricula_grup);
    ?>
    <script language="JavaScript">
    var name_alumn_matricula_grup = ["<?php echo join("\", \"", $name_alumn_matricula_grup); ?>"];
    </script>
    <?php
```

```
//take the users that have the role id 19 (alumne matricula particular)
$query = "SELECT uid FROM users_roles WHERE rid = '19'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_matricula_particular[] = $row_2[0];
}
sort($name_alumn_matricula_particular);
?>
<script language="JavaScript">
var name_alumn_matricula_particular = ["<?php echo join("\", \"", $name_alumn_matricula_particular); ?>"];
</script>
<?php

//take the users that have the role id 9 (alumne pack hores grup)
$query = "SELECT uid FROM users_roles WHERE rid = '9'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_pack_particular[] = $row_2[0];
}
sort($name_alumn_pack_particular);
?>
<script language="JavaScript">
var name_alumn_pack_particular = ["<?php echo join("\", \"", $name_alumn_pack_particular); ?>"];
</script>
<?php

//take the users that have the role id 18 (alumne pack hores particular)
$query = "SELECT uid FROM users_roles WHERE rid = '18'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_pack_grup[] = $row_2[0];
}
sort($name_alumn_pack_grup);
?>
<script language="JavaScript">
var name_alumn_pack_grup = ["<?php echo join("\", \"", $name_alumn_pack_grup); ?>"];
```

```
</script>
<?php

//take the users that have the role id 13 (alumne proves acces academia)
$query = "SELECT uid FROM users_roles WHERE rid = '13'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_proves_acces_academia[] = $row_2[0];
}
sort($name_alumn_proves_acces_academia);
?>
<script language="JavaScript">
var name_alumn_proves_acces_academia = ["<?php echo join("\", \"",
$name_alumn_proves_acces_academia); ?>"];
</script>
<?php

//take the users that have the role id 16 (alumne selectivitat)
$query = "SELECT uid FROM users_roles WHERE rid = '16'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_selectivitat[] = $row_2[0];
}
sort($name_selectivitat);
?>
<script language="JavaScript">
var name_selectivitat = ["<?php echo join("\", \"", $name_selectivitat); ?>"];
</script>
<?php

//take the users that have the role id 14 (alumne centre client)
$query = "SELECT uid FROM users_roles WHERE rid = '14'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_centre_client[] = $row_2[0];
}
```

```
sort($name_centre_client);
?>
<script language="JavaScript">
var name_centre_client = ["<?php echo join("\", \"", $name_centre_client); ?>"];
</script>
<?php

//take the users that have the role id 15 (responsable centre client)
$query = "SELECT uid FROM users_roles WHERE rid = '15'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_responsable_centre_client[] = $row_2[0];
}
sort($name_responsable_centre_client);
?>
<script language="JavaScript">
var name_responsable_centre_client = ["<?php echo join("\", \"", $name_responsable_centre_client); ?>"];
</script>
<?php

//take the users that have the role id 20 (professor)
$query = "SELECT uid FROM users_roles WHERE rid = '20'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_professor[] = $row_2[0];
}
sort($name_professor);
?>
<script language="JavaScript">
var name_professor = ["<?php echo join("\", \"", $name_professor); ?>"];
</script>
<?php

//take the users that have the role id 4 (professor soci)
$query = "SELECT uid FROM users_roles WHERE rid = '4'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
```

```
$row_2 = sql_row($r_2, 0, $conn);
$name_professor_soci[] = $row_2[0];
}
sort($name_professor_soci);
?>
<script language="JavaScript">
var name_professor_soci = ["<?php echo join("\", \"", $name_professor_soci); ?>"];
</script>
<?php

//take the users that have the role id 17 (alumne offline)
$query = "SELECT uid FROM users_roles WHERE rid = '17'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_offline[] = $row_2[0];
}
sort($name_alumn_offline);
?>
<script language="JavaScript">
var name_alumn_offline = ["<?php echo join("\", \"", $name_alumn_offline); ?>"];
</script>
<?php

//take the users that have the role id 40 (universitat)
$query = "SELECT uid FROM users_roles WHERE rid = '40'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_universitat[] = $row_2[0];
}
sort($name_universitat);
?>
<script language="JavaScript">
var name_universitat = ["<?php echo join("\", \"", $name_universitat); ?>"];
</script>
<?php

//take the users that have the role id 38 (primaria)
$query = "SELECT uid FROM users_roles WHERE rid = '38'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
```

```
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_primaria[] = $row_2[0];
}
sort($name_primaria);
?>
<script language="JavaScript">
var name_primaria = ["<?php echo join("\", \"", $name_primaria); ?>"];
</script>
<?php

//take the users that have the role id 39 (eso)
$query = "SELECT uid FROM users_roles WHERE rid = '39'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_eso[] = $row_2[0];
}
sort($name_eso);
?>
<script language="JavaScript">
var name_eso = ["<?php echo join("\", \"", $name_eso); ?>"];
</script>
<?php

//take the users that have the role id 37 (batxillerat)
$query = "SELECT uid FROM users_roles WHERE rid = '37'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_batxillerat[] = $row_2[0];
}
sort($name_batxillerat);
?>
<script language="JavaScript">
var name_batxillerat = ["<?php echo join("\", \"", $name_batxillerat); ?>"];
</script>
<?php

//take the users that have the role id 41 (GES)
```

```
$query = "SELECT uid FROM users_roles WHERE rid = '41'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_ges[] = $row_2[0];
}
sort($name_ges);
?>
<script language="JavaScript">
var name_ges = ["<?php echo join("\", \"", $name_ges); ?>"];
</script>
<?php

//take the users that have the role id 42 (Grau Mitja)
$query = "SELECT uid FROM users_roles WHERE rid = '42'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_grau_mitja[] = $row_2[0];
}
sort($name_grau_mitja);
?>
<script language="JavaScript">
var name_grau_mitja = ["<?php echo join("\", \"", $name_grau_mitja); ?>"];
</script>
<?php

//take the users that have the role id 43 (Grau Superior)
$query = "SELECT uid FROM users_roles WHERE rid = '43'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_grau_superior[] = $row_2[0];
}
sort($name_grau_superior);
?>
<script language="JavaScript">
var name_grau_superior = ["<?php echo join("\", \"", $name_grau_superior); ?>"];
</script>
```

```
<?php

//take the users that have the role id 44 (Universitat Majors 25)
$query = "SELECT uid FROM users_roles WHERE rid = '44'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_universitat_majors_25[] = $row_2[0];
}
sort($name_universitat_majors_25);
?>
<script language="JavaScript">
var name_universitat_majors_25 = ["<?php echo join("\", \"", $name_universitat_majors_25); ?>"];
</script>
<?php

//take the users that have the role id 45 (Tallers_i_cursos_varis)
$query = "SELECT uid FROM users_roles WHERE rid = '45'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_tallers[] = $row_2[0];
}
sort($name_tallers);
?>
<script language="JavaScript">
var name_tallers = ["<?php echo join("\", \"", $name_tallers); ?>"];
</script>

<?php
echo "<div id=\"div_add_drupal_users\">\n";
echo "<label for=\"add_drupal_users\">" . get_vocab("add_drupal_users") . ":</label>\n";
echo "<div class=\"group\">\n";

//use a javascript function to creat a subselect for the roles that have subselect options only
?>
<script language="JavaScript">
drupal_client_level_roles = new Array();
drupal_client_level_roles[0] = new Array("alumn_pack_grup", "Tots");
drupal_client_level_roles[1] = new Array("alumn_pack_particular", "Tots");
drupal_client_level_roles[2] = new Array("alumn_matricula_particular", "Tots");
drupal_client_level_roles[3] = new Array("alumn_matricula_grup", "Tots");
```

```
    drupal_client_level_roles[4] = new Array("alumn_matricula_particular", "universitat");
    drupal_client_level_roles[5] = new Array("alumn_matricula_particular", "batxillerat");
    drupal_client_level_roles[6] = new Array("alumn_matricula_particular", "eso");
    drupal_client_level_roles[7] = new Array("alumn_matricula_particular", "primaria");
    drupal_client_level_roles[8] = new Array("alumn_pack_particular", "universitat");
    drupal_client_level_roles[9] = new Array("alumn_pack_particular", "batxillerat");
    drupal_client_level_roles[10] = new Array("alumn_pack_particular", "eso");
    drupal_client_level_roles[11] = new Array("alumn_pack_particular", "primaria");
    drupal_client_level_roles[12] = new Array("alumn_pack_grup", "batxillerat");
    drupal_client_level_roles[13] = new Array("alumn_pack_grup", "eso");
    drupal_client_level_roles[14] = new Array("alumn_pack_grup", "primaria");
    drupal_client_level_roles[15] = new Array("alumn_matricula_grup", "batxillerat");
    drupal_client_level_roles[16] = new Array("alumn_matricula_grup", "eso");
    drupal_client_level_roles[17] = new Array("alumn_matricula_grup", "primaria");
drupal_client_level_roles[18] = new Array("alumn_centre_client", "Tots");
drupal_client_level_roles[19] = new Array("alumn_centre_client", "ges");
drupal_client_level_roles[20] = new Array("alumn_centre_client", "grau mitja");
drupal_client_level_roles[21] = new Array("alumn_centre_client", "grau superior");
drupal_client_level_roles[22] = new Array("alumn_centre_client", "universitat majors");
drupal_client_level_roles[22] = new Array("alumn_centre_client", "tallers");
drupal_client_level_roles[23] = new Array("alumn_proves_acces_academia", "Tots");
drupal_client_level_roles[24] = new Array("alumn_proves_acces_academia", "ges");
drupal_client_level_roles[25] = new Array("alumn_proves_acces_academia", "grau mitja");
drupal_client_level_roles[26] = new Array("alumn_proves_acces_academia", "grau superior");
drupal_client_level_roles[27] = new Array("alumn_proves_acces_academia", "universitat majors");


    function DrupalUsersAndSubRoles(role)
    {
     var opRoles = document.getElementById("f_sub_roles").options
     var opUsers = document.getElementById("f_filtered_users").options
     var noOptions = true;
     while(opRoles.length > 0) {
      opRoles.remove(0);
     }

     while(opUsers.length > 0) {
      opUsers.remove(0);
     }

     if(role.length > 0) {

      for( var i=0; i < drupal_client_level_roles.length; i++ ) {
            if(drupal_client_level_roles[i][0] == role) {
                if (noOptions == true) {
                opRoles.add(new Option("Selecciona el nivell de l'usuari", ""));
                }
             opRoles.add(new Option(drupal_client_level_roles[i][1], drupal_client_level_roles[i][1]));
```

```
        noOptions = false;
      }
}
if (noOptions == true) {
      if (role == "general"){
       for( var j=0; j < name_all_users.length; j++ ) {
         opUsers.add(new Option(name_all_users[j], name_all_users[j]));
       }
      }
      if (role == "alumn_estiu"){
       for( var j=0; j < name_alumn_estiu.length; j++ ) {
         opUsers.add(new Option(name_alumn_estiu[j], name_alumn_estiu[j]));
       }
      }
      if (role == "alumn_selectivitat"){
       for( var j=0; j < name_alumn_selectivitat.length; j++ ) {
         opUsers.add(new Option(name_alumn_selectivitat[j], name_alumn_selectivitat[j]));
       }
      }
      if (role == "responsable_centre_client"){
       for( var j=0; j < name_responsable_centre_client.length; j++ ) {
         opUsers.add(new Option(name_responsable_centre_client[j], name_responsable_centre_client[j]));
       }
      }
      if (role == "professor"){
       for( var j=0; j < name_professor.length; j++ ) {
         opUsers.add(new Option(name_professor[j], name_professor[j]));
       }
      }
      if (role == "professor_soci"){
       for( var j=0; j < name_professor_soci.length; j++ ) {
         opUsers.add(new Option(name_professor_soci[j], name_professor_soci[j]));
       }
      }
      if (role == "alumn_offline"){
       for( var j=0; j < name_alumn_offline.length; j++ ) {
         opUsers.add(new Option(name_alumn_offline[j], name_alumn_offline[j]));
       }
      }
      if (role == "primaria"){
       for( var j=0; j < name_primaria.length; j++ ) {
         opUsers.add(new Option(name_primaria[j], name_primaria[j]));
       }
      }
      if (role == "eso"){
       for( var j=0; j < name_eso.length; j++ ) {
         opUsers.add(new Option(name_eso[j], name_eso[j]));
       }
```

```
                }
                if (role == "batxillerat"){
                  for( var j=0; j < name_batxillerat.length; j++ ) {
                    opUsers.add(new Option(name_batxillerat[j], name_batxillerat[j]));
                  }
                }
                if (role == "universitat"){
                  for( var j=0; j < name_universitat.length; j++ ) {
                    opUsers.add(new Option(name_universitat[j], name_universitat[j]));
                  }
                }
                if (role == "tallers"){
                  for( var j=0; j < name_tallers.length; j++ ) {
                    opUsers.add(new Option(name_tallers[j], name_tallers[j]));
                  }
                }
              }
             }
            }
            function DrupalUsersForSubRoles(subrole)
            {
              //matricula grup selected index = 3, mat part = 4 pack grup =5, pack part =6, proves acces academia = 7,
centre extern = 9
              var selectedRole = document.getElementById("f_drupal_roles").selectedIndex
                var opUsers = document.getElementById("f_filtered_users").options
                while(opUsers.length > 0) {
                  opUsers.remove(0);
                }
                if (subrole == "Tots") {
                        if (selectedRole == 3){
                          for( var j=0; j < name_alumn_matricula_grup.length; j++ ) {
                                opUsers.add(new Option(name_alumn_matricula_grup[j],
name_alumn_matricula_grup[j]));
                          }
                        }
                        if (selectedRole == 4){
                          for( var j=0; j < name_alumn_matricula_particular.length; j++ ) {
                                opUsers.add(new Option(name_alumn_matricula_particular[j],
name_alumn_matricula_particular[j]));
                          }
                        }
                        if (selectedRole == 5){
                          for( var j=0; j < name_alumn_pack_grup.length; j++ ) {
                                opUsers.add(new Option(name_alumn_pack_grup[j], name_alumn_pack_grup[j]));
                          }
                        }
                        if (selectedRole == 6){
                          for( var j=0; j < name_alumn_pack_particular.length; j++ ) {
```

```
                                opUsers.add(new Option(name_alumn_pack_particular[j],
name_alumn_pack_particular[j]));
                        }
                    }
                    if (selectedRole == 7){
                        for( var j=0; j < name_alumn_proves_acces_academia.length; j++ ) {
                                opUsers.add(new Option(name_alumn_proves_acces_academia[j],
name_alumn_proves_acces_academia[j]));
                        }
                    }
                    if (selectedRole == 9){
                        for( var j=0; j < name_centre_client.length; j++ ) {
                                opUsers.add(new Option(name_centre_client[j], name_centre_client[j]));
                        }
                    }
                }
                if (subrole == "ges") {
                        if (selectedRole == 7){
                            var name_alumn_proves_academia_ges = new Array();
                            var index = 0;
                            //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                            for( var k=0; k < name_alumn_proves_acces_academia.length; k++ ) {
                                if(name_ges.indexOf(name_alumn_proves_acces_academia[k]) != -1){
                                    name_alumn_proves_academia_ges[index] =
name_alumn_proves_acces_academia[k];
                                    index++;
                                }
                                for( var j=0; j < name_alumn_proves_academia_ges.length; j++ ) {
                                    opUsers.add(new Option(name_alumn_proves_academia_ges[j],
name_alumn_proves_academia_ges[j]));
                                }
                            }
                        }
                        if (selectedRole == 9){
                            var name_alumn_extern_ges = new Array();
                            var index = 0;
                            //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                            for( var k=0; k < name_centre_client.length; k++ ) {
                                if(name_ges.indexOf(name_centre_client[k]) != -1){
                                    name_alumn_extern_ges[index] = name_centre_client[k];
                                    index++;
                                }
                                for( var j=0; j < name_alumn_extern_ges.length; j++ ) {
                                    opUsers.add(new Option(name_alumn_extern_ges[j],
name_alumn_extern_ges[j]));
                                }
```

```
                }
            }
        }
        if (subrole == "grau mitja") {
            if (selectedRole == 7){
                var name_alumn_proves_academia_grau_mitja = new Array();
                var index = 0;
                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                for( var k=0; k < name_alumn_proves_acces_academia.length; k++ ) {
                    if(name_grau_mitja.indexOf(name_alumn_proves_acces_academia[k]) != -1){
                        name_alumn_proves_academia_grau_mitja[index] =
name_alumn_proves_acces_academia[k];
                        index++;
                    }
                    for( var j=0; j < name_alumn_proves_academia_grau_mitja.length; j++ ) {
                        opUsers.add(new Option(name_alumn_proves_academia_grau_mitja[j],
name_alumn_proves_academia_grau_mitja[j]));
                    }
                }
            }
            if (selectedRole == 9){
                var name_alumn_extern_grau_mitja = new Array();
                var index = 0;
                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                for( var k=0; k < name_centre_client.length; k++ ) {
                    if(name_grau_mitja.indexOf(name_centre_client[k]) != -1){
                        name_alumn_extern_grau_mitja[index] = name_centre_client[k];
                        index++;
                    }
                    for( var j=0; j < name_alumn_extern_grau_mitja.length; j++ ) {
                        opUsers.add(new Option(name_alumn_extern_grau_mitja[j],
name_alumn_extern_grau_mitja[j]));
                    }
                }
            }
        }
        if (subrole == "grau superior") {
            if (selectedRole == 7){
                var name_alumn_proves_academia_grau_superior = new Array();
                var index = 0;
                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                for( var k=0; k < name_alumn_proves_acces_academia.length; k++ ) {
                    if(name_grau_superior.indexOf(name_alumn_proves_acces_academia[k]) != -1){
                        name_alumn_proves_academia_grau_superior[index] =
name_alumn_proves_acces_academia[k];
```

```
                                        index++;
                                    }
                                    for( var j=0; j < name_alumn_proves_academia_grau_superior.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_proves_academia_grau_superior[j],
name_alumn_proves_academia_grau_superior[j]));
                                    }
                                }
                            }
                            if (selectedRole == 9){
                                var name_alumn_extern_grau_superior = new Array();
                                var index = 0;
                                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                                for( var k=0; k < name_centre_client.length; k++ ) {
                                    if(name_grau_superior.indexOf(name_centre_client[k]) != -1){
                                        name_alumn_extern_grau_superior[index] = name_centre_client[k];
                                        index++;
                                    }
                                    for( var j=0; j < name_alumn_extern_grau_superior.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_extern_grau_superior[j],
name_alumn_extern_grau_superior[j]));
                                    }
                                }
                            }
                        }
                        if (subrole == "universitat majors") {
                            if (selectedRole == 7){
                                var name_alumn_proves_academia_universitat_majors_25 = new Array();
                                var index = 0;
                                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                                for( var k=0; k < name_alumn_proves_acces_academia.length; k++ ) {
                                    if(name_universitat_majors_25.indexOf(name_alumn_proves_acces_academia[k]) !
= -1){
                                        name_alumn_proves_academia_universitat_majors_25[index] =
name_alumn_proves_acces_academia[k];
                                        index++;
                                    }
                                    for( var j=0; j < name_alumn_proves_academia_universitat_majors_25.length; j+
+ ) {
                                        opUsers.add(new
Option(name_alumn_proves_academia_universitat_majors_25[j],
name_alumn_proves_academia_universitat_majors_25[j]));
                                    }
                                }
                            }
                            if (selectedRole == 9){
                                var name_alumn_extern_universitat_majors_25 = new Array();
```

```
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                        for( var k=0; k < name_centre_client.length; k++ ) {
                                if(name_universitat_majors_25.indexOf(name_centre_client[k]) != -1){
                                        name_alumn_extern_universitat_majors_25[index] =
name_centre_client[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_extern_universitat_majors_25.length; j++ ) {
                                         opUsers.add(new Option(name_alumn_extern_universitat_majors_25[j],
name_alumn_extern_universitat_majors_25[j]));
                                }
                        }
                       }
                }
                if (subrole == "tallers") {
            //per tallers, nomes alumne extern
                        if (selectedRole == 9){
                         var name_alumn_extern_tallers = new Array();
                         var index = 0;
                         //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                         for( var k=0; k < name_centre_client.length; k++ ) {
                                if(name_tallers.indexOf(name_centre_client[k]) != -1){
                                        name_alumn_extern_tallers[index] = name_centre_client[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_extern_tallers.length; j++ ) {
                                         opUsers.add(new Option(name_alumn_extern_tallers[j],
name_alumn_extern_tallers[j]));
                                }
                         }
                        }
                }
                if (subrole == "universitat") {
                        if (selectedRole == 3){
                         var name_alumn_matricula_grup_universitat = new Array();
                         var index = 0;
                         //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                         for( var k=0; k < name_alumn_matricula_grup.length; k++ ) {
                                if(name_universitat.indexOf(name_alumn_matricula_grup[k]) != -1){
                                        name_alumn_matricula_grup_universitat[index] =
name_alumn_matricula_grup[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_matricula_grup_universitat.length; j++ ) {
```

```
                                        opUsers.add(new Option(name_alumn_matricula_grup_universitat[j],
name_alumn_matricula_grup_universitat[j]));
                                    }
                                }
                            }
                            if (selectedRole == 4){
                                var name_alumn_matricula_particular_universitat = new Array();
                                var index = 0;
                                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                                for( var k=0; k < name_alumn_matricula_particular.length; k++ ) {
                                    if(name_universitat.indexOf(name_alumn_matricula_particular[k]) != -1){
                                        name_alumn_matricula_particular_universitat[index] =
name_alumn_matricula_particular[k];
                                        index++;
                                    }
                                    for( var j=0; j < name_alumn_matricula_particular_universitat.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_matricula_particular_universitat[j],
name_alumn_matricula_particular_universitat[j]));
                                    }
                                }
                            }
                            if (selectedRole == 5){
                                var name_alumn_pack_grup_universitat = new Array();
                                var index = 0;
                                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                                for( var k=0; k < name_alumn_pack_grup.length; k++ ) {
                                    if(name_universitat.indexOf(name_alumn_pack_grup[k]) != -1){
                                        name_alumn_pack_grup_universitat[index] = name_alumn_pack_grup[k];
                                        index++;
                                    }
                                    for( var j=0; j < name_alumn_pack_grup_universitat.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_pack_grup_universitat[j],
name_alumn_pack_grup_universitat[j]));
                                    }
                                }
                            }
                            if (selectedRole == 6){
                                var name_alumn_pack_particular_universitat = new Array();
                                var index = 0;
                                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                                for( var k=0; k < name_alumn_pack_particular.length; k++ ) {
                                    if(name_universitat.indexOf(name_alumn_pack_particular[k]) != -1){
                                        name_alumn_pack_particular_universitat[index] =
name_alumn_pack_particular[k];
                                        index++;
```

```
                                        }
                                        for( var j=0; j < name_alumn_pack_particular_universitat.length; j++ ) {
                                                opUsers.add(new Option(name_alumn_pack_particular_universitat[j],
name_alumn_pack_particular_universitat[j]));
                                        }
                                }
                        }
                }
                if (subrole == "batxillerat") {
                        if (selectedRole == 3){
                          var name_alumn_matricula_grup_batxillerat = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                          for( var k=0; k < name_alumn_matricula_grup.length; k++ ) {
                                        if(name_batxillerat.indexOf(name_alumn_matricula_grup[k]) != -1){
                                                name_alumn_matricula_grup_batxillerat[index] =
name_alumn_matricula_grup[k];
                                                index++;
                                        }
                                        for( var j=0; j < name_alumn_matricula_grup_batxillerat.length; j++ ) {
                                                opUsers.add(new Option(name_alumn_matricula_grup_batxillerat[j],
name_alumn_matricula_grup_batxillerat[j]));
                                        }
                          }
                        }
                        if (selectedRole == 4){
                          var name_alumn_matricula_particular_batxillerat = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex batxillerat)
                          for( var k=0; k < name_alumn_matricula_particular.length; k++ ) {
                                        if(name_batxillerat.indexOf(name_alumn_matricula_particular[k]) != -1){
                                                name_alumn_matricula_particular_batxillerat[index] =
name_alumn_matricula_particular[k];
                                                index++;
                                        }
                                        for( var j=0; j < name_alumn_matricula_particular_batxillerat.length; j++ ) {
                                                opUsers.add(new Option(name_alumn_matricula_particular_batxillerat[j],
name_alumn_matricula_particular_batxillerat[j]));
                                        }
                          }
                        }
                        if (selectedRole == 5){
                          var name_alumn_pack_grup_batxillerat = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex batxillerat)
```

```
                        for( var k=0; k < name_alumn_pack_grup.length; k++ ) {
                                if(name_batxillerat.indexOf(name_alumn_pack_grup[k]) != -1){
                                        name_alumn_pack_grup_batxillerat[index] = name_alumn_pack_grup[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_pack_grup_batxillerat.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_pack_grup_batxillerat[j],
name_alumn_pack_grup_batxillerat[j]));
                                }
                        }
                       }
                       if (selectedRole == 6){
                        var name_alumn_pack_particular_batxillerat = new Array();
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex batxillerat)
                        for( var k=0; k < name_alumn_pack_particular.length; k++ ) {
                                if(name_batxillerat.indexOf(name_alumn_pack_particular[k]) != -1){
                                        name_alumn_pack_particular_batxillerat[index] =
name_alumn_pack_particular[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_pack_particular_batxillerat.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_pack_particular_batxillerat[j],
name_alumn_pack_particular_batxillerat[j]));
                                }
                        }
                       }
                }
                if (subrole == "eso") {
                        if (selectedRole == 3){
                        var name_alumn_matricula_grup_eso = new Array();
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                        for( var k=0; k < name_alumn_matricula_grup.length; k++ ) {
                                if(name_eso.indexOf(name_alumn_matricula_grup[k]) != -1){
                                        name_alumn_matricula_grup_eso[index] = name_alumn_matricula_grup[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_matricula_grup_eso.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_matricula_grup_eso[j],
name_alumn_matricula_grup_eso[j]));
                                }
                        }
                       }
                       if (selectedRole == 4){
                        var name_alumn_matricula_particular_eso = new Array();
```

```
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                        for( var k=0; k < name_alumn_matricula_particular.length; k++ ) {
                                if(name_eso.indexOf(name_alumn_matricula_particular[k]) != -1){
                                        name_alumn_matricula_particular_eso[index] =
name_alumn_matricula_particular[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_matricula_particular_eso.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_matricula_particular_eso[j],
name_alumn_matricula_particular_eso[j]));
                                }
                        }
                       }
                       if (selectedRole == 5){
                        var name_alumn_pack_grup_eso = new Array();
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                        for( var k=0; k < name_alumn_pack_grup.length; k++ ) {
                                if(name_eso.indexOf(name_alumn_pack_grup[k]) != -1){
                                        name_alumn_pack_grup_eso[index] = name_alumn_pack_grup[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_pack_grup_eso.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_pack_grup_eso[j],
name_alumn_pack_grup_eso[j]));
                                }
                        }
                       }
                       if (selectedRole == 6){
                        var name_alumn_pack_particular_eso = new Array();
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                        for( var k=0; k < name_alumn_pack_particular.length; k++ ) {
                                if(name_eso.indexOf(name_alumn_pack_particular[k]) != -1){
                                        name_alumn_pack_particular_eso[index] =
name_alumn_pack_particular[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_pack_particular_eso.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_pack_particular_eso[j],
name_alumn_pack_particular_eso[j]));
                                }
                        }
                       }
```

```
		}
		if (subrole == "primaria") {
			if (selectedRole == 3){
			  var name_alumn_matricula_grup_primaria = new Array();
			  var index = 0;
			  //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
			  for( var k=0; k < name_alumn_matricula_grup.length; k++ ) {
				if(name_primaria.indexOf(name_alumn_matricula_grup[k]) != -1){
					name_alumn_matricula_grup_primaria[index] =
name_alumn_matricula_grup[k];
					index++;
				}
				for( var j=0; j < name_alumn_matricula_grup_primaria.length; j++ ) {
					 opUsers.add(new Option(name_alumn_matricula_grup_primaria[j],
name_alumn_matricula_grup_primaria[j]));
				}
			  }
			}
			if (selectedRole == 4){
			  var name_alumn_matricula_particular_primaria = new Array();
			  var index = 0;
			  //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex primaria)
			  for( var k=0; k < name_alumn_matricula_particular.length; k++ ) {
				if(name_primaria.indexOf(name_alumn_matricula_particular[k]) != -1){
					name_alumn_matricula_particular_primaria[index] =
name_alumn_matricula_particular[k];
					index++;
				}
				for( var j=0; j < name_alumn_matricula_particular_primaria.length; j++ ) {
					 opUsers.add(new Option(name_alumn_matricula_particular_primaria[j],
name_alumn_matricula_particular_primaria[j]));
				}
			  }
			}
			if (selectedRole == 5){
			  var name_alumn_pack_grup_primaria = new Array();
			  var index = 0;
			  //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex primaria)
			  for( var k=0; k < name_alumn_pack_grup.length; k++ ) {
				if(name_primaria.indexOf(name_alumn_pack_grup[k]) != -1){
					name_alumn_pack_grup_primaria[index] = name_alumn_pack_grup[k];
					index++;
				}
				for( var j=0; j < name_alumn_pack_grup_primaria.length; j++ ) {
```

```
                                        opUsers.add(new Option(name_alumn_pack_grup_primaria[j],
name_alumn_pack_grup_primaria[j]));
                                    }
                                }
                            }
                            if (selectedRole == 6){
                                var name_alumn_pack_particular_primaria = new Array();
                                var index = 0;
                                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex primaria)
                                for( var k=0; k < name_alumn_pack_particular.length; k++ ) {
                                    if(name_primaria.indexOf(name_alumn_pack_particular[k]) != -1){
                                        name_alumn_pack_particular_primaria[index] =
name_alumn_pack_particular[k];
                                        index++;
                                    }
                                    for( var j=0; j < name_alumn_pack_particular_primaria.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_pack_particular_primaria[j],
name_alumn_pack_particular_primaria[j]));
                                    }
                                }
                            }
                        }
                    }
                    function AddUserToSelection(SelectedUser)
                    {
                      var separador = ' | ';
                var usuari_seleccionat_sol = SelectedUser;
                SelectedUser = SelectedUser + separador + document.getElementById("f_drupal_roles").value
                 if (document.getElementById("f_sub_roles").value.length > 0){
                   SelectedUser = SelectedUser + separador + document.getElementById("f_sub_roles").value
                 }
                var userExists = false;
                /*check the first empty position*/
                for(var j=1; j<=room_capacity; j++) {
                    if (j == room_capacity) {
                        userExists = true;
                        alert ('L Aula no te mes capacitat. Si vols reservar mes usuaris ho hauras de fer en una altra aula o ampliar
la capacitat d aquesta.');
                    }
                    if (j<=9){
                      var input_id = 'f_Alumne_0' + j;
                    }
                    else{
                      var input_id = 'f_Alumne_' + j;
                    }
                            if(document.getElementById(input_id).value=="") {
                        /*match, exit iteration*/
```

```
                    j=41;
                }
            }
/*check if user exists*/
for(var j=1; j<=40; j++) {
   if (j<=9){
     var input_id_check = 'f_Alumne_0' + j;
   }
   else{
     var input_id_check = 'f_Alumne_' + j;
   }
   var user_name_complete = document.getElementById(input_id_check).value;
   var user_name_only = user_name_complete.substr(0,user_name_complete.indexOf(" | "));
   if(user_name_only == usuari_seleccionat_sol){
     userExists = true;
     alert ('L usuari ja esta a la reserva');
   }
}
/*check if user is current user*/
if(usuari_seleccionat_sol == current_user){
      userExists = true;
      alert ('No fa falta que t introdueixis a la reserva');
}
     if (userExists == false){
        document.getElementById(input_id).value = SelectedUser;
     }
   }
   </script>

   <select id="f_drupal_roles" onchange="DrupalUsersAndSubRoles(this.options[this.selectedIndex].value);">
   <option value="">Selecciona un tipus d'usuari</option>
   <option value="general">Tots els usuaris (inclos registrats no alumnes)</option>
   <option value="alumn_estiu">Alumne estiu</option>
   <option value="alumn_matricula_grup">Alumne matricula grup</option>
   <option value="alumn_matricula_particular">Alumne matricula particular</option>
   <option value="alumn_pack_grup">Alumne pack grup</option>
   <option value="alumn_pack_particular">Alumne pack particular</option>
   <option value="alumn_proves_acces_academia">Alumne proves acces academia</option>
   <option value="alumn_selectivitat">Alumne selectivitat</option>
   <option value="alumn_centre_client">Alumne centre client</option>
   <option value="responsable_centre_client">responsable centre client</option>
   <option value="professor">professor</option>
   <option value="professor_soci">professor soci</option>
   <option value="alumn_offline">alumne offline</option>
   <option value="universitat">Tots els alumnes d'universitat</option>
   <option value="batxillerat">Tots els alumnes de batxillerat</option>
   <option value="eso">Tots els alumnes d'eso</option>
   <option value="primaria">Tots els alumnes de primaria</option>
```

```
        <option value="tallers" title="Inclou els que no tenen rol de alumne centre client">Alumnes de Tallers i cursos Varis</option>
            </select>
            </div>
            </div>
            <div id="add_drupal_users_subroles">
            <label for="add_drupal_users_subroles">Nivell</label>
            <select id="f_sub_roles" onchange="DrupalUsersForSubRoles(this.options[this.selectedIndex].value);">
            </select>
            </div>
            <div id="add_drupal_users_to_selection">
            <label for="add_drupal_users_to_selection">Usuaris</label>
            <select id="f_filtered_users" size="5" onclick="AddUserToSelection(this.options[this.selectedIndex].value);">
            </select>
           <?php
            echo "<br>";
            echo "</div>\n";
  }
}

// Get non-standard form variables
$hour = get_form_var('hour', 'int');
$minute = get_form_var('minute', 'int');
$period = get_form_var('period', 'int');
$id = get_form_var('id', 'int');
$copy = get_form_var('copy', 'int');
$edit_type = get_form_var('edit_type', 'string', '');
$returl = get_form_var('returl', 'string');
// The following variables are used when coming via a JavaScript drag select
$drag = get_form_var('drag', 'int');
$start_seconds = get_form_var('start_seconds', 'int');
$end_seconds = get_form_var('end_seconds', 'int');
$selected_rooms = get_form_var('rooms', 'array');
$start_date = get_form_var('start_date', 'string');
$end_date = get_form_var('end_date', 'string');


// Check the user is authorised for this page
checkAuthorised();

// Also need to know whether they have admin rights
$user = getUserName();
$is_admin = (authGetUserLevel($user) >= 2);
// You're only allowed to make repeat bookings if you're an admin
// or else if $auth['only_admin_can_book_repeat'] is not set
$repeats_allowed = $is_admin || empty($auth['only_admin_can_book_repeat']);
// Similarly for multi-day
```

```
$multiday_allowed = $is_admin || empty($auth['only_admin_can_book_multiday']);
// Similarly for multiple room selection
$multiroom_allowed = $is_admin || empty($auth['only_admin_can_select_multiroom']);



if (isset($start_seconds))
{
  $minutes = intval($start_seconds/60);
  if ($enable_periods)
  {
    $period = $minutes - (12*60);
  }
  else
  {
    $hour = intval($minutes/60);
    $minute = $minutes%60;
  }
}

if (isset($start_date))
{
  list($year, $month, $day) = explode('-', $start_date);
  if (isset($end_date) && ($start_date != $end_date) && $repeats_allowed)
  {
    $rep_type = REP_DAILY;
    list($rep_end_year, $rep_end_month, $rep_end_day) = explode('-', $end_date);
  }
}



// We might be going through edit_entry more than once, for example if we have to log on on the way.  We
// still need to preserve the original calling page so that once we've completed edit_entry_handler we can
// go back to the page we started at (rather than going to the default view).  If this is the first time
// through, then $HTTP_REFERER holds the original caller.    If this is the second time through we will have
// stored it in $returl.
if (!isset($returl))
{
  $returl = isset($HTTP_REFERER) ? $HTTP_REFERER : "";
}



// This page will either add or modify a booking

// We need to know:
//  Name of booker
```

```
//  Description of meeting
//modified
//  the drupal users to add
//  Date (option select box for day, month, year)
//  Time
//  Duration
//  Internal/External

$fields = sql_field_info($tbl_entry);
$custom_fields = array();
//modified
$drupal_users = array();

// Firstly we need to know if this is a new booking or modifying an old one
// and if it's a modification we need to get all the old data from the db.
// If we had $id passed in then it's a modification.

if (isset($id))
{
  $sql = "SELECT *
           FROM $tbl_entry
          WHERE id=$id
          LIMIT 1";

  $res = sql_query($sql);
  if (! $res)
  {
    trigger_error(sql_error(), E_USER_WARNING);
    fatal_error(TRUE, get_vocab("fatal_db_error"));
  }
  if (sql_count($res) != 1)
  {
    fatal_error(1, get_vocab("entryid") . $id . get_vocab("not_found"));
  }

  $row = sql_row_keyed($res, 0);
  sql_free($res);

  // We've possibly got a new room and area, so we need to update the settings
  // for this area.
  $area = get_area($row['room_id']);
  get_area_settings($area);

  $private = $row['status'] & STATUS_PRIVATE;
  if ($private_mandatory)
  {
    $private = $private_default;
  }
```

```
// Need to clear some data if entry is private and user
// does not have permission to edit/view details
if (isset($copy) && ($create_by != $row['create_by']))
{
  // Entry being copied by different user
  // If they don't have rights to view details, clear them
  $privatewriteable = getWritable($row['create_by'], $user, $row['room_id']);
  $keep_private = (is_private_event($private) && !$privatewriteable);
}
else
{
  $keep_private = FALSE;
}

foreach ($row as $column => $value)
{
  switch ($column)
  {
    // Don't bother with these columns
    case 'id':
    case 'timestamp':
    case 'reminded':
    case 'info_time':
    case 'info_user':
    case 'info_text':
      break;

    // These columns cannot be made private
    case 'room_id':
      // We need to preserve the original room_id for existing bookings and pass
      // it through to edit_entry_handler.    We need this because we need to know
      // in edit_entry_handler which room contains the original booking.   It's
      // possible in this form to select multiple rooms, or even change the room.
      // We will need to know which booking is the "original booking" because the
      // original booking will keep the same ical_uid and have the ical_sequence
      // incremented, whereas new bookings will have a new ical_uid and start with
      // an ical_sequence of 0.    (If there is more than one room when we get to
      // edit_entry_handler and the original room isn't among them, then we will
      // just have to make an arbitrary choice as to which is the room containing
      // the original booking.)
      // NOTE:  We do not set the original_room_id if we are copying an entry,
      // because when we are copying we are effectively making a new entry and
      // so we want edit_entry_handler to assign a new UID, etc.
      if (!$copy)
      {
        $original_room_id = $row['room_id'];
      }
    case 'ical_uid':
```

```
    case 'ical_sequence':
    case 'ical_recur_id':
    case 'entry_type':
      $$column = $row[$column];
      break;

    // These columns can be made private [not sure about 'type' though - haven't
    // checked whether it makes sense/works to make the 'type' column private]
    case 'name':
    case 'description':
    case 'type':
      $$column = ($keep_private && $is_private_field["entry.$column"]) ? '' : $row[$column];
      break;

    case 'status':
      // No need to do the privacy status as we've already done that.
      // Just do the confirmation status
      $confirmed = !($row['status'] & STATUS_TENTATIVE);
      break;

    case 'repeat_id':
      $rep_id      = $row['repeat_id'];
      break;

    case 'create_by':
      // If we're copying an existing entry then we need to change the create_by (they could be
      // different if it's an admin doing the copying)
      $create_by   = (isset($copy)) ? $user : $row['create_by'];
      break;

    case 'start_time':
      $start_time = $row['start_time'];
      break;

    case 'end_time':
      $end_time = $row['end_time'];
      $duration = $row['end_time'] - $row['start_time'] - cross_dst($row['start_time'], $row['end_time']);
      break;

    default:
      $custom_fields[$column] = ($keep_private && $is_private_field["entry.$column"]) ? '' : $row[$column];
      break;
  }
}


if(($entry_type == ENTRY_RPT_ORIGINAL) || ($entry_type == ENTRY_RPT_CHANGED))
{
```

```
$sql = "SELECT rep_type, start_time, end_time, end_date, rep_opt, rep_num_weeks
         FROM $tbl_repeat
        WHERE id=$rep_id
        LIMIT 1";

$res = sql_query($sql);
if (! $res)
{
  trigger_error(sql_error(), E_USER_WARNING);
  fatal_error(TRUE, get_vocab("fatal_db_error"));
}
if (sql_count($res) != 1)
{
  fatal_error(1,
            get_vocab("repeat_id") . $rep_id . get_vocab("not_found"));
}

$row = sql_row_keyed($res, 0);
sql_free($res);

$rep_type = $row['rep_type'];

// If it's a repeating entry get the repeat details
if (isset($rep_type) && ($rep_type != REP_NONE))
{
  // If we're editing the series we want the start_time and end_time to be the
  // start and of the first entry of the series, not the start of this entry
  if ($edit_type == "series")
  {
    $start_time = $row['start_time'];
    $end_time = $row['end_time'];
  }

  $rep_end_day   = (int)strftime('%d', $row['end_date']);
  $rep_end_month = (int)strftime('%m', $row['end_date']);
  $rep_end_year  = (int)strftime('%Y', $row['end_date']);
  // Get the end date in string format as well, for use when
  // the input is disabled
  $rep_end_date = utf8_strftime('%A %d %B %Y',$row['end_date']);

  $rep_day = array();
  switch ($rep_type)
  {
    case REP_WEEKLY:
    case REP_N_WEEKLY:
      for ($i=0; $i<7; $i++)
      {
        if ($row['rep_opt'][$i])
```

```
          {
           $rep_day[] = $i;
          }
         }
         // Get the repeat days as an array for use
         // when the input is disabled
         $rep_opt = $row['rep_opt'];

         if ($rep_type == REP_N_WEEKLY)
         {
          $rep_num_weeks = $row['rep_num_weeks'];
         }

         break;

       default:
         break;
      }
    }
  }
}
else
{
  // It is a new booking. The data comes from whichever button the user clicked
  $edit_type     = "series";
  $name          = "";
  $create_by     = $user;
  $description   = $default_description;
  $type          = $default_type;
  $room_id       = $room;
  $rep_id        = 0;
  if (!isset($rep_type))  // We might have set it through a drag selection
  {
   $rep_type      = REP_NONE;
   $rep_end_day   = $day;
   $rep_end_month = $month;
   $rep_end_year  = $year;
  }
  $rep_day       = array();
  $private       = $private_default;
  $confirmed     = $confirmed_default;

  // now initialise the custom fields
  foreach ($fields as $field)
  {
   if (!in_array($field['name'], $standard_fields['entry']))
   {
    $custom_fields[$field['name']] = '';
```

```
  }
}

// Get the hour and minute, converting a period to its MRBS time
// Set some sensible defaults
if ($enable_periods)
{
  if (isset($period))
  {
    $hour = 12 + intval($period/60);
    $minute = $period % 60;
  }
  else
  {
    $hour = 0;
    $minute = 0;
  }
}
else
{
  if (!isset($hour) || !isset($minute))
  {
    $hour = $morningstarts;
    $minute = $morningstarts_minutes;
  }
}

$start_time = mktime($hour, $minute, 0, $month, $day, $year);

if (isset($end_seconds))
{
  $end_minutes = intval($end_seconds/60);
  $end_hour = intval($end_minutes/60);
  $end_minute = $end_minutes%60;
  $end_time = mktime($end_hour, $end_minute, 0, $month, $day, $year);
  $duration = $end_time - $start_time - cross_dst($start_time, $end_time);
}
else
{
  if (!isset($default_duration))
  {
    $default_duration = (60 * 60);
  }
  $duration    = ($enable_periods ? 60 : $default_duration);
  $end_time = $start_time + $duration;
  // The end time can't be past the end of the booking day
  $pm7 = mktime($eveningends, $eveningends_minutes, 0, $month, $day, $year);
  $end_time = min($end_time, $pm7 + $resolution);
```

```
  }
}

$start_hour  = strftime('%H', $start_time);
$start_min   = strftime('%M', $start_time);

// These next 4 if statements handle the situation where
// this page has been accessed directly and no arguments have
// been passed to it.
// If we have not been provided with a room_id
if (empty( $room_id ) )
{
  $sql = "SELECT id FROM $tbl_room WHERE disabled=0 LIMIT 1";
  $res = sql_query($sql);
  $row = sql_row_keyed($res, 0);
  $room_id = $row['id'];
}

// Determine the area id of the room in question first
$area_id = mrbsGetRoomArea($room_id);


// Remove "Undefined variable" notice
if (!isset($rep_num_weeks))
{
  $rep_num_weeks = "";
}

$enable_periods ? toPeriodString($start_min, $duration, $dur_units) : toTimeString($duration, $dur_units);

//now that we know all the data to fill the form with we start drawing it

if (!getWritable($create_by, $user, $room_id))
{
  showAccessDenied($day, $month, $year, $area, isset($room) ? $room : "");
  exit;
}

print_header($day, $month, $year, $area, isset($room) ? $room : "");

// Get the details of all the enabled rooms
$rooms = array();
$sql = "SELECT R.id, R.room_name, R.area_id, R.capacity
          FROM $tbl_room R, $tbl_area A
         WHERE R.area_id = A.id
           AND R.disabled=0
           AND A.disabled=0
      ORDER BY R.area_id, R.sort_key"; // modified
```

```
$res = sql_query($sql);
if ($res)
{
  for ($i = 0; ($row = sql_row_keyed($res, $i)); $i++)
  {
    $rooms[$row['id']] = $row;
  }
}

// Get the details of all the enabled areas
$areas = array();
$sql = "SELECT id, area_name, resolution, default_duration, enable_periods,
               morningstarts, morningstarts_minutes, eveningends , eveningends_minutes
          FROM $tbl_area
         WHERE disabled=0
      ORDER BY area_name";
$res = sql_query($sql);
if ($res)
{
  for ($i = 0; ($row = sql_row_keyed($res, $i)); $i++)
  {
    $areas[$row['id']] = $row;
    // The following config settings aren't yet per-area, but we'll treat them as if
    // they are to make it easier to change them to per-area settings in the future.
    $areas[$row['id']]['max_duration_enabled'] = $max_duration_enabled;
    $areas[$row['id']]['max_duration_secs']    = $max_duration_secs;
    $areas[$row['id']]['max_duration_periods'] = $max_duration_periods;
    // Clean up the settings, getting rid of any nulls and casting boolean fields into bools
    $areas[$row['id']] = clean_area_row($areas[$row['id']]);
    // Generate some derived settings
    $areas[$row['id']]['max_duration_qty'] = $areas[$row['id']]['max_duration_secs'];
    toTimeString($areas[$row['id']]['max_duration_qty'], $areas[$row['id']]['max_duration_units']);
  }
}

?>

<script type="text/javascript">
//<![CDATA[

var currentArea = <?php echo $area_id ?>;
var areas = new Array();
<?php
// give JavaScript a copy of the PHP array $areas
foreach ($areas as $area)
{
  echo "areas[${area['id']}] = new Array();\n";
  foreach ($area as $key => $value)
```

```
  {
   if (in_array($key, array('area_name', 'max_duration_units')))
   {
    // Enclose strings in quotes
    $value = "'" . escape_js($value) . "'";
   }
   elseif (in_array($key, $boolean_fields['area']))
   {
    // Convert booleans
    $value = ($value) ? 'true' : 'false';
   }
   echo "areas[${area['id']}]['$key'] = $value;\n";
  }
}
?>

// do a little form verifying
function validate(form_id)
{
  <?php
  // First of all check that a name (brief description) has been entered.
  // Only do this if the name is being entered via an INPUT box.   If it's
  // being entered via a SELECT box there's no need to do this because there's
  // bound to be a value and the test below will fail on some browsers (eg IE)
  ?>
  var form = document.getElementById(form_id);
  if (form.name.tagName.toLowerCase() == 'input')
  {
   // null strings and spaces only strings not allowed
   if(/(^$)|(^\s+$)/.test(form.name.value))
   {
    alert("<?php echo escape_js(get_vocab('you_have_not_entered')) . '\n' . escape_js(get_vocab('brief_description')) ?
>");
    return false;
   }
  }

  <?php
  // Check that the start date is not after the end date
  ?>
  var dateDiff = getDateDifference(form);
  if (dateDiff < 0)
  {
   alert("<?php echo escape_js(get_vocab('start_after_end_long'))?>");
   return false;
  }

  // check form element exist before trying to access it
```

```
  if (form.id )
  {
   i1 = parseInt(form.id.value);
  }
  else
  {
   i1 = 0;
  }

  i2 = parseInt(form.rep_id.value);
  if (form.rep_num_weeks)
  {
    n = parseInt(form.rep_num_weeks.value);
  }
  if ((!i1 || (i1 && i2)) &&
      form.rep_type &&
      (form.rep_type.value != <?php echo REP_NONE ?>) &&
      form.rep_type[<?php echo REP_N_WEEKLY ?>].checked &&
      (!n || n < 2))
  {
    alert("<?php echo escape_js(get_vocab('you_have_not_entered')) . '\n' . escape_js(get_vocab('useful_n-
weekly_value')) ?>");
    return false;
  }


  // check that a room(s) has been selected
  // this is needed as edit_entry_handler does not check that a room(s)
  // has been chosen
  if (form.elements['rooms'].selectedIndex == -1 )
  {
   alert("<?php echo escape_js(get_vocab('you_have_not_selected')) . '\n' . escape_js(get_vocab('valid_room')) ?>");
   return false;
  }

  <?php
  if (count($is_mandatory_field))
  {
   $m_fields = array();
   foreach ($is_mandatory_field as $field => $value)
   {
     if ($value)
     {
       $field = preg_replace('/^entry\./', 'f_', $field);
       $m_fields[] = "'".str_replace("'", "\\'", $field)."'";
     }
   }
   echo "var mandatory_fields = [".implode(', ', $m_fields)."];\n";
```

```
  ?>

    var return_val = true;

    $.each(mandatory_fields,
           function(index, value)
           {
             var field = $("#"+value);
             <?php
             // If it's a checkbox then it needs to be checked.    If it's
             // an ordinary field then it must have some content.
             ?>
             if ( ((field.attr('type').toLowerCase() == 'checkbox') && !field.attr('checked')) ||
                  (field.val() == '') )
             {
               label = $("label[for="+value+"]").html();
               label = label.replace(/:$/, '');
               alert('"' + label + '" ' +
                 <?php echo '"' . escape_js(get_vocab('is_mandatory_field')) . '"'; ?>);
               return_val = false;
             }
           });
    if (!return_val)
    {
      return return_val;
    }
  <?php

  }

  // Form submit can take some times, especially if mails are enabled and
  // there are more than one recipient. To avoid users doing weird things
  // like clicking more than one time on submit button, we hide it as soon
  // it is clicked.
  ?>
  form.save_button.disabled = true;

  // would be nice to also check date to not allow Feb 31, etc...

  return true;
}

// set up some global variables for use by OnAllDayClick().
var old_start, old_end;

// Executed when the user clicks on the all_day checkbox.
function OnAllDayClick(el)
{
```

```
 var form = document.forms["main"];
 if (form)
 {
  var startSelect = form["start_seconds" + currentArea];
  var endSelect = form["end_seconds" + currentArea];
  var allDay = form["all_day" + currentArea];
  var i;
  if (allDay.checked) // If checking the box...
  {
   <?php
   // Save the old values, disable the inputs and, to avoid user confusion,
   // show the start and end times as the beginning and end of the booking
   // (Note that we save the value rather than the index because the number
   // of options in the select box will change)
   ?>
   old_start = startSelect.options[startSelect.selectedIndex].value;
   startSelect.selectedIndex = 0;
   startSelect.disabled = true;

   old_end = endSelect.options[endSelect.selectedIndex].value;
   endSelect.selectedIndex = endSelect.options.length - 1;
   endSelect.disabled = true;
  }
  else  <?php // restore the old values and re-enable the inputs ?>
  {
   startSelect.disabled = false;
   for (i=0; i<startSelect.options.length; i++)
   {
    if (startSelect.options[i].value == old_start)
    {
     startSelect.options.selectedIndex = i;
     break;
    }
   }
   endSelect.disabled = false;
   for (i=0; i<endSelect.options.length; i++)
   {
    if (endSelect.options[i].value == old_end)
    {
     endSelect.options.selectedIndex = i;
     break;
    }
   }
   prevStartValue = undefined;  <?php // because we don't want adjustSlotSelectors() to change the end time ?>
  }
  adjustSlotSelectors(form); <?php // need to get the duration right ?>
 }
}
```

```
//]]>
</script>

<?php

if (isset($id) && !isset($copy))
{
  if ($edit_type == "series")
  {
    $token = "editseries";
  }
  else
  {
    $token = "editentry";
  }
}
else
{
  if (isset($copy))
  {
    if ($edit_type == "series")
    {
      $token = "copyseries";
    }
    else
    {
      $token = "copyentry";
    }
  }
  else
  {
    $token = "addentry";
  }
}
?>


<form class="form_general" id="main" action="edit_entry_handler.php" method="post">
  <fieldset>
  <legend><?php echo get_vocab($token); ?></legend>

<?php

// Fill $edit_entry_field_order with not yet specified entries.
$entry_fields = array('name', 'description', 'start_date', 'end_date', 'areas',
  'rooms', 'type', 'confirmation_status', 'privacy_status', 'add_drupal_users');
foreach( $entry_fields as $field )
{
```

```
  if( ! in_array( $field, $edit_entry_field_order ) )
    $edit_entry_field_order[] = $field;
}

// CUSTOM FIELDS
$custom_fields_map = array();
foreach ($fields as $field)
{
  $key = $field['name'];
  if (!in_array($key, $standard_fields['entry']))
  {
    $custom_fields_map[$key] = $field;
    if( ! in_array( $key, $edit_entry_field_order ) )
      $edit_entry_field_order[] = $key;
  }
}

foreach( $edit_entry_field_order as $key )
{
  switch( $key )
  {
  case 'name':
    create_field_entry_name();
    break;

  case 'description':
    create_field_entry_description();
    break;

  case 'start_date':
    create_field_entry_start_date();
    break;

  case 'end_date':
    create_field_entry_end_date();
    break;

  case 'areas':
    create_field_entry_areas();
    break;

  case 'rooms':
    create_field_entry_rooms();
    break;

  case 'type':
    create_field_entry_type();
    break;
```

```
  case 'confirmation_status':
    create_field_entry_confirmation_status();
    break;

  case 'privacy_status':
    create_field_entry_privacy_status();
    break;

  case 'add_drupal_users':
    create_field_entry_add_drupal_users();
    break;

  default:
    create_field_entry_custom_field($custom_fields_map[$key], $key);
    break;
  }
}

    // REPEAT BOOKING INPUTS
    if (($edit_type == "series") && $repeats_allowed)
    {
      // If repeats are allowed and the edit_type is a series (which means
      // that either you're editing an existing series or else you're making
      // a new booking) then print the repeat inputs
      echo "<fieldset id=\"rep_info\">\n";
      echo "<legend></legend>\n";
      ?>
      <div id="rep_type">
        <label><?php echo get_vocab("rep_type")?>:</label>
        <div class="group">
          <?php
          for ($i = 0; isset($vocab["rep_type_$i"]); $i++)
          {
            echo "      <label><input class=\"radio\" name=\"rep_type\" type=\"radio\" value=\"" . $i . "\"";
            if ($i == $rep_type)
            {
              echo " checked=\"checked\"";
            }
            echo ">" . get_vocab("rep_type_$i") . "</label>\n";
          }
          ?>
        </div>
      </div>

      <div id="rep_end_date">
        <?php
        echo "<label>" . get_vocab("rep_end_date") . ":</label>\n";
```

```
  genDateSelector("rep_end_", $rep_end_day, $rep_end_month, $rep_end_year);
  ?>
 </div>

 <div id="rep_day">
  <label><?php echo get_vocab("rep_rep_day")?>:<br><?php echo get_vocab("rep_for_weekly")?></label>
  <div class="group">
   <?php
   // Display day name checkboxes according to language and preferred weekday start.
   for ($i = 0; $i < 7; $i++)
   {
    $wday = ($i + $weekstarts) % 7;
    echo "      <label><input class=\"checkbox\" name=\"rep_day[]\" value=\"$wday\" type=\"checkbox\"";
    if (in_array($wday, $rep_day))
    {
     echo " checked=\"checked\"";
    }
    echo ">" . day_name($wday) . "</label>\n";
   }
   ?>
  </div>
 </div>

 <?php
 echo "<div>\n";
 $label_text = get_vocab("rep_num_weeks") . ":<br>" . get_vocab("rep_for_nweekly");
 generate_input($label_text, 'rep_num_weeks', $rep_num_weeks);
 echo "</div>\n";
 // Checkbox for skipping past conflicts
 echo "<div>\n";
 echo "<label for=\"skip\">" . get_vocab("skip_conflicts") . ":</label>\n";
 echo "<input type=\"checkbox\" class=\"checkbox\" " .
        "id=\"skip\" name=\"skip\" value=\"1\" " .
        ((!empty($skip_default)) ? " checked=\"checked\"" : "") .
        ">\n";
 echo "</div>\n";

 echo "</fieldset>\n";
}
elseif (isset($id))
{
 // otherwise, if it's an existing booking, show the repeat information
 // and pass it through to the handler but do not let the user edit it
 // (because they're either not allowed to, or else they've chosen to edit
 // an individual entry rather than a series).
 // (NOTE: when repeat bookings are restricted to admins, an ordinary user
 // would not normally be able to get to the stage of trying to edit a series.
 // But we have to cater for the possibility because it could happen if (a) the
```

```
    // series was created before the policy was introduced or (b) the user has
    // been demoted since the series was created).
    $key = "rep_type_" . (isset($rep_type) ? $rep_type : REP_NONE);
    echo "<fieldset id=\"rep_info\">\n";
    echo "<legend></legend>\n";
    echo "<div>\n";
    echo "<label>" . get_vocab("rep_type") . ":</label>\n";
    echo "<select disabled=\"disabled\">\n";
    echo "<option>" . get_vocab($key) . "</option>\n";
    echo "</select>\n";
    echo "<input type=\"hidden\" name=\"rep_type\" value=\"" . REP_NONE . "\">\n";
    echo "</div>\n";
    if (isset($rep_type) && ($rep_type != REP_NONE))
    {
     $opt = "";
     if (($rep_type == REP_WEEKLY) || ($rep_type == REP_N_WEEKLY))
     {
      // Display day names according to language and preferred weekday start.
      for ($i = 0; $i < 7; $i++)
      {
       $wday = ($i + $weekstarts) % 7;
       if ($rep_opt[$wday])
       {
        $opt .= day_name($wday) . " ";
       }
      }
     }
     if($opt)
     {
      echo "  <div><label>".get_vocab("rep_rep_day").":</label><input type=\"text\" value=\"$opt\"
disabled=\"disabled\"></div>\n";
     }
     echo "  <div><label>".get_vocab("rep_end_date").":</label><input type=\"text\" value=\"$rep_end_date\"
disabled=\"disabled\"></div>\n";
     if ($rep_type == REP_N_WEEKLY)
     {
      echo "<div>\n";
      echo "<label for=\"rep_num_weeks\">" . get_vocab("rep_num_weeks") . ":<br>" .
get_vocab("rep_for_nweekly") . "</label>\n";
      echo "<input type=\"text\" id=\"rep_num_weeks\" name=\"rep_num_weeks\" value=\"$rep_num_weeks\"
disabled=\"disabled\">\n";
      echo "</div>\n";
     }
    }
    echo "</fieldset>\n";
   }

   ?>
```

```
<input type="hidden" name="returl" value="<?php echo htmlspecialchars($returl) ?>">
<input type="hidden" name="create_by" value="<?php echo htmlspecialchars($create_by)?>">
<input type="hidden" name="rep_id" value="<?php echo $rep_id?>">
<input type="hidden" name="edit_type" value="<?php echo $edit_type?>">
<?php
// The original_room_id will only be set if this was an existing booking.
// If it is an existing booking then edit_entry_handler needs to know the
// original room id and the ical_uid and the ical_sequence, because it will
// have to keep the ical_uid and increment the ical_sequence for the room that
// contained the original booking.  If it's a new booking it will generate a new
// ical_uid and start the ical_sequence at 0.
if (isset($original_room_id))
{
  echo "<input type=\"hidden\" name=\"original_room_id\" ".
    "value=\"$original_room_id\">\n";
  echo "<input type=\"hidden\" name=\"ical_uid\" value=\"".
    htmlspecialchars($ical_uid)."\">\n";
  echo "<input type=\"hidden\" name=\"ical_sequence\" value=\"".
    htmlspecialchars($ical_sequence)."\">\n";
  echo "<input type=\"hidden\" name=\"ical_recur_id\" value=\"".
    htmlspecialchars($ical_recur_id)."\">\n";
}
if(isset($id) && !isset($copy))
{
  echo "<input type=\"hidden\" name=\"id\" value=\"$id\">\n";
}

// Buttons
echo "<fieldset class=\"submit_buttons\">\n";
echo "<legend></legend>\n";
// The Back button
echo "<div id=\"edit_entry_submit_back\">\n";
echo "<input class=\"submit\" type=\"submit\" name=\"back_button\" value=\"" . get_vocab("back") . "\">\n";
echo "</div>\n";

// The Submit button
echo "<div id=\"edit_entry_submit_save\">\n";
echo "<input class=\"submit\" type=\"submit\" name=\"save_button\" value=\"" .
  get_vocab("save") . "\">\n";
echo "</div>\n";

// divs to hold the results of the Ajax checking of the booking
echo "<div id=\"conflict_check\">\n";
echo "</div>\n";

echo "<div id=\"policy_check\">\n";
echo "</div>\n";
```

```
    echo "</fieldset>";

    // and a div to hold the dialog box which gives more details.    The dialog
    // box contains a set of tabs.   And because we want the tabs to act as the
    // dialog box we add an extra tab where we're going to put the dialog close
    // button and then we hide the dialog itself
    echo "<div id=\"check_results\" style=\"display: none\">\n";
    echo "<div id=\"check_tabs\">\n";
    echo "<ul id=\"details_tabs\">\n";
    echo "<li><a href=\"#schedule_details\">" . get_vocab("schedule") . "</a></li>\n";
    echo "<li><a href=\"#policy_details\">" . get_vocab("policy") . "</a></li>\n";
    echo "<li id=\"ui-tab-dialog-close\"></li>\n";
    echo "</ul>\n";
    echo "<div id=\"schedule_details\"></div>\n";
    echo "<div id=\"policy_details\"></div>\n";
    echo "</div>\n";
    echo "</div>\n";
    ?>
  </fieldset>
</form>



<?php
//modified
$sql_1 = "SELECT capacity
        FROM mrbs_room
        WHERE id=\"$room\"";
$room_capacity = sql_query1($sql_1);
?>
<script language="JavaScript">
var current_user = ["<?php echo $user; ?>"];
var room_capacity = ["<?php echo $room_capacity; ?>"];
</script>
<?php
require_once "trailer.inc"
?>
```

Fitxer view_entry.php (el codi modificat es pot veure sota el tag //modified):

```
<?php
// $Id: view_entry.php 2210 2011-12-21 22:44:17Z cimorrison $

require_once "defaultincludes.inc";
require_once "mrbs_sql.inc";
require_once "functions_view.inc";
```

```
//modified
//amagar camps amb valors buits
// Generates a single button
function generateButton($form_action, $id, $series, $action_type, $returl, $submit_value, $title='')
{
  global $room_id;

  echo "<form action=\"".htmlspecialchars($form_action).
    "?id=$id&amp;series=$series\" method=\"post\">\n";
  echo "<fieldset>\n";
  echo "<legend></legend>\n";
  echo "<input type=\"hidden\" name=\"action\" value=\"$action_type\">\n";
  echo "<input type=\"hidden\" name=\"room_id\" value=\"$room_id\">\n";
  echo "<input type=\"hidden\" name=\"returl\" value=\"" . htmlspecialchars($returl) . "\">\n";
  echo "<input type=\"submit\" title=\"" . htmlspecialchars($title) . "\" value=\"$submit_value\">\n";
  echo "</fieldset>\n";
  echo "</form>\n";
}

// Generates the Approve, Reject and More Info buttons
function generateApproveButtons($id, $series)
{
  global $returl, $PHP_SELF;
  global $entry_info_time, $entry_info_user, $repeat_info_time, $repeat_info_user;

  $info_time = ($series) ? $repeat_info_time : $entry_info_time;
  $info_user = ($series) ? $repeat_info_user : $entry_info_user;

  $this_page = basename($PHP_SELF);
  if (empty($info_time))
  {
    $info_title = get_vocab("no_request_yet");
  }
  else
  {
    $info_title = get_vocab("last_request") . ' ' . time_date_string($info_time);
    if (!empty($info_user))
    {
      $info_title .= " " . get_vocab("by") . " $info_user";
    }
  }

  echo "<tr>\n";
  echo "<td>" . ($series ? get_vocab("series") : get_vocab("entry")) . ":</td>\n";
  echo "<td>\n";
  generateButton("approve_entry_handler.php", $id, $series, "approve", $returl, get_vocab("approve"));
  generateButton($this_page, $id, $series, "reject", $returl, get_vocab("reject"));
  generateButton($this_page, $id, $series, "more_info", $returl, get_vocab("more_info"), $info_title);
```

```
  echo "</td>\n";
  echo "</tr>\n";
}

function generateOwnerButtons($id, $series)
{
  global $user, $create_by, $status, $area;
  global $PHP_SELF, $reminders_enabled, $last_reminded, $reminder_interval;

  $this_page = basename($PHP_SELF);

  // Remind button if you're the owner AND there's a booking awaiting
  // approval AND sufficient time has passed since the last reminder
  // AND we want reminders in the first place
  if (($reminders_enabled) &&
      ($user == $create_by) &&
      ($status & STATUS_AWAITING_APPROVAL) &&
      (working_time_diff(time(), $last_reminded) >= $reminder_interval))
  {
    echo "<tr>\n";
    echo "<td> </td>\n";
    echo "<td>\n";
    generateButton("approve_entry_handler.php", $id, $series, "remind", $this_page . "?id=$id&amp;area=$area",
get_vocab("remind_admin"));
    echo "</td>\n";
    echo "</tr>\n";
  }
}

function generateTextArea($form_action, $id, $series, $action_type, $returl, $submit_value, $caption, $value='')
{
  echo "<tr><td id=\"caption\" colspan=\"2\">$caption:</td></tr>\n";
  echo "<tr>\n";
  echo "<td id=\"note\" colspan=\"2\">\n";
  echo "<form action=\"$form_action\" method=\"post\">\n";
  echo "<fieldset>\n";
  echo "<legend></legend>\n";
  echo "<textarea name=\"note\">" . htmlspecialchars($value) . "</textarea>\n";
  echo "<input type=\"hidden\" name=\"id\" value=\"$id\">\n";
  echo "<input type=\"hidden\" name=\"series\" value=\"$series\">\n";
  echo "<input type=\"hidden\" name=\"returl\" value=\"$returl\">\n";
  echo "<input type=\"hidden\" name=\"action\" value=\"$action_type\">\n";
  echo "<input type=\"submit\" value=\"$submit_value\">\n";
  echo "</fieldset>\n";
  echo "</form>\n";
  echo "</td>\n";
  echo "<tr>\n";
}
```

```
//modified
//function to add drupal users in the serie
function add_drupal_users($id, $day, $month, $year, $area, $capacity)
{
//get all the drupal users name that have the roles that we want and put it in array

  global $auth;

  if ($area == 7){

        //pass the varibales of the booking to javascript
        ?>
        <script language="JavaScript">
            var id = ["<?php echo $id; ?>"];
            var day = ["<?php echo $day; ?>"];
            var month = ["<?php echo $month; ?>"];
            var year = ["<?php echo $year; ?>"];
            var area = ["<?php echo $area; ?>"];
            var id = ["<?php echo $id; ?>"];
        var capacity = ["<?php echo $capacity; ?>"];
            </script>
        <?php
            $conn = sql_connect($auth['drupal']['db_system'],
                                        $auth['drupal']['db_host'],
                                        $auth['drupal']['db_username'],
                                        $auth['drupal']['db_password'],
                                        $auth['drupal']['db_name']);

            //take all users name
            $query = "SELECT name FROM users WHERE uid != 0 ORDER BY name";
            $r = sql_query($query, $conn);
            for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
            {
            $name_all_users[] = $row[0];
            }
            ?>
            <script language="JavaScript">
            var name_all_users = ["<?php echo join("\", \"", $name_all_users); ?>"];
            </script>
            <?php

            //take the users that have the role id 12 (alumne estiu)
            $query = "SELECT uid FROM users_roles WHERE rid = '12'";
            $r = sql_query($query, $conn);
            for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
            {
            $query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
```

```
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_estiu[] = $row_2[0];
}
sort($name_alumn_estiu);
?>
<script language="JavaScript">
var name_alumn_estiu = ["<?php echo join("\", \"", $name_alumn_estiu); ?>"];
</script>
<?php

//take the users that have the role id 10 (alumne matricula grup)
$query = "SELECT uid FROM users_roles WHERE rid = '10'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_matricula_grup[] = $row_2[0];
}
sort($name_alumn_matricula_grup);
?>
<script language="JavaScript">
var name_alumn_matricula_grup = ["<?php echo join("\", \"", $name_alumn_matricula_grup); ?>"];
</script>
<?php

//take the users that have the role id 19 (alumne matricula particular)
$query = "SELECT uid FROM users_roles WHERE rid = '19'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_matricula_particular[] = $row_2[0];
}
sort($name_alumn_matricula_particular);
?>
<script language="JavaScript">
var name_alumn_matricula_particular = ["<?php echo join("\", \"", $name_alumn_matricula_particular); ?>"];
</script>
<?php

//take the users that have the role id 9 (alumne pack hores grup)
$query = "SELECT uid FROM users_roles WHERE rid = '9'";
$r = sql_query($query, $conn);
```

```
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_pack_particular[] = $row_2[0];
}
sort($name_alumn_pack_particular);
?>
<script language="JavaScript">
var name_alumn_pack_particular = ["<?php echo join("\", \"", $name_alumn_pack_particular); ?>"];
</script>
<?php

//take the users that have the role id 18 (alumne pack hores particular)
$query = "SELECT uid FROM users_roles WHERE rid = '18'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_pack_grup[] = $row_2[0];
}
sort($name_alumn_pack_grup);
?>
<script language="JavaScript">
var name_alumn_pack_grup = ["<?php echo join("\", \"", $name_alumn_pack_grup); ?>"];
</script>
<?php

//take the users that have the role id 13 (alumne proves acces academia)
$query = "SELECT uid FROM users_roles WHERE rid = '13'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_proves_acces_academia[] = $row_2[0];
}
sort($name_alumn_proves_acces_academia);
?>
<script language="JavaScript">
var name_alumn_proves_acces_academia = ["<?php echo join("\", \"",
$name_alumn_proves_acces_academia); ?>"];
</script>
<?php
```

```
//take the users that have the role id 16 (alumne selectivitat)
$query = "SELECT uid FROM users_roles WHERE rid = '16'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_selectivitat[] = $row_2[0];
}
sort($name_selectivitat);
?>
<script language="JavaScript">
var name_selectivitat = ["<?php echo join("\", \"", $name_selectivitat); ?>"];
</script>
<?php

//take the users that have the role id 14 (alumne centre client)
$query = "SELECT uid FROM users_roles WHERE rid = '14'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_centre_client[] = $row_2[0];
}
sort($name_centre_client);
?>
<script language="JavaScript">
var name_centre_client = ["<?php echo join("\", \"", $name_centre_client); ?>"];
</script>
<?php

//take the users that have the role id 15 (responsable centre client)
$query = "SELECT uid FROM users_roles WHERE rid = '15'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_responsable_centre_client[] = $row_2[0];
}
sort($name_responsable_centre_client);
?>
<script language="JavaScript">
```

```
var name_responsable_centre_client = ["<?php echo join("\", \"", $name_responsable_centre_client); ?>"];
</script>
<?php

//take the users that have the role id 20 (professor)
$query = "SELECT uid FROM users_roles WHERE rid = '20'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_professor[] = $row_2[0];
}
sort($name_professor);
?>
<script language="JavaScript">
var name_professor = ["<?php echo join("\", \"", $name_professor); ?>"];
</script>
<?php

//take the users that have the role id 4 (professor soci)
$query = "SELECT uid FROM users_roles WHERE rid = '4'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_professor_soci[] = $row_2[0];
}
sort($name_professor_soci);
?>
<script language="JavaScript">
var name_professor_soci = ["<?php echo join("\", \"", $name_professor_soci); ?>"];
</script>
<?php

//take the users that have the role id 17 (alumne offline)
$query = "SELECT uid FROM users_roles WHERE rid = '17'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_alumn_offline[] = $row_2[0];
}
```

```
sort($name_alumn_offline);
?>
<script language="JavaScript">
var name_alumn_offline = ["<?php echo join("\", \"", $name_alumn_offline); ?>"];
</script>
<?php

//take the users that have the role id 40 (universitat)
$query = "SELECT uid FROM users_roles WHERE rid = '40'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_universitat[] = $row_2[0];
}
sort($name_universitat);
?>
<script language="JavaScript">
var name_universitat = ["<?php echo join("\", \"", $name_universitat); ?>"];
</script>
<?php

//take the users that have the role id 38 (primaria)
$query = "SELECT uid FROM users_roles WHERE rid = '38'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_primaria[] = $row_2[0];
}
sort($name_primaria);
?>
<script language="JavaScript">
var name_primaria = ["<?php echo join("\", \"", $name_primaria); ?>"];
</script>
<?php

//take the users that have the role id 39 (eso)
$query = "SELECT uid FROM users_roles WHERE rid = '39'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
```

```
$row_2 = sql_row($r_2, 0, $conn);
$name_eso[] = $row_2[0];
}
sort($name_eso);
?>
<script language="JavaScript">
var name_eso = ["<?php echo join("\", \"", $name_eso); ?>"];
</script>
<?php

//take the users that have the role id 37 (batxillerat)
$query = "SELECT uid FROM users_roles WHERE rid = '37'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_batxillerat[] = $row_2[0];
}
sort($name_batxillerat);
?>
<script language="JavaScript">
var name_batxillerat = ["<?php echo join("\", \"", $name_batxillerat); ?>"];
</script>
<?php

//take the users that have the role id 41 (GES)
$query = "SELECT uid FROM users_roles WHERE rid = '41'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_ges[] = $row_2[0];
}
sort($name_ges);
?>
<script language="JavaScript">
var name_ges = ["<?php echo join("\", \"", $name_ges); ?>"];
</script>
<?php

//take the users that have the role id 42 (Grau Mitja)
$query = "SELECT uid FROM users_roles WHERE rid = '42'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
```

```
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_grau_mitja[] = $row_2[0];
}
sort($name_grau_mitja);
?>
<script language="JavaScript">
var name_grau_mitja = ["<?php echo join("\", \"", $name_grau_mitja); ?>"];
</script>
<?php

//take the users that have the role id 43 (Grau Superior)
$query = "SELECT uid FROM users_roles WHERE rid = '43'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_grau_superior[] = $row_2[0];
}
sort($name_grau_superior);
?>
<script language="JavaScript">
var name_grau_superior = ["<?php echo join("\", \"", $name_grau_superior); ?>"];
</script>
<?php

//take the users that have the role id 44 (Universitat Majors 25)
$query = "SELECT uid FROM users_roles WHERE rid = '44'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_universitat_majors_25[] = $row_2[0];
}
sort($name_universitat_majors_25);
?>
<script language="JavaScript">
var name_universitat_majors_25 = ["<?php echo join("\", \"", $name_universitat_majors_25); ?>"];
</script>
<?php

//take the users that have the role id 45 (Tallers_i_cursos_varis)
```

```
$query = "SELECT uid FROM users_roles WHERE rid = '45'";
$r = sql_query($query, $conn);
for ($i = 0; ($row = sql_row($r, $i, $conn)); $i++)
{
$query_2 = "SELECT name FROM users WHERE uid = '$row[0]'";
$r_2 = sql_query($query_2, $conn);
$row_2 = sql_row($r_2, 0, $conn);
$name_tallers[] = $row_2[0];
}
sort($name_tallers);
?>
<script language="JavaScript">
var name_tallers = ["<?php echo join("\", \"", $name_tallers); ?>"];
</script>

<?php
echo "<br><br><div id=\"div_add_drupal_users\">\n";
echo "<label for=\"add_drupal_users\" style='font-weight:bold'>Afegeix Usuaris a la reserva
multiple:</label>\n";

//use a javascript function to creat a subselect for the roles that have subselect options only
?>
<script language="JavaScript">
drupal_client_level_roles = new Array();
drupal_client_level_roles[0] = new Array("alumn_pack_grup", "Tots");
drupal_client_level_roles[1] = new Array("alumn_pack_particular", "Tots");
drupal_client_level_roles[2] = new Array("alumn_matricula_particular", "Tots");
drupal_client_level_roles[3] = new Array("alumn_matricula_grup", "Tots");
drupal_client_level_roles[4] = new Array("alumn_matricula_particular", "universitat");
drupal_client_level_roles[5] = new Array("alumn_matricula_particular", "batxillerat");
drupal_client_level_roles[6] = new Array("alumn_matricula_particular", "eso");
drupal_client_level_roles[7] = new Array("alumn_matricula_particular", "primaria");
drupal_client_level_roles[8] = new Array("alumn_pack_particular", "universitat");
drupal_client_level_roles[9] = new Array("alumn_pack_particular", "batxillerat");
drupal_client_level_roles[10] = new Array("alumn_pack_particular", "eso");
drupal_client_level_roles[11] = new Array("alumn_pack_particular", "primaria");
drupal_client_level_roles[12] = new Array("alumn_pack_grup", "batxillerat");
drupal_client_level_roles[13] = new Array("alumn_pack_grup", "eso");
drupal_client_level_roles[14] = new Array("alumn_pack_grup", "primaria");
drupal_client_level_roles[15] = new Array("alumn_matricula_grup", "batxillerat");
drupal_client_level_roles[16] = new Array("alumn_matricula_grup", "eso");
drupal_client_level_roles[17] = new Array("alumn_matricula_grup", "primaria");
drupal_client_level_roles[18] = new Array("alumn_centre_client", "Tots");
drupal_client_level_roles[19] = new Array("alumn_centre_client", "ges");
drupal_client_level_roles[20] = new Array("alumn_centre_client", "grau mitja");
drupal_client_level_roles[21] = new Array("alumn_centre_client", "grau superior");
drupal_client_level_roles[22] = new Array("alumn_centre_client", "universitat majors");
drupal_client_level_roles[22] = new Array("alumn_centre_client", "tallers");
```

```
drupal_client_level_roles[23] = new Array("alumn_proves_acces_academia", "Tots");
drupal_client_level_roles[24] = new Array("alumn_proves_acces_academia", "ges");
drupal_client_level_roles[25] = new Array("alumn_proves_acces_academia", "grau mitja");
drupal_client_level_roles[26] = new Array("alumn_proves_acces_academia", "grau superior");
drupal_client_level_roles[27] = new Array("alumn_proves_acces_academia", "universitat majors");

    function DrupalUsersAndSubRoles(role)
    {
     var opRoles = document.getElementById("f_sub_roles").options
     var opUsers = document.getElementById("f_filtered_users").options
     var noOptions = true;
     while(opRoles.length > 0) {
      opRoles.remove(0);
     }

     while(opUsers.length > 0) {
      opUsers.remove(0);
     }

     if(role.length > 0) {

      for( var i=0; i < drupal_client_level_roles.length; i++ ) {
            if(drupal_client_level_roles[i][0] == role) {
                if (noOptions == true) {
                opRoles.add(new Option("Selecciona el nivell de l'usuari", ""));
                }
             opRoles.add(new Option(drupal_client_level_roles[i][1], drupal_client_level_roles[i][1]));
             noOptions = false;
            }
      }
      if (noOptions == true) {
            if (role == "general"){
             for( var j=0; j < name_all_users.length; j++ ) {
               opUsers.add(new Option(name_all_users[j], name_all_users[j]));
             }
            }
            if (role == "alumn_estiu"){
             for( var j=0; j < name_alumn_estiu.length; j++ ) {
               opUsers.add(new Option(name_alumn_estiu[j], name_alumn_estiu[j]));
             }
            }
            if (role == "alumn_selectivitat"){
             for( var j=0; j < name_alumn_selectivitat.length; j++ ) {
               opUsers.add(new Option(name_alumn_selectivitat[j], name_alumn_selectivitat[j]));
             }
            }
            if (role == "responsable_centre_client"){
             for( var j=0; j < name_responsable_centre_client.length; j++ ) {
```

```
            opUsers.add(new Option(name_responsable_centre_client[j], name_responsable_centre_client[j]));
          }
        }
        if (role == "professor"){
          for( var j=0; j < name_professor.length; j++ ) {
            opUsers.add(new Option(name_professor[j], name_professor[j]));
          }
        }
        if (role == "professor_soci"){
          for( var j=0; j < name_professor_soci.length; j++ ) {
            opUsers.add(new Option(name_professor_soci[j], name_professor_soci[j]));
          }
        }
        if (role == "alumn_offline"){
          for( var j=0; j < name_alumn_offline.length; j++ ) {
            opUsers.add(new Option(name_alumn_offline[j], name_alumn_offline[j]));
          }
        }
        if (role == "primaria"){
          for( var j=0; j < name_primaria.length; j++ ) {
            opUsers.add(new Option(name_primaria[j], name_primaria[j]));
          }
        }
        if (role == "eso"){
          for( var j=0; j < name_eso.length; j++ ) {
            opUsers.add(new Option(name_eso[j], name_eso[j]));
          }
        }
        if (role == "batxillerat"){
          for( var j=0; j < name_batxillerat.length; j++ ) {
            opUsers.add(new Option(name_batxillerat[j], name_batxillerat[j]));
          }
        }
        if (role == "universitat"){
          for( var j=0; j < name_universitat.length; j++ ) {
            opUsers.add(new Option(name_universitat[j], name_universitat[j]));
          }
        }
        if (role == "tallers"){
          for( var j=0; j < name_tallers.length; j++ ) {
            opUsers.add(new Option(name_tallers[j], name_tallers[j]));
          }
        }
    }
  }
}
function DrupalUsersForSubRoles(subrole)
{
```

```
//matricula grup selected index = 3, mat part = 4 pack grup =5 i pack part =6
var selectedRole = document.getElementById("f_drupal_roles").selectedIndex
	var opUsers = document.getElementById("f_filtered_users").options
	while(opUsers.length > 0) {
	  opUsers.remove(0);
	}
	if (subrole == "Tots") {
		if (selectedRole == 3){
		  for( var j=0; j < name_alumn_matricula_grup.length; j++ ) {
			opUsers.add(new Option(name_alumn_matricula_grup[j],
name_alumn_matricula_grup[j]));
		  }
		}
		if (selectedRole == 4){
		  for( var j=0; j < name_alumn_matricula_particular.length; j++ ) {
			opUsers.add(new Option(name_alumn_matricula_particular[j],
name_alumn_matricula_particular[j]));
		  }
		}
		if (selectedRole == 5){
		  for( var j=0; j < name_alumn_pack_grup.length; j++ ) {
			opUsers.add(new Option(name_alumn_pack_grup[j], name_alumn_pack_grup[j]));
		  }
		}
		if (selectedRole == 6){
		  for( var j=0; j < name_alumn_pack_particular.length; j++ ) {
			opUsers.add(new Option(name_alumn_pack_particular[j],
name_alumn_pack_particular[j]));
		  }
		}
		if (selectedRole == 7){
		  for( var j=0; j < name_alumn_proves_acces_academia.length; j++ ) {
			opUsers.add(new Option(name_alumn_proves_acces_academia[j],
name_alumn_proves_acces_academia[j]));
		  }
		}
		if (selectedRole == 9){
		  for( var j=0; j < name_centre_client.length; j++ ) {
			opUsers.add(new Option(name_centre_client[j], name_centre_client[j]));
		  }
		}
	}
    if (subrole == "ges") {
		if (selectedRole == 7){
		  var name_alumn_proves_academia_ges = new Array();
		  var index = 0;
		  //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
```

```
                        for( var k=0; k < name_alumn_proves_acces_academia.length; k++ ) {
                                if(name_ges.indexOf(name_alumn_proves_acces_academia[k]) != -1){
                                        name_alumn_proves_academia_ges[index] =
name_alumn_proves_acces_academia[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_proves_academia_ges.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_proves_academia_ges[j],
name_alumn_proves_academia_ges[j]));
                                }
                        }
                       }
                       if (selectedRole == 9){
                        var name_alumn_extern_ges = new Array();
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                        for( var k=0; k < name_centre_client.length; k++ ) {
                                if(name_ges.indexOf(name_centre_client[k]) != -1){
                                        name_alumn_extern_ges[index] = name_centre_client[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_extern_ges.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_extern_ges[j],
name_alumn_extern_ges[j]));
                                }
                        }
                  }
                 }
                 if (subrole == "grau mitja") {
                        if (selectedRole == 7){
                         var name_alumn_proves_academia_grau_mitja = new Array();
                         var index = 0;
                         //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                         for( var k=0; k < name_alumn_proves_acces_academia.length; k++ ) {
                                 if(name_grau_mitja.indexOf(name_alumn_proves_acces_academia[k]) != -1){
                                         name_alumn_proves_academia_grau_mitja[index] =
name_alumn_proves_acces_academia[k];
                                         index++;
                                 }
                                 for( var j=0; j < name_alumn_proves_academia_grau_mitja.length; j++ ) {
                                         opUsers.add(new Option(name_alumn_proves_academia_grau_mitja[j],
name_alumn_proves_academia_grau_mitja[j]));
                                 }
                         }
                        }
                        if (selectedRole == 9){
```

```
                        var name_alumn_extern_grau_mitja = new Array();
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                        for( var k=0; k < name_centre_client.length; k++ ) {
                                if(name_grau_mitja.indexOf(name_centre_client[k]) != -1){
                                        name_alumn_extern_grau_mitja[index] = name_centre_client[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_extern_grau_mitja.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_extern_grau_mitja[j],
name_alumn_extern_grau_mitja[j]));
                                }
                        }
                    }
                }
                if (subrole == "grau superior") {
                        if (selectedRole == 7){
                        var name_alumn_proves_academia_grau_superior = new Array();
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                        for( var k=0; k < name_alumn_proves_acces_academia.length; k++ ) {
                                if(name_grau_superior.indexOf(name_alumn_proves_acces_academia[k]) != -1){
                                        name_alumn_proves_academia_grau_superior[index] =
name_alumn_proves_acces_academia[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_proves_academia_grau_superior.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_proves_academia_grau_superior[j],
name_alumn_proves_academia_grau_superior[j]));
                                }
                        }
                        }
                        if (selectedRole == 9){
                        var name_alumn_extern_grau_superior = new Array();
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                        for( var k=0; k < name_centre_client.length; k++ ) {
                                if(name_grau_superior.indexOf(name_centre_client[k]) != -1){
                                        name_alumn_extern_grau_superior[index] = name_centre_client[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_extern_grau_superior.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_extern_grau_superior[j],
name_alumn_extern_grau_superior[j]));
                                }
```

```
                        }
                    }
                }
                if (subrole == "universitat majors") {
                    if (selectedRole == 7){
                      var name_alumn_proves_academia_universitat_majors_25 = new Array();
                      var index = 0;
                      //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                      for( var k=0; k < name_alumn_proves_acces_academia.length; k++ ) {
                          if(name_universitat_majors_25.indexOf(name_alumn_proves_acces_academia[k]) !
= -1){
                              name_alumn_proves_academia_universitat_majors_25[index] =
name_alumn_proves_acces_academia[k];
                              index++;
                          }
                          for( var j=0; j < name_alumn_proves_academia_universitat_majors_25.length; j+
+ ) {
                              opUsers.add(new
Option(name_alumn_proves_academia_universitat_majors_25[j],
name_alumn_proves_academia_universitat_majors_25[j]));
                          }
                      }
                    }
                    if (selectedRole == 9){
                      var name_alumn_extern_universitat_majors_25 = new Array();
                      var index = 0;
                      //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                      for( var k=0; k < name_centre_client.length; k++ ) {
                          if(name_universitat_majors_25.indexOf(name_centre_client[k]) != -1){
                              name_alumn_extern_universitat_majors_25[index] =
name_centre_client[k];
                              index++;
                          }
                          for( var j=0; j < name_alumn_extern_universitat_majors_25.length; j++ ) {
                              opUsers.add(new Option(name_alumn_extern_universitat_majors_25[j],
name_alumn_extern_universitat_majors_25[j]));
                          }
                      }
                    }
                }
            if (subrole == "tallers") {
                //per tallers, nomes alumne extern
                    if (selectedRole == 9){
                      var name_alumn_extern_tallers = new Array();
                      var index = 0;
```

```
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                        for( var k=0; k < name_centre_client.length; k++ ) {
                                if(name_tallers.indexOf(name_centre_client[k]) != -1){
                                        name_alumn_extern_tallers[index] = name_centre_client[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_extern_tallers.length; j++ ) {
                                         opUsers.add(new Option(name_alumn_extern_tallers[j],
name_alumn_extern_tallers[j]));
                                }
                         }
                        }
                }
                if (subrole == "universitat") {
                        if (selectedRole == 3){
                         var name_alumn_matricula_grup_universitat = new Array();
                         var index = 0;
                         //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                         for( var k=0; k < name_alumn_matricula_grup.length; k++ ) {
                                if(name_universitat.indexOf(name_alumn_matricula_grup[k]) != -1){
                                        name_alumn_matricula_grup_universitat[index] =
name_alumn_matricula_grup[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_matricula_grup_universitat.length; j++ ) {
                                         opUsers.add(new Option(name_alumn_matricula_grup_universitat[j],
name_alumn_matricula_grup_universitat[j]));
                                }
                         }
                        }
                        if (selectedRole == 4){
                         var name_alumn_matricula_particular_universitat = new Array();
                         var index = 0;
                         //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                         for( var k=0; k < name_alumn_matricula_particular.length; k++ ) {
                                if(name_universitat.indexOf(name_alumn_matricula_particular[k]) != -1){
                                        name_alumn_matricula_particular_universitat[index] =
name_alumn_matricula_particular[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_matricula_particular_universitat.length; j++ ) {
                                         opUsers.add(new Option(name_alumn_matricula_particular_universitat[j],
name_alumn_matricula_particular_universitat[j]));
                                }
                         }
```

```
                        }
                        if (selectedRole == 5){
                          var name_alumn_pack_grup_universitat = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                          for( var k=0; k < name_alumn_pack_grup.length; k++ ) {
                                  if(name_universitat.indexOf(name_alumn_pack_grup[k]) != -1){
                                          name_alumn_pack_grup_universitat[index] = name_alumn_pack_grup[k];
                                          index++;
                                  }
                                  for( var j=0; j < name_alumn_pack_grup_universitat.length; j++ ) {
                                           opUsers.add(new Option(name_alumn_pack_grup_universitat[j],
name_alumn_pack_grup_universitat[j]));
                                  }
                          }
                        }
                        if (selectedRole == 6){
                          var name_alumn_pack_particular_universitat = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                          for( var k=0; k < name_alumn_pack_particular.length; k++ ) {
                                  if(name_universitat.indexOf(name_alumn_pack_particular[k]) != -1){
                                          name_alumn_pack_particular_universitat[index] =
name_alumn_pack_particular[k];
                                          index++;
                                  }
                                  for( var j=0; j < name_alumn_pack_particular_universitat.length; j++ ) {
                                           opUsers.add(new Option(name_alumn_pack_particular_universitat[j],
name_alumn_pack_particular_universitat[j]));
                                  }
                          }
                        }
                }
                if (subrole == "batxillerat") {
                        if (selectedRole == 3){
                          var name_alumn_matricula_grup_batxillerat = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                          for( var k=0; k < name_alumn_matricula_grup.length; k++ ) {
                                  if(name_batxillerat.indexOf(name_alumn_matricula_grup[k]) != -1){
                                          name_alumn_matricula_grup_batxillerat[index] =
name_alumn_matricula_grup[k];
                                          index++;
                                  }
                                  for( var j=0; j < name_alumn_matricula_grup_batxillerat.length; j++ ) {
```

```
                                        opUsers.add(new Option(name_alumn_matricula_grup_batxillerat[j],
name_alumn_matricula_grup_batxillerat[j]));
                                    }
                                }
                            }
                            if (selectedRole == 4){
                                var name_alumn_matricula_particular_batxillerat = new Array();
                                var index = 0;
                                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex batxillerat)
                                for( var k=0; k < name_alumn_matricula_particular.length; k++ ) {
                                    if(name_batxillerat.indexOf(name_alumn_matricula_particular[k]) != -1){
                                        name_alumn_matricula_particular_batxillerat[index] =
name_alumn_matricula_particular[k];
                                        index++;
                                    }
                                    for( var j=0; j < name_alumn_matricula_particular_batxillerat.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_matricula_particular_batxillerat[j],
name_alumn_matricula_particular_batxillerat[j]));
                                    }
                                }
                            }
                            if (selectedRole == 5){
                                var name_alumn_pack_grup_batxillerat = new Array();
                                var index = 0;
                                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex batxillerat)
                                for( var k=0; k < name_alumn_pack_grup.length; k++ ) {
                                    if(name_batxillerat.indexOf(name_alumn_pack_grup[k]) != -1){
                                        name_alumn_pack_grup_batxillerat[index] = name_alumn_pack_grup[k];
                                        index++;
                                    }
                                    for( var j=0; j < name_alumn_pack_grup_batxillerat.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_pack_grup_batxillerat[j],
name_alumn_pack_grup_batxillerat[j]));
                                    }
                                }
                            }
                            if (selectedRole == 6){
                                var name_alumn_pack_particular_batxillerat = new Array();
                                var index = 0;
                                //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex batxillerat)
                                for( var k=0; k < name_alumn_pack_particular.length; k++ ) {
                                    if(name_batxillerat.indexOf(name_alumn_pack_particular[k]) != -1){
                                        name_alumn_pack_particular_batxillerat[index] =
name_alumn_pack_particular[k];
                                        index++;
```

```
                                        }
                                        for( var j=0; j < name_alumn_pack_particular_batxillerat.length; j++ ) {
                                                opUsers.add(new Option(name_alumn_pack_particular_batxillerat[j],
name_alumn_pack_particular_batxillerat[j]));
                                        }
                                }
                        }
                }
                if (subrole == "eso") {
                        if (selectedRole == 3){
                          var name_alumn_matricula_grup_eso = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                          for( var k=0; k < name_alumn_matricula_grup.length; k++ ) {
                                if(name_eso.indexOf(name_alumn_matricula_grup[k]) != -1){
                                        name_alumn_matricula_grup_eso[index] = name_alumn_matricula_grup[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_matricula_grup_eso.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_matricula_grup_eso[j],
name_alumn_matricula_grup_eso[j]));
                                }
                          }
                        }
                        if (selectedRole == 4){
                          var name_alumn_matricula_particular_eso = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                          for( var k=0; k < name_alumn_matricula_particular.length; k++ ) {
                                if(name_eso.indexOf(name_alumn_matricula_particular[k]) != -1){
                                        name_alumn_matricula_particular_eso[index] =
name_alumn_matricula_particular[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_matricula_particular_eso.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_matricula_particular_eso[j],
name_alumn_matricula_particular_eso[j]));
                                }
                          }
                        }
                        if (selectedRole == 5){
                          var name_alumn_pack_grup_eso = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                          for( var k=0; k < name_alumn_pack_grup.length; k++ ) {
```

```
                                        if(name_eso.indexOf(name_alumn_pack_grup[k]) != -1){
                                                name_alumn_pack_grup_eso[index] = name_alumn_pack_grup[k];
                                                index++;
                                        }
                                        for( var j=0; j < name_alumn_pack_grup_eso.length; j++ ) {
                                                opUsers.add(new Option(name_alumn_pack_grup_eso[j],
name_alumn_pack_grup_eso[j]));
                                        }
                                }
                        }
                        if (selectedRole == 6){
                          var name_alumn_pack_particular_eso = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex eso)
                          for( var k=0; k < name_alumn_pack_particular.length; k++ ) {
                                        if(name_eso.indexOf(name_alumn_pack_particular[k]) != -1){
                                                name_alumn_pack_particular_eso[index] =
name_alumn_pack_particular[k];
                                                index++;
                                        }
                                        for( var j=0; j < name_alumn_pack_particular_eso.length; j++ ) {
                                                opUsers.add(new Option(name_alumn_pack_particular_eso[j],
name_alumn_pack_particular_eso[j]));
                                        }
                                }
                        }
                }
                if (subrole == "primaria") {
                        if (selectedRole == 3){
                          var name_alumn_matricula_grup_primaria = new Array();
                          var index = 0;
                          //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex universitat)
                          for( var k=0; k < name_alumn_matricula_grup.length; k++ ) {
                                        if(name_primaria.indexOf(name_alumn_matricula_grup[k]) != -1){
                                                name_alumn_matricula_grup_primaria[index] =
name_alumn_matricula_grup[k];
                                                index++;
                                        }
                                        for( var j=0; j < name_alumn_matricula_grup_primaria.length; j++ ) {
                                                opUsers.add(new Option(name_alumn_matricula_grup_primaria[j],
name_alumn_matricula_grup_primaria[j]));
                                        }
                                }
                        }
                        if (selectedRole == 4){
                          var name_alumn_matricula_particular_primaria = new Array();
```

```
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex primaria)
                        for( var k=0; k < name_alumn_matricula_particular.length; k++ ) {
                                if(name_primaria.indexOf(name_alumn_matricula_particular[k]) != -1){
                                        name_alumn_matricula_particular_primaria[index] =
name_alumn_matricula_particular[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_matricula_particular_primaria.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_matricula_particular_primaria[j],
name_alumn_matricula_particular_primaria[j]));
                                }
                        }
                       }
                       if (selectedRole == 5){
                        var name_alumn_pack_grup_primaria = new Array();
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex primaria)
                        for( var k=0; k < name_alumn_pack_grup.length; k++ ) {
                                if(name_primaria.indexOf(name_alumn_pack_grup[k]) != -1){
                                        name_alumn_pack_grup_primaria[index] = name_alumn_pack_grup[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_pack_grup_primaria.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_pack_grup_primaria[j],
name_alumn_pack_grup_primaria[j]));
                                }
                        }
                       }
                       if (selectedRole == 6){
                        var name_alumn_pack_particular_primaria = new Array();
                        var index = 0;
                        //recorrem l'array de noms del rol principal (p.ex alumn matricula grup) i generem un nou
array que contingui els noms que tambe apareixen en l'array del rol de nivell (p.ex primaria)
                        for( var k=0; k < name_alumn_pack_particular.length; k++ ) {
                                if(name_primaria.indexOf(name_alumn_pack_particular[k]) != -1){
                                        name_alumn_pack_particular_primaria[index] =
name_alumn_pack_particular[k];
                                        index++;
                                }
                                for( var j=0; j < name_alumn_pack_particular_primaria.length; j++ ) {
                                        opUsers.add(new Option(name_alumn_pack_particular_primaria[j],
name_alumn_pack_particular_primaria[j]));
                                }
                         }
                       }
```

```
                }
            }
        function GoToAfegirUsuari()
            {
          var separador = ' | ';
          var usuari_seleccionat = document.getElementById("f_filtered_users").value + separador +
document.getElementById("f_drupal_roles").value
          if (document.getElementById("f_sub_roles").value.length > 0){
           usuari_seleccionat = usuari_seleccionat + separador + document.getElementById("f_sub_roles").value
          }
          var usuari_seleccionat_sol = document.getElementById("f_filtered_users").value
          if (usuari_seleccionat.length == 0 ){
           alert('has de Seleccionar un usuari');
          }
          else if (usuari_seleccionat_sol ==  current_user){
           alert('No fa falta que t introdueixis a la reserva');
          }
          else{
           var tr=entry.getElementsByTagName("tr");
                var td;
           var existeix_usuari = false;

                for(var i=1; i<tr.length; i++) {
                        td=tr[i].getElementsByTagName("td");

                        for(var j=0; j<td.length; j++) {
                  var user_name_only = td[j].innerHTML.substr(0,td[j].innerHTML.indexOf(" | "));
                                if(user_name_only==usuari_seleccionat_sol) {
                                        existeix_usuari=true;
                           alert('Aquest Usuari ja esta a la Reserva');
                                }
                        }
                }

          if (existeix_usuari == false) {
            window.location.href = "afegir_usuari_reserva_multiple.php?id=" + id + "&day=" + day + "&month=" +
month + "&year=" + year + "&area=" + area + "&usuari=" + usuari_seleccionat + "&capacity=" + capacity;
          }
          }
           }
          </script>

          <select id="f_drupal_roles" onchange="DrupalUsersAndSubRoles(this.options[this.selectedIndex].value);">
          <option value="">Selecciona un tipus d'usuari</option>
          <option value="general">Tots els usuaris (inclos registrats no alumnes)</option>
          <option value="alumn_estiu">Alumne estiu</option>
          <option value="alumn_matricula_grup">Alumne matricula grup</option>
          <option value="alumn_matricula_particular">Alumne matricula particular</option>
```

```
        <option value="alumn_pack_grup">Alumne pack grup</option>
        <option value="alumn_pack_particular">Alumne pack particular</option>
        <option value="alumn_proves_acces_academia">Alumne proves acces academia</option>
        <option value="alumn_selectivitat">Alumne selectivitat</option>
        <option value="alumn_centre_client">Alumne centre client</option>
        <option value="responsable_centre_client">responsable centre client</option>
        <option value="professor">professor</option>
        <option value="professor_soci">professor soci</option>
        <option value="alumn_offline">alumne offline</option>
        <option value="universitat">Tots els alumnes d'universitat</option>
        <option value="batxillerat">Tots els alumnes de batxillerat</option>
        <option value="eso">Tots els alumnes d'eso</option>
        <option value="primaria">Tots els alumnes de primaria</option>
    <option value="tallers" title="Inclou els que no tenen rol de alumne centre client">Alumnes de Tallers i cursos
Varis</option>
        </select>
    <br>
                    
                 
                 
         
        <select id="f_sub_roles" onchange="DrupalUsersForSubRoles(this.options[this.selectedIndex].value);">
        </select>
    <br>
                    
                 
                 
         
    <select id="f_filtered_users" size="5">
        </select>
        <br><br>
                 
                 
                 
        
    <input type="submit" id="afegir_usuaris" value="Afegeix l' Usuari a la Reserva Multiple"
onclick="GoToAfegirUsuari();">
        <br>
                 
                 
                 
        
L'usuari de la llista s'afegira a aquesta reserva i totes les reserves futures de la mateixa classe i franja horaria (reserva
multiple), una vegada s'apreti el boto Superior
       <?php
        echo "<br>";
    echo "<br>";
        echo "</div>\n";
```

```
  }
}

// Get non-standard form variables
//
// If $series is TRUE, it means that the $id is the id of an
// entry in the repeat table.  Otherwise it's from the entry table.
$id = get_form_var('id', 'int');
$series = get_form_var('series', 'int');
$action = get_form_var('action', 'string');
$returl = get_form_var('returl', 'string');
$error = get_form_var('error', 'string');

// Check the user is authorised for this page
checkAuthorised();

// Also need to know whether they have admin rights
$user = getUserName();

$is_admin = (authGetUserLevel($user) >= 2);
// You're only allowed to make repeat bookings if you're an admin
// or else if $auth['only_admin_can_book_repeat'] is not set
$repeats_allowed = $is_admin || empty($auth['only_admin_can_book_repeat']);

$row = mrbsGetBookingInfo($id, $series);

// Get the area settings for the entry's area.    In particular we want
// to know how to display private/public bookings in this area.
get_area_settings($row['area_id']);

// Work out whether the room or area is disabled
$room_disabled = $row['room_disabled'] || $row['area_disabled'];
// Get the status
$status = $row['status'];
// Get the creator
$create_by = $row['create_by'];
// Work out whether this event should be kept private
$private = $row['status'] & STATUS_PRIVATE;
$writeable = getWritable($row['create_by'], $user, $row['room_id']);
$keep_private = (is_private_event($private) && !$writeable);

// Work out when the last reminder was sent
$last_reminded = (empty($row['reminded'])) ? $row['last_updated'] : $row['reminded'];


if ($series == 1)
{
  $repeat_id = $id;  // Save the repeat_id
```

```
  // I also need to set $id to the value of a single entry as it is a
  // single entry from a series that is used by del_entry.php and
  // edit_entry.php
  // So I will look for the first entry in the series where the entry is
  // as per the original series settings
  $sql = "SELECT id
          FROM $tbl_entry
          WHERE repeat_id=\"$id\" AND entry_type=" . ENTRY_RPT_ORIGINAL . "
          ORDER BY start_time
          LIMIT 1";
  $id = sql_query1($sql);
  if ($id < 1)
  {
    // if all entries in series have been modified then
    // as a fallback position just select the first entry
    // in the series
    // hopefully this code will never be reached as
    // this page will display the start time of the series
    // but edit_entry.php will display the start time of the entry
    $sql = "SELECT id
            FROM $tbl_entry
            WHERE repeat_id=\"$id\"
            ORDER BY start_time
            LIMIT 1";
    $id = sql_query1($sql);
  }
  $repeat_info_time = $row['repeat_info_time'];
  $repeat_info_user = $row['repeat_info_user'];
  $repeat_info_text = $row['repeat_info_text'];
}
else
{
  $repeat_id = $row['repeat_id'];

  $entry_info_time = $row['entry_info_time'];
  $entry_info_user = $row['entry_info_user'];
  $entry_info_text = $row['entry_info_text'];
}


// PHASE 2 - EXPORTING ICALENDAR FILES
// -----------------------------------

if (isset($action) && ($action == "export"))
{
  if ($keep_private  || $enable_periods)
  {
    // should never normally be able to get here, but if we have then
```

```
    // go somewhere safe.
    header("Location: index.php");
    exit;
   }
   else
   {
    require_once "functions_ical.inc";
    header("Content-Type: application/ics;  charset=" . get_charset(). "; name=\"" . $mail_settings['ics_filename'] .
".ics\"");
    header("Content-Disposition: attachment; filename=\"" . $mail_settings['ics_filename'] . ".ics\"");

    // Construct the SQL query
    $sql = "SELECT E.*, "
         .  sql_syntax_timestamp_to_unix("E.timestamp") . " AS last_updated, "
         . "A.area_name, R.room_name, "
         . "A.approval_enabled, A.confirmation_enabled";
    if ($series)
    {
      // If it's a series we want the repeat information
      $sql .= ", T.rep_type, T.end_date, T.rep_opt, T.rep_num_weeks";
    }
    $sql .= " FROM $tbl_area A, $tbl_room R, $tbl_entry E";
    if ($series)
    {
      $sql .= ", $tbl_repeat T"
            . " WHERE E.repeat_id=$repeat_id"
            . " AND E.repeat_id=T.id"
            . " ORDER BY E.ical_recur_id";
    }
    else
    {
      $sql .= " WHERE E.id=$id";
    }
    $res = sql_query($sql);

    // Export the calendar
    export_icalendar($res, $keep_private);
    exit;
   }
}

// PHASE 1 - VIEW THE ENTRY
// ------------------------

print_header($day, $month, $year, $area, isset($room) ? $room : "");


// Need to tell all the links where to go back to after an edit or delete
```

```
if (!isset($returl))
{
  if (isset($HTTP_REFERER))
  {
    $returl = $HTTP_REFERER;
  }
  // If we haven't got a referer (eg we've come here from an email) then construct
  // a sensible place to go to afterwards
  else
  {
    switch ($default_view)
    {
      case "month":
        $returl = "month.php";
        break;
      case "week":
        $returl = "week.php";
        break;
      default:
        $returl = "day.php";
    }
    $returl .= "?year=$year&month=$month&day=$day&area=$area";
  }
}
$link_returl = urlencode($returl);  // for use in links

if (empty($series))
{
  $series = 0;
}
else
{
  $series = 1;
}


// Now that we know all the data we start drawing it

echo "<h3" . (($keep_private && $is_private_field['entry.name']) ? " class=\"private\"" : "") . ">\n";
echo ($keep_private && $is_private_field['entry.name']) ? "[" . get_vocab("private") . "]" :
htmlspecialchars($row['name']);
if (is_private_event($private) && $writeable)
{
  echo ' ('.get_vocab("private").')';
}
echo "</h3>\n";
```

```
echo "<table id=\"entry\">\n";

// Output any error messages
if (!empty($error))
{
  echo "<tr><td> </td><td class=\"error\">" . get_vocab($error) . "</td></tr>\n";
}

// If bookings require approval, and the room is enabled, put the buttons
// to do with managing the bookings in the footer
if ($approval_enabled && !$room_disabled && ($status & STATUS_AWAITING_APPROVAL))
{
  echo "<tfoot id=\"approve_buttons\">\n";
  // PHASE 2 - REJECT
  if (isset($action) && ($action == "reject"))
  {
    // del_entry expects the id of a member of a series
    // when deleting a series and not the repeat_id
    generateTextArea("del_entry.php", $id, $series,
                     "reject", $returl,
                     get_vocab("reject"),
                     get_vocab("reject_reason"));
  }
  // PHASE 2 - MORE INFO
  elseif (isset($action) && ($action == "more_info"))
  {
    // but approve_entry_handler expects the id to be a repeat_id
    // if $series is true (ie behaves like the rest of MRBS).
    // Sometime this difference in behaviour should be rationalised
    // because it is very confusing!
    $target_id = ($series) ? $repeat_id : $id;
    $info_time = ($series) ? $repeat_info_time : $entry_info_time;
    $info_user = ($series) ? $repeat_info_user : $entry_info_user;
    $info_text = ($series) ? $repeat_info_text : $entry_info_text;

    if (empty($info_time))
    {
      $value = '';
    }
    else
    {
      $value = get_vocab("sent_at") . time_date_string($info_time);
      if (!empty($info_user))
      {
        $value .= "\n" . get_vocab("by") . " $info_user";
      }
      $value .= "\n----\n";
      $value .= $info_text;
```

```
    }
    generateTextArea("approve_entry_handler.php", $target_id, $series,
                     "more_info", $returl,
                     get_vocab("send"),
                     get_vocab("request_more_info"),
                     $value);
  }
  // PHASE 1 - first time through this page
  else
  {
    // Buttons for those who are allowed to approve this booking
    if (auth_book_admin($user, $row['room_id']))
    {
      if (!$series)
      {
        generateApproveButtons($id, FALSE);
      }
      if (!empty($repeat_id) || $series)
      {
        generateApproveButtons($repeat_id, TRUE);
      }
    }
    // Buttons for the owner of this booking
    elseif ($user == $create_by)
    {
      generateOwnerButtons($id, $series);
    }
    // Others don't get any buttons
    else
    {
      // But valid HTML requires that there's something inside the <tfoot></tfoot>
      echo "<tr><td></td><td></td></tr>\n";
    }
  }
  echo "</tfoot>\n";
}
//modified (S'utilitza a create details row de functions_view.inc)
global $number_alumnes_for_view;
$number_alumnes_for_view = 0;
echo create_details_body($row, TRUE, $keep_private, $room_disabled, $repeat_id, $id, $day, $month, $year, $area);

//modified
//only show rows with values
?>
</table>
<script type="text/javascript" language="JavaScript">
function checkTable() {
        var tr=entry.getElementsByTagName("tr");
```

```
        var td;

        for(var i=1; i<tr.length; i++) {
                td=tr[i].getElementsByTagName("td");

                for(var j=0; j<td.length; j++) {
                        if(td[j].innerHTML=="") {
                                tr[i].style.display="none"; //invisible, doesn't take up space
                        }
                }
        }
}
checkTable();
</script>

<?php
//modified
//get the capacity of the room that of the booking
  $sql_1 = "SELECT mrbs_room.capacity
        FROM mrbs_room, mrbs_entry
        WHERE mrbs_entry.id=\"$id\" AND mrbs_entry.room_id=mrbs_room.id
        LIMIT 1";
  $capacity = sql_query1($sql_1);
//call the function to add users to a serie
add_drupal_users($id, $day, $month, $year, $area, $capacity); ?>

<div id="view_entry_nav">
  <?php
  // Only show the links for Edit and Delete if the room is enabled.    We're
  // allowed to view and copy existing bookings in disabled rooms, but not to
  // modify or delete them.
  if (!$room_disabled)
  {
    // Edit and Edit Series
    echo "<div>\n";
    if (!$series)
    {
      echo "<a href=\"edit_entry.php?id=$id&amp;returl=$link_returl\">". get_vocab("editentry") ."</a>";
    }
    if (!empty($repeat_id)  && !$series && $repeats_allowed)
    {
      echo " - ";
    }
    if ((!empty($repeat_id) || $series) && $repeats_allowed)
    {
      //modified
      //evitem que es modifiqui una serie per a que no se sobreescriguin els camps alumne_x
```

```
    if ($area == 7){
      echo "No es pot editar la reserva multiple. Si vols pots afegir o borrar usuaris utilitzant els enllaços ' Afegeix l' Usuari a la Reserva Multiple' o 'Borra l'Usuari de la Reserva Multiple' o pots editar els camps de les reserves que necessitis d'una a una, utilitzant l'enllaç 'editar reserva'";
    }
    else {
      echo "<a href=\"edit_entry.php?id=$id&amp;edit_type=series&amp;day=$day&amp;month=$month&amp;year=$year&amp;returl=$link_returl\">".get_vocab("editseries")." (Es canviaran les dades futures i pasades)</a>";
    }
  }
  echo "</div>\n";

  // Delete and Delete Series
  echo "<div>\n";
  if (!$series)
  {
    echo "<a href=\"del_entry.php?id=$id&amp;series=0&amp;returl=$link_returl\" onclick=\"return confirm('".get_vocab("confirmdel")."');\">".get_vocab("deleteentry")."</a>";
  }
  if (!empty($repeat_id) && !$series && $repeats_allowed)
  {
    echo " - ";
  }
  if ((!empty($repeat_id) || $series) && $repeats_allowed)
  {
    echo "<a href=\"del_entry.php?id=$id&amp;series=1&amp;day=$day&amp;month=$month&amp;year=$year&amp;returl=$link_returl\" onClick=\"return confirm('".get_vocab("confirmdel")."');\">".get_vocab("deleteseries")."</a>";
  }
  echo "</div>\n";
 }

 // Copy and Copy Series
 echo "<div>\n";
 if (!$series)
 {
  echo "<a href=\"edit_entry.php?id=$id&amp;copy=1&amp;returl=$link_returl\">". get_vocab("copyentry") ."</a>";
 }
 if (!empty($repeat_id) && !$series && $repeats_allowed)
 {
  echo " - ";
 }
 if ((!empty($repeat_id) || $series) && $repeats_allowed)
 {
  echo "<a href=\"edit_entry.php?id=$id&amp;edit_type=series&amp;day=$day&amp;month=$month&amp;year=$year&amp;copy=1&amp;returl=$link_returl\">".get_vocab("copyseries")."</a>";
```

```
  }
  echo "</div>\n";

  // Export and Export Series
  if (!$keep_private && !$enable_periods)
  {
    // The iCalendar information has the full booking details in it, so we will not allow
    // it to be exported if it is private and the user is not authorised to see it.
    // iCalendar information doesn't work with periods at the moment (no periods to times mapping)
    echo "<div>\n";
    if (!$series)
    {
      echo "<a href=\"view_entry.php?action=export&amp;id=$id&amp;returl=$link_returl\">".
get_vocab("exportentry") ."</a>";
    }
    if (!empty($repeat_id)  && !$series)
    {
      echo " - ";
    }
    if (!empty($repeat_id) || $series)
    {
      echo "<a href=\"view_entry.php?
action=export&amp;id=$repeat_id&amp;series=1&amp;day=$day&amp;month=$month&amp;year=$year&amp;retur
l=$link_returl\">".get_vocab("exportseries")."</a>";
    }
    echo "</div>\n";
  }
  ?>
  <div id="returl">
    <?php
    if (isset($HTTP_REFERER)) //remove the link if displayed from an email
    {
    ?>
    <a href="<?php echo htmlspecialchars($HTTP_REFERER) ?>"><?php echo get_vocab("returnprev") ?></a>
    <?php
    }
    ?>
  </div>
</div>

<?php
//modified
?>
<script language="JavaScript">
var current_user = ["<?php echo $user; ?>"];
</script>
<?php
require_once "trailer.inc";
```

```
?>
```

Fitxer functions_view.inc (Només s'afegeixen les funcions modificades, amb el codi modificat sota el tag //modified):

```
//modified
require_once "mrbs_sql.inc";
function create_details_row($label, $value, $as_html=FALSE, $class='', $is_alumne, $repeat_id, $id, $day, $month,
$year, $area)
{
  global $number_alumnes_for_view;

  $result = '';
  if ($as_html)
  {
    $result .= "<tr>\n";
    $result .= "<td>$label:</td>\n";
    $result .= "<td" .
               ((!empty($class)) ? " class=\"$class\"" : "") .
               ">" . mrbs_nl2br(htmlspecialchars($value)) . "</td>\n";
    //modified
    if ($is_alumne && $repeat_id){
      $number_alumnes_for_view ++;
      //get the name of the user to delete (in other bookings of the serie mybe its in a diferent position)

      //first get the alumne field to delete
      if ($number_alumnes_for_view <= 9){
        $number_alumne = "Alumne_0".$number_alumnes_for_view;
      }
      else{
        $number_alumne = "Alumne_".$number_alumnes_for_view;
      }
      $sql = "SELECT $number_alumne FROM mrbs_entry WHERE id=\"$id\"";
      $alumne_to_delete = sql_query1($sql);
      $result .= "<td><a href='esborrar_usuari_reserva_multiple.php?
alumne=$alumne_to_delete&amp;id=$id&amp;day=$day&amp;month=$month&amp;year=$year&amp;area=$area'>
Borra l'usuari de la Reserva multiple</a></td>\n";
      $result .= "<td> &nbspClica aquest enllaç si vols eliminar l'alumne d'aquesta i les futures reserves (de la
classe i franja horaria seleccionada)</td>\n";
    }
    $result .= "</tr>\n";
  }
  else
  {
    // Some of the vocab strings contain  
    $result .= str_replace(' ', ' ', $label) . ": $value\n";
  }
```

```
  return $result;
}

// Returns a string containg a set of details for $data consisting of a label/value
// pair for each data element in the array $data.   If $as_html is TRUE then the string
// is the HTML for a table body, ie looks like "<tbody> ... </tbody>".
//    $keep_private   boolean   if TRUE then any private fields will be given the class 'private';
//                              note that $data must already have had values substituted
//                              for private fields
//    $room_disabled  boolean   if TRUE then a note will be added that the room is disabled
function create_details_body($data, $as_html=FALSE, $keep_private=FALSE, $room_disabled=FALSE, $repeat_id, $id,
$day, $month, $year, $area)
{
  global $enable_periods, $confirmation_enabled, $approval_enabled;
  global $is_private_field, $standard_fields;
  global $strftime_format;
  global $tbl_entry;
  global $select_options;

  // Get the duration if we haven't got it already
  if (!isset($data['duration']))
  {
    // We will translate the units later
    $d = get_duration($data['start_time'], $data['end_time'], FALSE);
    $data['duration'] = $d['duration'];
    $data['dur_units'] = $d['dur_units'];
  }

  // Set a rep_type if it hasn't been
  if (!isset($data['rep_type']))
  {
    $data['rep_type'] = REP_NONE;
  }

  // Go throuh each of the columns and for each of them that can be made private
  // substitute the private text if the user is not allowed to see the data
  $private_text = "[" . get_vocab("private") . "]";

  foreach ($data as $key => $value)
  {
    // We could just test each column against $is_private_field["entry.$key"]
    // but restricting the test to the columns below guards against the possibility
    // that somebody has accidentally configured a 'system' field to be private
    switch ($key)
    {
      case 'name':
      case 'description':
      case 'create_by':
```

```
      case 'room_name':
      case 'area_name':
      case 'type':
      case 'room_id':
      case 'entry_info_time':
      case 'entry_info_user':
      case 'entry_info_text':
      case 'repeat_info_time':
      case 'repeat_info_user':
      case 'repeat_info_text':
        $data[$key] = ($keep_private && $is_private_field["entry.$key"]) ? $private_text : $data[$key];
        break;

      default:
        if (!in_array($key, $standard_fields['entry']))
        {
          $data[$key] = ($keep_private && $is_private_field["entry.$key"]) ? $private_text : $data[$key];
        }
        break;
    }
  }


  $tbody = '';
  $tbody .= ($as_html) ? "<tbody>\n" : "";
  // Description
  $class = ($keep_private & $is_private_field['entry.description']) ? "private" : "";
  $tbody .= create_details_row(get_vocab("description"), $data['description'], $as_html, $class);
  // Confirmation status
  if ($confirmation_enabled)
  {
    $value = ($data['status'] & STATUS_TENTATIVE) ? get_vocab("tentative") : get_vocab("confirmed");
    $tbody .= create_details_row(get_vocab("confirmation_status"), $value, $as_html);
  }
  // Approval status
  if ($approval_enabled)
  {
    $value = ($data['status'] & STATUS_AWAITING_APPROVAL) ? get_vocab("awaiting_approval") :
get_vocab("approved");
    $tbody .= create_details_row(get_vocab("approval_status"), $value, $as_html);
  }
  // Room
  $value = $data['area_name'] . " - " . $data['room_name'];
  if ($room_disabled)
  {
    $value .= " (" . get_vocab("disabled") . ")";
  }
  $tbody .= create_details_row(get_vocab("room"), $value, $as_html);
```

```
// Start date
if ($enable_periods)
{
  list($start_period, $start_date) =  period_date_string($data['start_time']);
}
else
{
  $start_date = time_date_string($data['start_time']);
}
$tbody .= create_details_row(get_vocab("start_date"), $start_date, $as_html);
// Duration
$tbody .= create_details_row(get_vocab("duration"), $data['duration'] . " " . get_vocab($data['dur_units']), $as_html);
// End date
if ($enable_periods)
{
  list( , $end_date) =  period_date_string($data['end_time'], -1);
}
else
{
  $end_date = time_date_string($data['end_time']);
}
$tbody .= create_details_row(get_vocab("end_date"), $end_date, $as_html);
// Type
$type = get_type_vocab($data['type']);
$value = (empty($type)) ? "?${data['type']}?" : $type;
$tbody .= create_details_row(get_vocab("type"), $value, $as_html);
// Created by
$class = ($keep_private && $is_private_field['entry.create_by']) ? "private" : "";
$tbody .= create_details_row(get_vocab("createdby"), $data['create_by'], $as_html, $class);
// Last updated
$tbody .= create_details_row(get_vocab("lastupdate"), time_date_string($data['last_updated']), $as_html);
// The custom fields
$fields = sql_field_info($tbl_entry);
foreach ($fields as $field)
{
  $key = $field['name'];
  if (!in_array($key, $standard_fields['entry']))
  {
    $label = get_loc_field_name($tbl_entry, $key);
    // Output a yes/no if it's a boolean or integer <= 2 bytes (which we will
    // assume are intended to be booleans)
    if (($field['nature'] == 'boolean') ||
        (($field['nature'] == 'integer') && isset($field['length']) && ($field['length'] <= 2)) )
    {
      if ($keep_private && $is_private_field["entry.$key"])
      {
        $value = $data[$key];  // Will have been set previously
      }
```

```
      else
      {
       $value = empty($data[$key]) ? get_vocab("no") : get_vocab("yes");
      }
     }
     // Otherwise output a string
     else
     {
      if (isset($data[$key]))
      {
       // If the custom field is an associative array then we want
       // the value rather than the array key
       if (isset($select_options["entry.$key"]) &&
           is_assoc($select_options["entry.$key"]) &&
           array_key_exists($data[$key], $select_options["entry.$key"]))
       {
        $value = $select_options["entry.$key"][$data[$key]];
       }
       else
       {
        $value = $data[$key];
       }
      }
      else
      {
       $value = "";
      }
     }
     $class = ($keep_private && $is_private_field["entry.$key"]) ? "private" : "";
     //modified
     $tbody .= create_details_row($label, $value, $as_html, $class, $is_alumne = TRUE, $repeat_id, $id, $day, $month,
$year, $area);
    }
   }
   // Repeat type
   $tbody .= create_details_row(get_vocab("rep_type"), get_vocab("rep_type_" . $data['rep_type']), $as_html);
   // Repeat details
   if($data['rep_type'] != REP_NONE)
   {
    if (($data['rep_type'] == REP_WEEKLY) || ($data['rep_type'] == REP_N_WEEKLY))
    {
     if ($data['rep_type'] == REP_N_WEEKLY)
     {
      // Repeat number of weeks
      $tbody .= create_details_row(get_vocab("rep_num_weeks")." ".get_vocab("rep_for_nweekly"),
$data['rep_num_weeks'], $as_html);
     }
     // Repeat days
```

```
    $tbody .= create_details_row(get_vocab("rep_rep_day"), get_rep_day_list($data['rep_opt']), $as_html);
   }
   // Repeat end date
   $tbody .= create_details_row(get_vocab("rep_end_date"),
                               utf8_strftime($strftime_format['date'],
                               $data['end_date']),
                               $as_html);
  }
  $tbody .= ($as_html) ? "</tbody>\n" : "";

  return $tbody;
}
```

Fitxer functions_table.inc (Només s'afegeixen les funcions modificades, amb el codi modificat sota el tag //modified):

```
function cell_html($cell, $query_strings)
{
  // draws a single cell in the main table of the day and week views
  //
  // $cell is a two dimensional array that is part of the map of the whole
  // display and looks like this:
  //
  // $cell[n][id]
  //        [is_repeat]
  //        [is_multiday_start]
  //        [is_multiday_end]
  //        [color]
  //        [data]
  //        [long_descr]
  //        [create_by]
  //        [room_id]
  //        [start_time]
  //        [slots]
  //        [status]
  //
  // where n is the number of the booking in the cell.    There can be none, one or many
  // bookings in a cell.    If there are no bookings then a blank cell is drawn with a link
  // to the edit entry form.     If there is one booking, then the booking is shown in that
  // cell.    If there is more than one booking then all the bookings are shown, but they can
  // be shown in two different ways: minimised and maximised.   By default they are shown
  // minimised, so that the standard row height is preserved.   By clicking a control
  // the cell can be maximised.   (Multiple bookings can arise in a cell if the resolution
  // of an existing database in increased or the booking day is shifted).

  // $query_strings is an array containg the query strings (or partial query strings) to be
```

```
// appended to the link used for the cell.    It is indexed as follows:
//    ['new_periods']   the string to be used for an empty cell if using periods
//    ['new_times']     the string to be used for an empty cell if using times
//    ['booking']       the string to be used for a full cell

global $main_cell_height, $main_table_cell_border_width;
global $area, $year, $month, $timetohighlight;
global $enable_periods, $times_along_top, $show_plus_link;
global $approval_enabled, $confirmation_enabled;

$html = '';
$user = getUserName();

// if the time slot has got multiple bookings, then draw a mini-table
if(isset($cell[1]["id"]))
{
  // Find out how many bookings there are (needed to calculate heights)
  $n_bookings = 0;
  while (isset($cell[$n_bookings]["id"]))
  {
    $n_bookings++;
  }

  // Make the class maximized by default so that if you don't have JavaScript then
  // you can still see all the bookings.    If you have JavaScript it will overwrite
  // the class and make it minimized.
  $html .= "<td class=\"multiple_booking maximized\">\n";

  // First draw the mini table
  $html .= "<div class=\"celldiv slots1 mini\">\n";
  $html .= "<div class=\"multiple_control\">+</div>\n";
  $html .= "<table>\n";
  $html .= "<tbody>\n";

  $row_height = $main_cell_height - (($n_bookings-1) * $main_table_cell_border_width);   // subtract the borders (first
row has no top border)
  $row_height = $row_height/$n_bookings;  // split what's left between the bookings
  $row_height = (int) ceil($row_height);  // round up, so that (a) there's no empty space at the bottom
                                          // and (b) each stripe is at least 1 unit high
  for ($n=0; $n<$n_bookings; $n++)
  {
    $id         = $cell[$n]["id"];
    $is_repeat   = $cell[$n]["is_repeat"];
    $is_multiday_start = $cell[$n]["is_multiday_start"];
    $is_multiday_end   = $cell[$n]["is_multiday_end"];
    $status      = $cell[$n]["status"];
    $color       = $cell[$n]["color"];
    $descr       = htmlspecialchars($cell[$n]["data"]);
```

```
  $long_descr  = htmlspecialchars($cell[$n]["long_descr"]);
  $create_by  = htmlspecialchars($cell[$n]["create_by"]);
  $class = $color;
  if ($status & STATUS_PRIVATE)
  {
    $class .= " private";
  }
  if ($approval_enabled && ($status & STATUS_AWAITING_APPROVAL))
  {
    $class .= " awaiting_approval";
  }
  if ($confirmation_enabled && ($status & STATUS_TENTATIVE))
  {
    $class .= " tentative";
  }
  if ($is_multiday_start)
  {
    $class .= " multiday_start";
  }
  if ($is_multiday_end)
  {
    $class .= " multiday_end";
  }

  $html .= "<tr>\n";
  $html .= "<td class=\"$class\"" .
           (($n==0) ? " style=\"border-top-width: 0\"" : "") .   // no border for first row
           ">\n";
  $html .= "<div style=\"overflow: hidden; " .
           "height: " . $row_height . "px; " .
           "max-height: " . $row_height . "px; " .
           "min-height: " . $row_height . "px\">\n";
  $html .= " \n";
  $html .= "</div>\n";
  $html .= "</td>\n";
  $html .= "</tr>\n";
}
$html .= "</tbody>\n";
$html .= "</table>\n";
$html .= "</div>\n";

// Now draw the maxi table
$html .= "<div class=\"maxi\">\n";
$total_height = $n_bookings * $main_cell_height;
$total_height += ($n_bookings - 1) * $main_table_cell_border_width;  // (first row has no top border)
$html .= "<div class=\"multiple_control\" " .
         "style =\"height: " . $total_height . "px; " .
                  "min-height: " . $total_height . "px; " .
```

```
                "max-height: " . $total_height . "px; " .
            "\">-</div>\n";
    $html .= "<table>\n";
    $html .= "<tbody>\n";
    for ($n=0; $n<$n_bookings; $n++)
    {
      $id         = $cell[$n]["id"];
      $is_repeat   = $cell[$n]["is_repeat"];
      $is_multiday_start = $cell[$n]["is_multiday_start"];
      $is_multiday_end   = $cell[$n]["is_multiday_end"];
      $status      = $cell[$n]["status"];
      $color       = $cell[$n]["color"];
      $descr       = htmlspecialchars($cell[$n]["start_time"] . " " . $cell[$n]["data"]);
      $long_descr  = htmlspecialchars($cell[$n]["long_descr"]);
      $create_by   = htmlspecialchars($cell[$n]["create_by"]);
      $class = $color;
      if ($status & STATUS_PRIVATE)
      {
        $class .= " private";
      }
      if ($approval_enabled && ($status & STATUS_AWAITING_APPROVAL))
      {
        $class .= " awaiting_approval";
      }
      if ($confirmation_enabled && ($status & STATUS_TENTATIVE))
      {
        $class .= " tentative";
      }
      if ($is_multiday_start)
      {
        $class .= " multiday_start";
      }
      if ($is_multiday_end)
      {
        $class .= " multiday_end";
      }
      $html .= "<tr>\n";
      $html .= "<td class=\"$class\"" .
              (($n==0) ? " style=\"border-top-width: 0\"" : "") .   // no border for first row
              ">\n";
      $html .= "<div class=\"celldiv slots1\">\n";     // we want clipping of overflow
      $html .= "<a href=\"view_entry.php?id=$id&amp;". $query_strings['booking'] . "\" title=\"$long_descr\">";
      $html .= ($is_repeat) ? "<img class=\"repeat_symbol\" src=\"images/repeat.png\" alt=\"" . get_vocab("series") . "\"
title=\"" . get_vocab("series") . "\" width=\"10\" height=\"10\">" : '';
      //modified
      // only the area 7 that is the area of the "aula" family
      if ($area == 7) {
         $html .= "$descr<br>Professor: $create_by";
```

```
        $sql_1 = "SELECT  Alumne_01 ,  Alumne_02 ,  Alumne_03 ,  Alumne_04 ,  Alumne_05 ,  Alumne_06 ,  Alumne_07 ,
Alumne_08 ,  Alumne_09 ,  Alumne_10 ,  Alumne_11 ,  Alumne_12 ,  Alumne_13 , Alumne_14 ,  Alumne_15 ,  Alumne_16
,  Alumne_17 ,  Alumne_18 ,  Alumne_19 ,  Alumne_20 ,  Alumne_21 ,  Alumne_22 ,  Alumne_23 ,  Alumne_24 ,
Alumne_25 ,  Alumne_26 ,  Alumne_27 , Alumne_28 ,  Alumne_29 ,  Alumne_30 ,  Alumne_31 ,  Alumne_32 ,  Alumne_33
,  Alumne_34 ,  Alumne_35 ,  Alumne_36 ,  Alumne_37 ,  Alumne_38 ,  Alumne_39 ,  Alumne_40 FROM mrbs_entry
WHERE id='$id'";
        $res_1 = sql_query($sql_1);
        $row_1 = sql_row_keyed($res_1, 0);
        for ($j = 1; $j<=40; $j++){
             if ($j <= 9){
               $column_name = "Alumne_0".$j;
             }
             else{
               $column_name = "Alumne_".$j;
             }
             if ($row_1[$column_name]){
               $nom_alumne = $row_1[$column_name];
               $html .= "Alumne: $nome_alumne";
             }
        }
     $html .= "</a>\n";
     }
     else {$html .= "$descr<br><br>$create_by</a>\n";}
     $html .= "</div>\n";

     $html .= "</td>\n";
     $html .= "</tr>\n";
   }
   $html .= "</tbody>\n";
   $html .= "</table>\n";
   $html .= "</div>\n";

   $html .= "</td>\n";
 }  // end of if isset ( ...[1]..)

 // otherwise draw a cell, showing either the booking or a blank cell
 else
 {
   if(isset($cell[0]["id"]))
   {
     $id         = $cell[0]["id"];
     $is_repeat  = $cell[0]["is_repeat"];
     $is_multiday_start = $cell[0]["is_multiday_start"];
     $is_multiday_end   = $cell[0]["is_multiday_end"];
     $status     = $cell[0]["status"];
     $color      = $cell[0]["color"];
     $descr      = htmlspecialchars($cell[0]["data"]);
```

```
  $long_descr = htmlspecialchars($cell[0]["long_descr"]);
  $create_by   = htmlspecialchars($cell[0]["create_by"]);
  $slots      = $cell[0]["slots"];
}
else  // id not set
{
  unset($id);
  $slots = 1;
}

// $c is the colour of the cell that the browser sees. Zebra stripes normally,
// row_highlight if we're highlighting that line and the appropriate colour if
// it is booked (determined by the type).
// We tell if its booked by $id having something in it
if (isset($id))
{
  $c = $color;
  if ($status & STATUS_PRIVATE)
  {
    $c .= " private";
  }
  if ($approval_enabled && ($status & STATUS_AWAITING_APPROVAL))
  {
    $c .= " awaiting_approval";
  }
  if ($confirmation_enabled && ($status & STATUS_TENTATIVE))
  {
    $c .= " tentative";
  }
  if ($is_multiday_start)
  {
    $c .= " multiday_start";
  }
  if ($is_multiday_end)
  {
    $c .= " multiday_end";
  }
  // Add a class to bookings that this user is allowed to edit so that the
  // JavaScript can turn them into resizable bookings
  if (getWritable($cell[0]['create_by'], $user, $cell[0]['room_id']))
  {
    $c .= " writable";
    if ($is_repeat)
    {
      $c .= " series";
    }
  }
}
```

```
  else
  {
   $c = "new";
  }

  // Don't put in a <td> cell if the slot is booked and there's no description.
  // This would mean that it's the second or subsequent slot of a booking and so the
  // <td> for the first slot would have had a rowspan that extended the cell down for
  // the number of slots of the booking.

  if (!(isset($id) && ($descr == "")))
  {
   $html .= tdcell($c, $slots);

   // If the room isn't booked then allow it to be booked
   if (!isset($id))
   {
    $html .= "<div class=\"celldiv slots1\">\n";  // a bookable slot is only one unit high
    $html .= "<a href=\"edit_entry.php?" .
          (($enable_periods) ? $query_strings['new_periods'] : $query_strings['new_times']) .
          "\">\n";
    if ($show_plus_link)
    {
     $html .= "<img src=\"images/new.gif\" alt=\"New\" width=\"10\" height=\"10\">\n";
    }
    $html .= "</a>\n";
    $html .= "</div>\n";
   }
   else               // if it is booked then show the booking
   {
    $html .= "<div data-id=\"$id\" class=\"celldiv slots" .
          (($times_along_top) ? "1" : $slots) .
          "\">\n";
    $html .= "<a href=\"view_entry.php?id=$id&amp;". $query_strings['booking'] . "\" title=\"$long_descr\">";
    $html .= ($is_repeat) ? "<img class=\"repeat_symbol $c\" src=\"images/repeat.png\" alt=\"" .
get_vocab("series") . "\" title=\"" . get_vocab("series") . "\" width=\"10\" height=\"10\">" : '';
    //modified
    if ($area == 7) {
     $html .= "$descr<br>Professor: $create_by.  Alumnes: ";

      $sql_1 = "SELECT  Alumne_01 ,  Alumne_02 ,  Alumne_03 ,  Alumne_04 ,  Alumne_05 ,  Alumne_06 ,  Alumne_07 ,
Alumne_08 ,  Alumne_09 ,  Alumne_10 ,  Alumne_11 ,  Alumne_12 ,  Alumne_13 , Alumne_14 ,  Alumne_15 ,  Alumne_16
,  Alumne_17 ,  Alumne_18 ,  Alumne_19 ,  Alumne_20 ,  Alumne_21 ,  Alumne_22 ,  Alumne_23 ,  Alumne_24 ,
Alumne_25 ,  Alumne_26 ,  Alumne_27 , Alumne_28 ,  Alumne_29 ,  Alumne_30 ,  Alumne_31 ,  Alumne_32 ,  Alumne_33
,  Alumne_34 ,  Alumne_35 ,  Alumne_36 ,  Alumne_37 ,  Alumne_38 ,  Alumne_39 ,  Alumne_40 FROM mrbs_entry
WHERE id='$id'";
      $res_1 = sql_query($sql_1);
      $row_1 = sql_row_keyed($res_1, 0);
```

```
        for ($j = 1; $j<=40; $j++){
              if ($j <= 9){
                $column_name = "Alumne_0".$j;
              }
              else{
                $column_name = "Alumne_".$j;
              }
              if ($row_1[$column_name]){
                $nom_alumne = $row_1[$column_name];
                $html .= "$nom_alumne, ";
              }
         }
       $html .= "</a>\n";
       }
       else {$html .= "$descr<br><br>$create_by</a>\n";}
       $html .= "</div>\n";
     }
     $html .= "</td>\n";
   }
  }

  return $html;
}  // end function draw_cell
```

Fitxer afegir_usuari_reserva.php:

```
<?php

require_once "defaultincludes.inc";
require_once "mrbs_sql.inc";

// Check the user is authorised for this page
checkAuthorised();

//avoid duplicate entries in reload page
if(!empty($_SESSION['form_submitted'])){echo "Ja has enviat aquest formulari una vegada";$returl = "day.php";
       echo "<body style='background-color:#E7EBEE'><br><a href='$returl'>Tornar a Reserva de
Sales</a></body>";exit;}

// Get non-standard form variables
//
$id = get_form_var('id', 'int');
$day = get_form_var('day', 'string');
$month = get_form_var('month', 'string');
$year = get_form_var('year', 'string');
$alumne_a_afegir = get_form_var('usuari', 'string');
$area = get_form_var('area', 'string');
```

```
$capacity = get_form_var('capacity', 'int');

if (!empty($id) && !empty($day) && !empty($month) && !empty($year) && !empty($alumne_a_afegir) && !
empty($area) && !empty($capacity)) {
        //set this to avoid duplicate enter if reload page
    $_SESSION['form_submitted'] = 1;

    //set the capacity to -1 as the teacher makes the booking and is not in the alumnes_x field that we use to ckeck
capacity
        $capacity --;

        //set the return url
        $returl = "view_entry.php?id=$id&area=$area&day=$day&month=$month&year=$year";

        //get the timestamp of the booking
        $booking_timestamp = mktime(0,0,0,$month,$day,$year);
        //get the repeat id
        $sql_1 = "SELECT repeat_id
                FROM $tbl_entry
                WHERE id=\"$id\"";
        $repeat_id = sql_query1($sql_1);

        //get all id with same repeat id and with a start time equal or bigger than the start day of the booking at 00:00
        $sql_2 = "SELECT id
                FROM $tbl_entry
                WHERE repeat_id=\"$repeat_id\"
                AND start_time>=\"$booking_timestamp\"";
        $id_row_query = sql_query($sql_2);
        //check if any booking have reached the maxim capacity of the room. If so exit.
        for ($i = 0; ($id_row = sql_row($id_row_query, $i)); $i++)
        {
           $sql_3 = "SELECT COUNT(CASE WHEN Alumne_01 <> '' THEN Alumne_01 END) + COUNT(CASE WHEN
Alumne_02 <> '' THEN Alumne_02 END) + COUNT(CASE WHEN Alumne_03 <> '' THEN Alumne_03 END) +
COUNT(CASE WHEN Alumne_04 <> '' THEN Alumne_04 END) + COUNT(CASE WHEN Alumne_05 <> '' THEN Alumne_05
END) + COUNT(CASE WHEN Alumne_06 <> '' THEN Alumne_06 END) + COUNT(CASE WHEN Alumne_07 <> '' THEN
Alumne_07 END) + COUNT(CASE WHEN Alumne_08 <> '' THEN Alumne_08 END) + COUNT(CASE WHEN Alumne_09 <>
'' THEN Alumne_09 END) + COUNT(CASE WHEN Alumne_10 <> '' THEN Alumne_10 END) + COUNT(CASE WHEN
Alumne_11 <> '' THEN Alumne_11 END) + COUNT(CASE WHEN Alumne_12 <> '' THEN Alumne_12 END) +
COUNT(CASE WHEN Alumne_13 <> '' THEN Alumne_13 END) + COUNT(CASE WHEN Alumne_14 <> '' THEN Alumne_14
END) + COUNT(CASE WHEN Alumne_15 <> '' THEN Alumne_15 END) + COUNT(CASE WHEN Alumne_16 <> '' THEN
Alumne_16 END) + COUNT(CASE WHEN Alumne_17 <> '' THEN Alumne_17 END) + COUNT(CASE WHEN Alumne_18 <>
'' THEN Alumne_18 END) + COUNT(CASE WHEN Alumne_19 <> '' THEN Alumne_19 END) + COUNT(CASE WHEN
Alumne_20 <> '' THEN Alumne_20 END) + COUNT(CASE WHEN Alumne_21 <> '' THEN Alumne_21 END) +
COUNT(CASE WHEN Alumne_22 <> '' THEN Alumne_22 END) + COUNT(CASE WHEN Alumne_23 <> '' THEN Alumne_23
END) + COUNT(CASE WHEN Alumne_24 <> '' THEN Alumne_24 END) + COUNT(CASE WHEN Alumne_25 <> '' THEN
Alumne_25 END) + COUNT(CASE WHEN Alumne_26 <> '' THEN Alumne_26 END) + COUNT(CASE WHEN Alumne_27 <>
'' THEN Alumne_27 END) + COUNT(CASE WHEN Alumne_28 <> '' THEN Alumne_28 END) + COUNT(CASE WHEN
Alumne_29 <> '' THEN Alumne_29 END) + COUNT(CASE WHEN Alumne_30 <> '' THEN Alumne_30 END) +
```

```
COUNT(CASE WHEN Alumne_31 <> '' THEN Alumne_31 END) + COUNT(CASE WHEN Alumne_32 <> '' THEN Alumne_32
END) + COUNT(CASE WHEN Alumne_33 <> '' THEN Alumne_33 END) + COUNT(CASE WHEN Alumne_34 <> '' THEN
Alumne_34 END) + COUNT(CASE WHEN Alumne_35 <> '' THEN Alumne_35 END) + COUNT(CASE WHEN Alumne_36 <>
'' THEN Alumne_36 END) + COUNT(CASE WHEN Alumne_37 <> '' THEN Alumne_37 END) + COUNT(CASE WHEN
Alumne_38 <> '' THEN Alumne_38 END) + COUNT(CASE WHEN Alumne_39 <> '' THEN Alumne_39 END) +
COUNT(CASE WHEN Alumne_40 <> '' THEN Alumne_40 END) AS total_alumnes FROM $tbl_entry WHERE
id=\"$id_row[0]\"";
            $number_alumnes=sql_query1($sql_3);
            if ($number_alumnes >= $capacity){
              $sql_4 = "SELECT timestamp FROM $tbl_entry WHERE id=\"$id_row[0]\"";
              $data_reserva_plena=sql_query1($sql_4);
              echo "<body style='background-color:#E7EBEE'>LA RESERVA NO S'HA POGUT FER! <br>La reserva del
dia $data_reserva_plena ja te $number_alumnes alumnes i el professor. L'aula te una capacitat de $capacity alumnes
mes el professor. Si vols fer reserva hauras d'utilitzar una altra aula o ampliar la capacitat de l'aula<br>";
              echo "<a href='$returl'>Tornar a veure la Reserva</a></body>";
              exit;
            }
          }
          //insert the user in actual and future bookings of the same serie
          $sql_2_2 = "SELECT id
                      FROM $tbl_entry
                      WHERE repeat_id=\"$repeat_id\"
                      AND start_time>=\"$booking_timestamp\"";
          $id_row_query = sql_query($sql_2_2);
          for ($i = 0; ($id_row = sql_row($id_row_query, $i)); $i++)
          {
            //check the first empty position of the booking, from 1 to the room capacity
            for ($j = 1; $j<=$capacity ; $j++){
                  if ($j <= 9){
                    $alumne_j = "Alumne_0".$j;
                  }
                  else{
                    $alumne_j = "Alumne_".$j;
                  }
                  $sql_4= "SELECT COUNT(CASE WHEN $alumne_j <> '' THEN Alumne_01 END) FROM $tbl_entry
WHERE id=\"$id_row[0]\"";
                  $check_if_is_empty = sql_query1($sql_4);
                  //if the field is empty take the $alumne_j and exit
                  if ($check_if_is_empty == 0){
                    $j = $capacity;
                    $j ++;
                  }
            }
            $sql_5 ="UPDATE $tbl_entry
                     SET $alumne_j = \"$alumne_a_afegir\"
                     WHERE id=\"$id_row[0]\"";
            sql_query1($sql_5);
          }
```

```
        echo "<body style='background-color:#E7EBEE'>L'usuari s'ha insertat correctament. Ara ja apareixera a
aquesta reserva i les reserves futures de la mateixa classe i franja horaria<br>";
        echo "<a href='$returl' title='Si l enllac no funciona utilitza l enllac Reserva de Sales del requadre
superior'>Tornar a veure la Reserva</a></body>";
}
else{
    $returl = "day.php";
    echo "<body style='background-color:#E7EBEE'><a href='$returl'>Tornar a Reserva de Sales</a></body>";
}
?>
```

Fitxer esborrar_usuari_reserva.php:

```
<?php

require_once "defaultincludes.inc";
require_once "mrbs_sql.inc";

// Check the user is authorised for this page
checkAuthorised();

// Get non-standard form variables
//
$id = get_form_var('id', 'int');
$day = get_form_var('day', 'string');
$month = get_form_var('month', 'string');
$year = get_form_var('year', 'string');
$alumne_to_delete = get_form_var('alumne', 'string');
$area = get_form_var('area', 'string');

if (!empty($id) && !empty($day) && !empty($month) && !empty($year) && !empty($alumne_to_delete) && !
empty($area)) {

        //get the timestamp of the booking
        $booking_timestamp = mktime(0,0,0,$month,$day,$year);
        //get the repeat id
        $sql = "SELECT repeat_id
                FROM $tbl_entry
                WHERE id=\"$id\"";
        $repeat_id = sql_query1($sql);

        //get all id with same repeat id and with a start time equal or bigger than the start day of the booking at 00:00
        $sql_2 = "SELECT id
                FROM $tbl_entry
                WHERE repeat_id=\"$repeat_id\"
                AND start_time>=\"$booking_timestamp\"";
        $id_row_query = sql_query($sql_2);
```

```
        //delete all the entries with matched number_alumne and matched id
        for ($i = 0; ($id_row = sql_row($id_row_query, $i)); $i++)
        {
          //check the colum that contains the user to delete (40 alumnes)
          for ($j = 1; $j<=40; $j++){
           if ($j <= 9){
                $column_name = "Alumne_0".$j;
           }
           else{
                $column_name = "Alumne_".$j;
           }
           $sql_3_1 = "SELECT COUNT(CASE WHEN $column_name = '$alumne_to_delete' THEN $column_name END)
FROM $tbl_entry WHERE id=\"$id_row[0]\"";
           $resultat_query=sql_query1($sql_3_1);
           if ($resultat_query == 1){
                //match. We have the column_name then go out of for
                $match = TRUE;
                $j=41;
           }
          }
          if ($match) {
                $sql_3 = "UPDATE $tbl_entry
                        SET $column_name = ''
                        WHERE id=\"$id_row[0]\"";
                sql_query1($sql_3);
          }
        }
        $returnurl = "view_entry.php?id=$id&area=$area&day=$day&month=$month&year=$year";
        echo "<body style='background-color:#E7EBEE'>L'usuari s'ha donat de baixa correctament. Ara ja no
apareixera en aquesta reserva i a les reserves futures de la mateixa classe i franja horaria<br>";
        echo "<a href='$returnurl' title='Si l enllac no funciona utilitza l enllac Reserva de Sales del requadre
superior'>Tornar a veure la Reserva</a></body>";
}
else{
     $returnurl = "day.php";
     echo "<body style='background-color:#E7EBEE'><a href='$returnurl'>Tornar a Reserva de Sales</a></body>";
}
?>
```

Les modificacions als fitxers css i language han sigut mínimes pel que no es consideren en aquest annex

Annex 3: Mòdul moodle_actions de Drupal

Arxiu moodle_actions.info:

```
; $Id$
name = moodle_actions
description = Custom module to alter some db params for moodle external database compatibility
package = CUSTOM MODULES
dependencies[] = user
core = 7.x
files[] = moodle_actions.module
```

Arxiu moodle_actions.module:

```
<?php

/**
 * @file
 * This file is required for all modules
 */
/**
 * Implementation of hook_moodle_actions_action_info().
 */

function moodle_actions_action_info() {
  return array(
    'user_update_user_roles_courses_moodle_table' => array(
      'label' => t('Update user_roles_courses_moodle table'),
      'type' => 'user',
      'configurable' => FALSE,
      'triggers' => array('any'),
    ),
  );
}

//modificat
//custom functions for moodle
function user_update_user_roles_courses_moodle_table() {

   /*--add and delete diferent user_roles rid and uid to user_roles_courses_moodle--*/
    $array = array();
    $array_2 = array();

    $array_insert = array();
    $array_delete = array();

    $result = db_query('SELECT rid,uid FROM {users_roles}');
    $result_2 = db_query('SELECT rid,uid FROM {user_roles_courses_moodle}');
```

```
$array_length = 0;
$array_2_length = 0;
$array_insert_length = 0;
$array_delete_length = 0;

foreach ($result as $record) {
  $array['uid'] [] = $record->uid;
  $array['rid'] [] = $record->rid;
  $array_length ++;
}
foreach ($result_2 as $record_2) {
  $array_2['uid'] [] = $record_2->uid;
  $array_2['rid'] [] = $record_2->rid;
  $array_2_length ++;
}
//check first array to see the new entries in users_roles table
for ($i=0; $i<$array_length; $i++){

    $match = False;
    for ($j=0; $j<$array_2_length; $j++){

                    if ($array['uid'][$i]==$array_2['uid'][$j] && $array['rid'][$i]==$array_2['rid'][$j]){
            $match = True;
            $j=$array_2_length;
          }
    }
    if ($match == False){
      $array_insert ['uid'][] = $array['uid'][$i];
      $array_insert ['rid'][] = $array['rid'][$i];
      $array_insert_length ++;
    }
}

//Then check second array to see the entries in users_roles_courses_moodle that we have to delete
for ($i=0; $i<$array_2_length; $i++){

    $match = False;
    for ($j=0; $j<$array_length; $j++){

                    if ($array_2['uid'][$i]==$array['uid'][$j] && $array_2['rid'][$i]==$array['rid'][$j]){
            $match = True;
            $j=$array_length;
          }
    }
    if ($match == False){
      $array_delete ['uid'][] = $array_2['uid'][$i];
      $array_delete ['rid'][] = $array_2['rid'][$i];
      $array_delete_length ++;
```

```
        }
    }

    //insert new fields in table
    for ($i=0; $i<$array_insert_length; $i++){
        $dades_entrar = db_insert('user_roles_courses_moodle')
                        ->fields(array(
                        'rid' => $array_insert['rid'][$i],
                        'uid' => $array_insert['uid'][$i],
                                ))
                        ->execute();
    }

    //delete old fields of the table
    for ($i=0; $i<$array_delete_length; $i++){
        $dades_eliminar= db_delete('user_roles_courses_moodle')
                        ->condition('rid', $array_delete['rid'][$i])
                        ->condition('uid', $array_delete['uid'][$i])
                        ->execute();
    }


   /*---add the username to user_roles_courses_moodle table---*/
   global $user;
   $user_name = $user->name;
   $uid = $user->uid;
   db_update('user_roles_courses_moodle')
     ->condition('uid', $uid)
     ->fields(array('username' => $user_name))
     ->execute();

    /*---add moodle roles to user_roles_courses_moodle table ---*/
    if (in_array('professor_soci', $user->roles)) {
                db_update('user_roles_courses_moodle')
                  ->condition('uid', $uid)
                  ->fields(array('rolename' => 'manager'))
                  ->execute();
     }
     if (in_array('professor', $user->roles)) {
                db_update('user_roles_courses_moodle')
                  ->condition('uid', $uid)
                  ->fields(array('rolename' => 'editingteacher'))
                  ->execute();
     }
}
?>
```

El contingut de l'arxiu LICENSE.txt es pot trobar a

http://api.drupal.org/api/drupal/LICENSE.txt/7.

Annex4: GNU Free Documentation License

GNU Free Documentation License Version 1.3, 3 November 2008

0. Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. Applicability and definitions

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the

public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. Verbatim Copying

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. Copying in Quantity

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited

to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. Modifications

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of

Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. Combining Documents

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. Collections of documents

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. Aggregation with independent works

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. Translation

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. Termination

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some

reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. Future Revisions of this licensce

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. Relicensing

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had

no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

Addendum: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (C) YEAR YOUR NAME.
>
> Permission is granted to copy, distribute and/or modify this document
>
> under the terms of the GNU Free Documentation License, Version 1.3
>
> or any later version published by the Free Software Foundation;
>
> with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
>
> A copy of the license is included in the section entitled "GNU
>
> Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with ... Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the
>
> Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Bibliografia i Documentació per a la realització del projecte i redacció de la memòria

- [1] Logan, Kit and Neumann, Tim (2010). "Comparison of Blackboard 9.1 and Moodel 2.0". Institute of Education, University o London. Trobat a: http://www.lkl.ac.uk/LTU/files/publications/Comparison_Bb9-Mdl2_Aug2010.pdf

- [2] Momani, Alaa M (2010). "Comparison between Two Learning Management Systems:Moodle and Blackboard". Trobat a: http://papers.ssrn.com/sol3/Delivery.cfm/SSRN_ID1608311_code1450767.pdf?abstractid=1608311&mirid=1
- [3] Dougmias, Martin (2000). "Constructivisme Social". Trobat a: http://dougiamas.com/writing/constructivism.html, http://docs.Moodle.org/22/en/Philosophy

- [4] "Videotutorial instalación Moodle" Font: http://www.youtube.com/watch?v=_vx3dvfWLHA

- [5] "Manual instalación Drupal" Font: http://drupal.org.es/manuales/manual_del_administrador

- [6] "MRBS Install manual" Font: http://mrbs.sourceforge.net/view_text.php?section=Documentation&file=INSTALL

- [7] PAPAIOANNOU, JON & PREECE, JON (2009). "Desenvolupament de Blocks de Moodle". Trobat a : http://docs.Moodle.org/dev/Blocks

- [8] BUYTAERT, DRIES & SNIJDER, HANS & DRUPAL COMMUNITY (2001). "Desenvolupament i suport de Drupal". Trobat a http://Drupal.org/community & http://Drupal.org/documentation/develop

- [9] BERANEK, J & MRBS COMMUNITY (2008). "Mailing list del projecte MRBS". Trobat a MRBS-general@lists.sourceforge.net

- [10] D'ELIA BRANCO, MARCELO & LEÓN MARTÍNEZ, MÓNICA & NOVO LÓPEZ, ALEJANDRO & OTERO GARCÍA, ALBERTO (2006). "Implantació de sistemes de programari lliure". Trobat a http://ocw.uoc.edu/informatica-tecnologia-i-multimedia/implantacio-de-sistemes-de-programari-lliure/materials/

- [11] BÜCHNER, ALEX (2011). "Moodle 2 Administration". Llibre