PIM Place & Ghost Application

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per
Javier Ferreiro
i dirigit per
Joan Sorribes Gomis
Bellaterra,21 de Setembre de 2007

Universitat Autònoma de Barcelona

etse
Escola Tècnica Superior d'Enginyeria

El sotasignat,Joan Sorribes Gomis

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en Javier Ferreiro Espinosa

I per tal que consti firma la present.

Signat: .....

Bellaterra, 21 de Setembre de 2007

Historial de revisions

Historial de revisions
Revisió 1.0 20-07-2007 Revisat por: jfe
Document inicial.
Historial de revisions
Revisió 1.1 04-09-2007 Revisat por: jfe
Corregides faltes d'ortografia i reemplaçament d'imatges.
Historial de revisions
Revisió 1.2 21-09-2007 Revisat por: jfe
Afegit Agraïments.

Dedicatòria

Vull donar les gràcies al meu director de projectes Joan Sorribes i a tot el personal del SIEE amb els que he desenvolupat aquest projecte, per la paciència mostrada i l'ajuda prestada més enllà del que es podria demanar.

Per suposat a la meva família: als meus pares i el meu germà ja que són els veritables autors de tot allò que he aconseguit al llarg de la meva vida, inclòs la carrera.

Finalment a tots els meus amics, encara que siguin lluny, faci temps que no els vegi o em passi el dia amb ells donant la tabarra. Es inevitable portar alguna cosa de cadascun d'ells dins meu i per tant també participen d'aquest projecte.

Index de contingut

Capítol 0.Introducció

Objectius

Aquest projecte té dos objectius principals, per un costat aprofitar recursos infrautilitzats, (pc's) per donar un nou servei: màquines virtuals executant-se dins els recursos infrautilitzats i, per altre banda, crear un framework de treball per poder desenvolupar de manera integrada i reaprofitant tot el codi possible, aplicacions web, on la primera aplicació que es desenvoluparà serà per poder accedir a les màquines virtuals anteriorment comentades.

Anàlisis de l'entorn.

Per poder entendre quines han estat les motivacions d'aquest projecte lo millor que podem fer és fer una breu descripció de quin és l'entorn de treball amb el que ens enfrontem dins del SIEE (suport informàtic escola d'enginyeries) ja que el desenvolupament d'aquest projecte neix amb la voluntat de solucionar algun d'aquests problemes.

L'escola d'enginyeries de la Universitat Autònoma de Barcelona ha sofert un creixement bastant important des de la seva creació. S'han incorporat noves carreres com ara la de Gestió aeronàutica o algunes ja existents han passat a oferir la possibilitat de ser enginyeries superiors, com ara la de Telecomunicacions. Tot això ha fet que si be la ampliació d'alumnes ha estat notòria, les instal·lacions informàtiques que disposem de manera general per tots els alumnes no han augmentat, provocant una congestió a les aules d'ordinadors, sobretot a la època més propera a l'entrega de totes les pràctiques.

Per altre banda durant aquest període s'han popularitzat cada cop més l'ús de portàtils entre els alumnes de l'ETSE (escola tècnica superior d'enginyeries). Sense entrar en grans detalls, aquest creixement ha estat possible gracies a l'abaratiment d'aquest dispositius al mercat i a algunes iniciatives per part de la direcció de l'escola per tal d'oferir promocions especials de portàtils als alumnes, amb condicions avantatjoses tant en preu com en finançament.

Altre dels problemes que ha generat l'augment d'alumnes ha estat la creixent demanda de noves funcionalitats per gestionar els serveis oferts als mateixos. Això ha provocat la creació de diferents aplicacions per manegar cadascuna d'aquestes necessitats, com per exemple son: PSG per gestionar les places dels diferents grups de pràctiques i les seves entregues, el servei de ftp per poder accedir a una unitat d'emmagatzemament personal per cada alumne, servei de préstecs de programari,etc.

En poques paraules, aquesta es la casuística de l'entorn que ha motivat la creació d'aquest projecte tot i que com es pot entendre no són els únics desafiaments que existeixen.

Focalització sobre las solucions.

La solució d'aquests problemes s'ha plantejat des de dues vessants diferenciades amb un punt comú que les uneix.

Per una banda tenim la congestió de les aules d'ordinadors i l'augment d'alumnes amb portàtils. Combinant aquests dos factors, obtenim que la millor manera de continuar donant el mateix servei sense fer fortes inversions en maquinari i en noves aules es crear sales d'ordinadors virtuals.

Bàsicament la idea consisteix en crear ordinadors virtuals amb imatges

similars a les que es fan per les aules, més especialitzades per aquesta funció, i transmetre tot l'escriptori o aplicacions concretes via xarxa fins a l'equip portàtil de l'alumne. D'aquesta manera l'alumne disposa el maquinari però es pot beneficiar, de manera fàcil i còmode, del programari llicenciat per la universitat i que de vegades tot i ser lliure o amb llicencia per estudiant es de no trivial instal·lació.

La segona part del projecte tracta de crear tota la infraestructura necessària per tenir un framework de treball. En aquest ha de ser fàcil integrar, poc a poc, totes o quasi totes les aplicacions que s'han anat desenvolupant durant tot aquest temps i crear noves. La intenció es que aquest framework no només doni agilitat al desenvolupament, sinó que alhora doni altres avantatges com ara single sign on i un look and feel similar per totes les aplicacions. El primer programa que s'integri dins aquest framework serà una aplicació web per donar accés a les màquines virtuals que tot just s'han comentat en el paràgraf anterior, essent aquest el nexe d'unió entre les dues parts del projecte.

Capítol 1.Màquines Virtuals

Anàlisis.

Requeriments No Funcionals

El principal requeriment no funcional que presenta aquest projecte ve donat per la plataforma de maquinari on es vol que s'executi les màquines. Degut a polítiques de la universitat una serie de 21 ordinadors recentment comprats han estat destinats a les aules on s'imparteixen classes, on la seva principal funció serà la de servir de base per la reproducció de presentacions que els professors mostren als alumnes. Les màquines son unes Dell Optiplex 750 amb un processador Pentium Core 2 duo i 2 Gb de Ram.

Fer servir aquestes màquines tant potents, per fer només presentacions es desaprofitar aquest recurs i es per això que el projecte aprofita aquestes màquines per, sense treure-li la funcionalitat per la que s'han comprat, incloure les màquines virtuals que seran servides per Internet.

Requeriments Funcionals.

Volem crear una serie de màquines virtuals que s'executaran alhora dins el mateix maquinari per poder aprofitar màximament el recursos que es descriuen als requeriments no funcionals. Per això es crearan 3 màquines virtuals diferents. Aquest número es degut a que una d'elles substituirà el s.o. que harà tenen aquestes màquines. Serà l'encarregat de donar continuitat a les funcionalitats que ja donaven les màquines anteriorment. Les altres dos seràn les encarregades de donar els nous serveis via xarxa. No s'han volgut

crear més per no crear un overhead pel fet de gestionar moltes màquines virtuals, tanmateix com per poder donar més MB de ram a aquestes dues.

La primera màquina virtual contindrà un sistema operatiu Windows XP, amb el programari necessari per poder fer presentacions multimèdia a les aules. Haurà de reproduir vídeo, documents de Microsoft Office, OpenOffice i àudio. També haurà de ser capaç de llegir els pen-drives on normalment els docents porten les seves presentacions, així com connectar-se a Internet. Per poder interactuar amb aquesta màquina virtual es necessitarà estar físicament al costat de la mateixa ja que la seva interacció es farà via teclat, ratolí i pantalla/projector que el propi maquinari porta.

La segona màquina Virtual també contindrà un sistema operatiu Windows XP amb programari especialment pensat pels alumnes de la ETSE, com ara: Matlab, Mapple, etc. El S.O. haurà d'estar unit al domini sieesj, que és on actualment es troben totes les màquines de l'aula. Aquest domini permetrà fer login amb el mateix nom i password d'usuari que a les aules. Aquesta màquina no es mostrarà per pantalla sinó que la interacció amb la mateixa vindrà donada per connexions remotes des de Internet que es faran via RDP.

Finalment, la tercera màquina virtual contindrà un sistema operatiu GNU/LINUX, en concret Ubuntu 7.04. Aquest s.o. anirà equipat també amb programari especific pels alumnes de la ETSE, validarà contra ldap del domini sieesj en aquest cas la connexió al mateix també vindrà donat de manera remota via Internet mitjançant FreeNX.

Estudi de possibles solucions.

S'ha fet un estudi de quina o quines serien les millors eines de virtualització sobre les que implementar les nostres màquines virtuals. Per poder destriar inicialment de les nombroses solucions que hi han al mercat, es van tenir en compte dos factors, que van determinar el conjunt sobre el qual es farien proves una mica més acuradament. El dos factors varen ser: que el

producte disposes una versió de programari lliure o almenys gratuïta per tal d'abaratir el costos que tindrà el desplegament i que tingués un recolzament important dins el mercat o una gran comunitat a darrera, ja que d'aquesta manera en assegurem un bon suport.

Dintre dels candidats que complien aquestes dues demandes, trobem dues gran famílies: las que fan servir virtualització i les que fan servir paravirtualització. Les que fan servir virtualització el que fan es emular tot el maquinari que hi ha en un computador, de tal manera que el s.o. que s'està executant dins d'aquest tipus de virtualitzadors “veu” tot un maquinari per ell com si fos l'únic S.O de la màquina. Aquesta solució permet fer córrer un s.o. dins del virtualitzador sense que el s.o. convidat (guest) hagi de patir cap modificació.

D'altre banda trobem els productes que fan servir tècniques de paravirtualització. Aquesta tècnica es basa en crear una api de crides al sistema, que el s.o. convidat ha de fer servir en lloc de manipular directament el maquinari. Aquestes crides son ateses pel denominat hipervisor qui es l'encarregat de fer efectiva la crida contra el maquinari. Es per això que els s.o. que corren sobre sistemes d'aquest tipus han de presentar un kernel modificat per adaptar-los a la api del hipervisor.

Dins aquesta última família de virtualització ha aparegut unes noves característiques als processadors de la marca Intel i AMD que han fet que finalment els sistemes de paravirtualització si que puguin executar sistemes operatius sense modificar.

Introducció a l'arquitectura x86

Per poder explicar amb una mica de més detall cadascuna de les propostes, hem de veure, encara que sigui per sobre, com funciona l'arquitectura x86. El nostre anàlisis es basa sobretot en els modes de

funcionament d'aquesta arquitectura que és el que ens determinarà gran part de les possibilitats de virtualització dels nostres entorns.

Normalment trobem que en els processadors de tipus x86 suporten quatre modes de funcionament, també coneguts per anells de protecció. Cadascun d'aquest anells té un nivell de privilegis associats pel que fa al control del maquinari. El sistemes operatius més estesos i que són objecte directa de les nostres proves de virtualització, basats en GNU/Linux (2.6) i Windows XP, fins ara només feien servir dos d'aquest quatre possibles modes, els anomenats user mode (Ring 3) o kernel mode (Ring 0).

Quan un programa funciona en kernel mode, es troba situat dins l'anell 0, el més privilegiat de tots. Dins d'aquest anell/mode de funcionament el programari té accés a qualsevol part del maquinari, com per exemple: accés complet a qualsevol punt de la memòria RAM, ús complet del repertori d'instruccions de la CPU fins i tot a les instruccions privilegiades,etc. S'entén que quan un programa funciona dins d'aquest mode es plenament de confiança ja que qualsevol fallada del mateix pot comprometre (tant en estabilitat com en seguretat) tota la resta del sistema, degut als privilegis que té.

L'altre mode de funcionament, el user mode, s'executa al ring 3, deixant desocupat la resta de nivells: ring 1 i ring 2. Quan un programa s'executa com a user mode el seu accés al maquinari es troba molt restringit per tal d'evitar que aquests tipus de programes comprometin l'estabilitat o la seguretat de la resta. Per exemple, un programa funcionant en user mode no pot accedir a la memòria de manera directa, evitant-se d'aquesta manera que pugui accedir a les dades d'altres processos. Per poder fer ús de la memòria ha de fer una petició a través d'unes interfícies ben definides a algun programa que s'executi en el ring 0, gairebé sempre el kernel del sistema operatiu, i serà aquest qui li atorgarà una regió de memòria.

Recentment, tant Intel com AMD, han introduït unes extensions en les

seves cpu's que han modificat lleugerament l'esquema anteriorment presentat. Aquestes extensions reben el nom de Intel-VT o de AMD-V depenent del fabricant. La funció d'aquestes extensions es donar suport a la virtualització a nivell de maquinari. La seva utilitat es la de permetre'ns córrer s.o.'s sense que hagin estat modificats dins d'estructures de paravirtualització que en principi requereixen d'aquest tret.

Farem una breu introducció a com funciona aquestes extensions amb els processadors de Intel. Hem de tenir en compte que els de AMD tot i no ser compatibles directament amb les instruccions d'Intel, la manera de treballar es semblant. Per tenir sistemes no modificats funcionant amb paravirtualització s'han creat dos nous modes d'operació VMX root i VMX non-root. VMX root es el mode d'operació en el que s'executa el virtual machine monitor o hypervisor. VMX non-root és el mode d'operació en el que s'executa el sistema amfitrió. Quan passem de VMX root mode a VMX non-root es diu que fem una vm entry i viceversa, quan passem de VMX non-root a VMX root mode fen una VM èxit. Cadascun d'aquests mètodes d'operació té els 4 nivells d'execució que s'havien comentat abans amb una particularitat: algunes instruccions que són altament crítiques provoquen una sortida del mode vmx non-root per passar al mode vmx root on el monitor s'encarrega de processar aquesta sol·licitud de manera adequada i segura. Així, sense retocar el kernel del s.o. guest, aquest es pot pensar que s'executa a un ring 0 normal, però en realitat es troba executant-se en el ring 0 VMX non-root i qualsevol crida que faci a instruccions privilegiades o intents d'accedir a zones de memòria privilegiades, provocaran com s'ha dit, un salt al ring 0 mode VMX root on l'hypervisor o monitor s'encarregarà de resoldre la petició.

Virtualització completa

La virtualització completa es aquella que, com ja hem comentat, podem agafar un S.O. sense ser modificat en cap aspecte i ser capaç de fer-lo funcionar tal i com funcionaria si estigues corrent sobre una plataforma de

maquinari normal. Per aconseguir aquest objectiu el programari de virtualització, normalment el que fa es “emular” tot el maquinari que el s.o. guest esperaria trobar-se. En la figura 1 podem veure un sistema tradicional on podem distingir, d'una banda, el maquinari i de l'altre el sistema operatiu que es troba com a mediador entre aquest maquinari i el programari que corre tot just per sobre del s.o. Quan es realitza una crida, per exemple de lectura d'un sector del disc dur, l'aplicatiu fa una crida al subsistema de fitxers del s.o. Aquest resol la crida contra el maquinari i finalment passa les dades a l'aplicatiu. Per contra, en la figura 2 podem veure que es manté l'esquema anterior, però a més s'afegeix una nova aplicació que es la capa de virtualització (“Virtualization Layer”). Aquesta es l'encarregada de simular que el sistema operatiu convidat (“Guest Operating System”) cregui que té el control del maquinari. Qualsevol petició del guest, la capa de virtualització s'encarrega de traduir-la a crides al s.o. host i aquest les resol, com anteriorment s'ha comentat contra el maquinari.

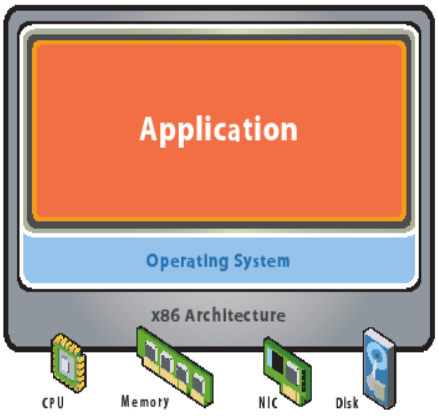


Figura 1.1.Esquema tradicional de maquinari i programari

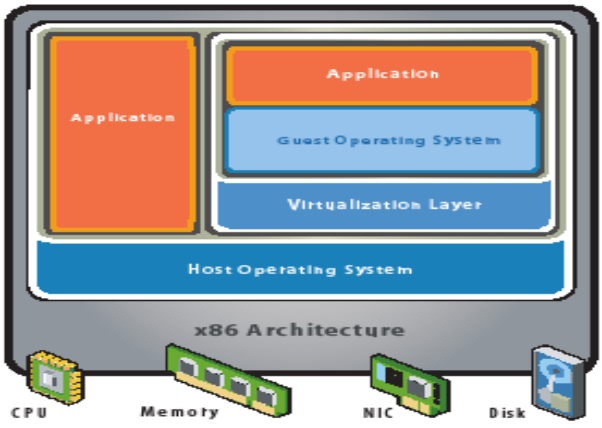


Figura 1.2.Esquema amb virtualització completa

Dins d'aquest mètodes analitzarem les solucions de Innotek i Vmware.

Innotek

Innotek comercialitza VirtualBox. Aquest producte virtualitza l'arquitectura x86 sobre plataformes Windows, Mac-Os i Linux de 32 bits i que permet tenir una amplia gama de S.O. convidats com ara:

- Windows: Vista, Xp, Server2003...
- Gnu/Linux: Ubuntu, Debian, Suse, Mandrake, Red Hat, fedora...
- Like *nix: OpenBSD, Solaris 10 (no es completament funcional).

El modus operandis de VirtualBox és el següent: Introdueix un driver dins el kernel del s.o. host per tal de que s'executi al ring 0. Aquest driver té bàsicament dues funcions: 1-)Reservar memòria dins el s.o. host. Un cop arrenca una màquina amb una determinada quantitat de ram, aquesta deixa d'estar disponible pel host i l'única manera de retornar aquesta ram és apagar el guest que ha fet la reserva. 2-)Salvar i restaurar l'estat del registres de la cpu i taules de descriptors quan el s.o. guest es interromput pel host. Fa algunes coses més però pel nostre objectiu ens quedarem amb aquestes dues, que són les més importants.

Un cop ja té aquest driver, quan executem un S.O. guest el que fa VirtualBox primer de tot és reservar la quantitat de memòria especificada a l'arxiu de configuració. Durant tota la vida d'execució del guest podem trobar-lo en algun d'aquests estats:

1-)S'està executant codi del ring 3 del guest de manera nativa, també anomenat “raw ring 3”. Això vol dir que els programes que està executant el s.o. guest no tenen cap requeriment especial més enllà que qualsevol altre programa i per això la cpu i el nostre sistema operatiu host l'executen com altre procés qualsevol. Aquí es on s'obté la màxima eficiència de la màquina virtual. Afortunadament és un mode de treball que no s'abandona gaire.

2-)Podem esta treballant en mode emulat dins del ring 3 del host. Això vol dir que quan no és segur executar el codi del guest com s'explica en el

primer estat, llavors entra la maquinaria d'emulació de VirtualBox, que evidentment, perjudica la performance d'execució. L'emulador esta basat en el projecte Qemu. Les situacions que poden disparar aquest estat són diverses, com: El codi del guest tracta de desactivar les interrupcions, tracta de executar alguna instrucció compromesa com per exemple LIDT (Li diu a la cpu on es troba de la memòria la taula d'interrupcions) i per últim quan tenim que processar el que s'anomena real-mode, es a dir accessos al maquinari directe, com per exemple, emulant la bios o fent l'arrancada d'un guest O.S.

3-)Finalment podem esta executant codi del ring 0 del guest de manera nativa. Aquesta es la part més delicada ja que el guest treballa com si estigues al ring 0, però en realitat es troba al ring 1 enganyat pel VirtualBox.Tot funciona bé fins que arriba a una instrucció privilegiada que, per suposat, no s'executarà de manera correcta al ring 1, només al ring 0. En aquest moment, el virtual machine monitor del virtualbox s'ha d'encarregar de capturar aquestes fallades i processar-les pel seu compte. Aquí si que penalitzem en velocitat ja que en el ring 0 del guest (que es troba al ring 1 del host) si que es fan moltes crides privilegiades i cadascuna d'elles s'han de emular.

Un cop vist com funciona internament passem a comentar quines són les diferents formes de distribució d'aquest programari. Existeixen dues versions: una lliure i una comercial. La versió lliure, anomenada per Innotek VirtualBox OSE (open source edition) és distribueix sota llicencia GPL i s'ha de baixar les fonts i compilar-les. La versió comercial, Innotek directament dona els binaris i els instal·ladors (tant per versions windows, com per versions Linux). A més a més les versions comercials tenen una serie de prestacions que no es troben incloses dins el paquet lliure com són: connexió a la màquina virtual via RDP, suport USB per derivar directament aquest dispositius a la màquina virtual, USB over RDP que és una combinació d'aquestes dues prestacions per poder connectar dispositius USB a una màquina virtual remota, Shared Folders per poder compartir directoris del host amb el guest i finalment iscsi initiator que permet iniciar màquines amb discs durs remots via iscsi tot i que el S.O. host no tingui instal·lat el sistema de iscsi.

Vmware

L'empresa Vmware, subsidiària de EMC Corporation, ens ofereix dos grans tipus de sol·lució: virtualització completa i una virtualització completa especial, sense s.o..

Dins la seva gama de virtualització, trobem 4 grans productes: vmplayer, vmware workstation, vmware gsx (també conegut com vmware server) i vmware esx que mirarem a part ja que és una mica més peculiar. A nivell teòric de virtualització, no trobem diferencies en cap dels tres productes, es a dir, totes emulen una màquina sencera dins un s.o. host que dona l'abstracció sobre el maquinari real. No podem fer un anàlisis tècnic molt més complet ja que, a diferencia de VirtualBox, el programari es propietari i no existeixen gaires detalls a nivell tècnic sobre el seu funcionament.

Les grans diferencies entre els tres productes es troben determinades pel rendiment superior conforme passem de vmware player a workstation i, finalment,a server. També es direrencien per altres prestacions com podrien ser la possibilitat de crear una nova màquina virtual (característica que a vmplayer no hi és), possibilitat de veure màquines virtuals que es troben a altres computadores (prestació exclusiva de vmware server),etc.

Vmware esx, és el que més prestacions té de tots els productes ja que fa una aproximació a la virtualització completa una mica diferent de la resta de productes vmware. En aquesta figura podem veure la diferència:

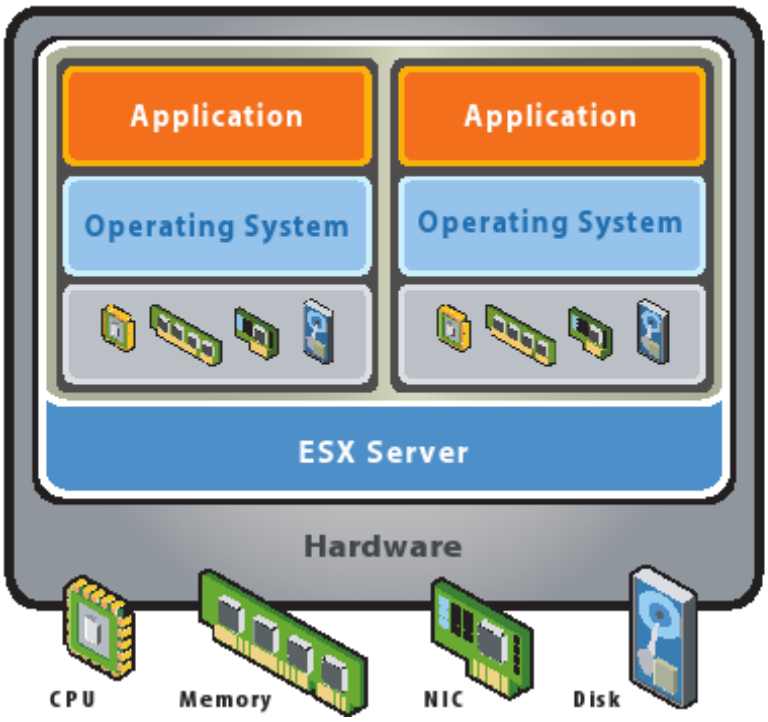


Figura 1.3. Esquema funcionament VMWare esx

Com és veu en aquest cas VMWare ESX no fa ús d'un S.O. complet sota el que es recolza, si no que ell mateix es qui gestiona directament tot el maquinari que té per sota. Llavors quan encenem la màquina tenim el vmkernel que manega directament tot el maquinari, en especial la cpu i la ram, ja que per manegar altres dispositius com ara targes de xarxa i dispositius d'emmagatzemament es recolza en moduls de linux pels quals ha creat un adaptador especial pel vmkernel.A més a a més integra una consola de management que es basa en un kernel 2.4 i en una distribució redhat desde on mitjançant cli (command line interface) podem manipular les diferents màquines virtuals.

En el tema de les llicencies, vmware ha anat variant la seva política. Quan es va començar aquest estudi, tots aquests productes eren totalment tancats, amb una llicencia propietària i només es permetia el seu ús a mode d'avaluació, on fins i tot en alguns cassos, necessitaves un número de serie que

donaven a la pròpia pàgina web, per poder activar l'avaluació. Amb el temps, aquesta política ha anat derivant cap a un ús més permissiu on, ara per ara ,es pot fer ús dels tres productes de forma gratuïta i fins i tot existeixen trossos de codi amb llicència GPL, tot i que l'ús continua estan limitat a proves i avaluacions del producte. A més a més tenen altres limitacions, com per exemple que no es pot copiar el programari a més d'una màquina, amb excepció si aquesta copia es per backup.

Paravirtualitzacio

La paravirtualització és un altre manera d'apropar-se al complex món de la virtualització. El secret d'aquesta tècnica radica en saltar-se el s.o. host i substituir-lo per un hypervisor que serà l'encarregat de multiplexar el maquinari per tots el s.o. guest que estiguin funcionant. Per poder fer això el s.o. guest ha d'estar modificat, ja que en lloc de manegar directament el maquinari mitjançant instruccions màquina privilegiades, el kernel ha de fer crides a l'api de l'hypervisor qui realment atacarà al maquinari. A la figura 1.4 podem veure aquesta idea:

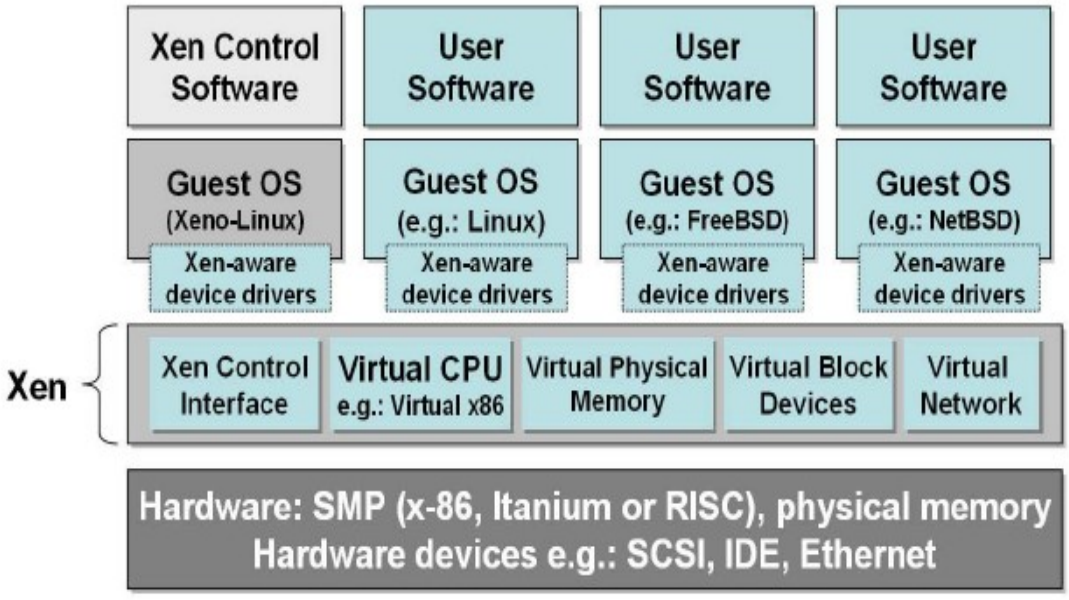


Figura 1.4. Esquema funcionament Xen

Com podem veure en aquesta figura els diferents sistemes operatius atacan directament a la capa de virtualització anomenada hypervisor. Dins d'aquests s.o. guest, existeix un privilegiat que es l'encarregat d'aixecar la resta i de controlar el seus paràmetres i per això té més privilegis, en el cas de la figura, és el que té a sota la Xen Control Interface.

Dins d'aquest món trobarem la solució de XenSource,Qumranet i UserModeLinux.

XenSource

Xen és un projecte de programari lliure que neix a la universitat de Cambridge i que amb el temps dona lloc a la formació de l'empresa XenSource que es l'encarregada de donar suport empresarial.

Xen es basa en la paravirtualització fent ús d'un hypervisor per controlar tot l'accés al maquinari. Podem distingir dos modes de funcionament: en mode de paravirtualització normal o en mode hvm. Per poder treballar en mode de paravirtualització normal necessitem que el S.O. guest estigui modificat per que el kernel guest en lloc d'intentar accedir al maquinari directament, faci crides a l'hypervisor xen qui serà en derrera instància qui resoldre la crida. En aquest mode, actualment xen 3.0 suporta: Linux 2.6, NetBSD 3.1, NetBSD 4.0_BETA2, FREEBSD amb alguns problemes i Solaris 10.

Si treballem en mode de virtualització hvm, el que estem fent es córrer el s.o. guest, amb ajuda de la cpu en mode VMX non-root (tal i com s'explica a la breu introducció que hi ha sobre l'arquitectura x86 del començament).

D'aquesta manera el kernel de l'operatiu guest creu que treballa al ring 0 tot i que en realitat treballa al ring 0 VMX non-root i que qualsevol intent d'execució d'una instrucció perillosa, provocarà un crida per part del processador del hypervisor qui s'encarregarà de resoldre aquesta situació. En quant als perifèrics i resta de maquinari necessari pel s.o. guest, s'encarrega d'emular-los una versió especial de qemu.

Les llicencies que podem trobar dins el producte de Xen són diverses. En principi tot el codi font base es distribueix sota llicencia GPL. Això ha propiciat que moltes distribucions de GNU/Linux incloguin dins la seva oferta de paquets, Xen. A més a més, XenSource ofereix tres productes més: Xen Express, Xen Server i Xen Enterprise. El primer d'ells es gratuït, permet 4 màquines virtuals, 4 GB de Ram 2 CPU físiques i permet virtualitzar tant linux com windows. No té cap mena de suport ni d'eina d'administració més avançada. El següent és Xen Server i és molt semblant a la versió express, les principals diferencies són que suporta fins a 8 GB de Ram, es troba enfocat a la virtualització de Windows només i que si que té suport. El seu cost és d'uns 99 dòlars americans per un any de suport. Finalment trobem Xen Enterprise que és la que més prestacions suporta: fins a 32 processadors, no té limit de Ram (tot i que Xen tècnicament no pot més enllà de 16GB de Ram), virtualitza tant windows com linux i la resta de kernels que anteriorment hem comentat, té suport i totes les eines d'administració avançades com el live migration que permet canviar una màquina virtual d'un host físic a un altre en qüestió de milisegons sense interrompre el servei de la màquina i mantenint la ip. El preu d'aquest últim començá als 1600 dolars americans.

User Model Linux

User Mode Linux és un programari fet per una comunitat de desenvolupadors que bàsicament consisteix en un kernel de Linux on mitjançant una serie de pegats podem fer córrer un altre kernel de Linux que s'hagi compilat per l'arquitectura UM (User Mode).

No s'ha trobat cap empresa que ofereixi support comercial per aquest sistema de virtualització, ni gaire informació técnica de com funciona internament. Per contra s'han trobat molta informació sobre com instal·larlo i crear màquines virtuals.

Qumranet

KVM és un producte de l'empresa Qumranet i que recentment ha estat inclòs dins del codi font del kernel de Linux com a mòdul. La idea de KVM es convertir, mitjançant aquest modul al que fèiem referència, el propi kernel de Linux en un hypervisor. D'aquesta manera un s.o. basat en linux i amb el modul de KVM podria fer alhora de s.o. normal i de hypervisor per altres s.o. guest. Mitjançant aquesta aproximació tenim un hypervisor que sense cap tipus d'esforç és madur, ja que a excepció de les lineas de kvm la resta són del propi kernel de Linux, ràpid, amb scheduling avançat ja que les màquines virtuals que corren dins són tractades com un procés normal de Linux i que té compatibilitat amb gran part del maquinari actual gràcies que els drivers que suporten aquest hypervisor són els propis del kernel de Linux. La idea la podem trobar reflexada a la següent figura:

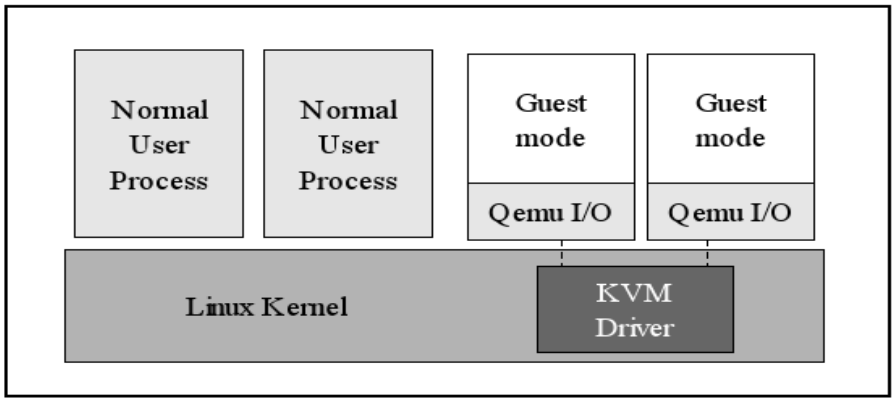


Figura 1.5.Esquema funcionament KVM.

Dins aquest model podem veure que per poder aconseguir dispositius d'entrada/sortida dins el s.o. guest hem de fer emulació del mateixos mitjançant la mediació d'una versió modificada de qemu.

Quan es va començar aquest estudi, tot just acabava de sortir dins del kernel oficial 2.20 i encara cap distribució l'havia empaquetat i qumranet encara no té cap producte oficial, es per aquest motiu que no es va arrivar a fer proves a fons com en la resta. Tot i això es tindrà en compte per aquest anàlisis i sobretot pel futur.

Disseny de la Màquina Virtual

Tecnologia aplicada

Un cop ja hem fet un estudi breu de totes les tecnologies que havíem seleccionat prèviament, passem al moment d'escollir quina és la que més ens convé per realitzar el nostre projecte. S'han fet proves amb gairebé totes les tecnologies estudiades a excepció de vmware esx on el que s'ha fet és consultar l'experiència d'un cas real on ja es troba implementat i en producció: Serveis D'informàtica centralitzats (SI) dins la UAB. Manegarem 5 paràmetres alhora de evaluar els productes : preu, suport, fiabilitat, facilitat d'instal·lació/manteniment i prestacions, en aquest ordre d'importància.

Per començar el primer punt que ens vam fixar per escollir la solució de virtualització és el preu. Tindrem un número relativament gran de màquines (21) i per tant comprar una solució per cadascuna d'elles pot suposar arribar a encarir el projecte de manera que fins i tot no es pugui realitzar. No es té una fita superior de diners, però el “life motive” d'aquest projecte és el de reaprofitar recursos de maquinari per oferir nous serveis.No té sentit

reaprofitar maquinari i després fer una gran despesa amb programari, per tant si es pot fer de manera que econòmicament sigui negligible seria l'ideal. En aquest apartat les eines de codi lliure, surten molt beneficiades: VirtualBox, Xen, UML i KVM. La gran perjudicada és VMWare que és la única que no té una versió realment lliure o gratuïta (només amb mode d'avaluació). Això que evita que podem implantar-lo a les màquines sense pagar la llicencia, que per altre banda, es bastant cara: el més barat, vmware workstation, 189 dòlars americans per llicencia.

El segon punt es el que tingués una companyia o una comunitat gran a darrera per estar segur de que tindríem recolzament per poder solucionar problemes a mesura que vagin sortint. En aquest apartat tenim que VMWare és la companyia que millor parada surt ja que és la que més temps porta al mercat i això, juntament amb l'èxit obtingut ha fet que tant la companyia, com la comunitat siguin molt grans. Seguidament trobaríem a Xen i VirtualBox que tot i no trobar-se tant desenvolupats en aquest sector tots dos tenen tant una comunitat, com una empresa que dona recolzament actiu a qualsevol problema que surtis. En el cas de necessitar recolzament per part de l'empresa, evidentment s'hauria de pagar les corresponents llicencies. En el cas de Xen, el creixement de la comunitat i l'empresa són molt notables i sembla que s'està convertint en un veritable competidor per VMWare. UML té una comunitat molt petita de desenvolupadors i de recolzament. Navegant per la web es troben enllaços trencats i tot dona una sensació de bastant abandonament. Finalment KVM es massa d'hora per tenir un criteri en aquest sentit, tot just acaba de sortir i a aquestes alçades no podem saber si la comunitat que es formarà serà gran o petita ja que acaba de néixer pràcticament com a solució estable.

El tercer punt que volíem considerar, era que el rendiment de la màquina virtual estigues lo més pròxim possible al rendiment,de la mateixa configuració, però sobre la màquina real. La intenció real d'aquestes proves no és tant aconseguir les millors xifres de rendiment de cadascuna de les solucions virtualitzadores, sinó comprovar que l'usuari no percep l'ús d'una màquina virtual. Aquesta decissió es basa en que el perfil d'usuari que donarà

feina a la màquina virtual no fa un treball intensiu en cpu, ni en I/O, ni en cap altre camp que podem destacar. Les possibles que podrien fer ús de més recurssos no faran ús intensiu dels mateixos ja que les pràctiques dels alumnes que són els usuaris que tindràn, a nivell general són fàcils de resoldre computacionalment parlant. Es per aquest motiu que no es va fer cap benchmark, tant sols s'executaven algunes aplicacions típiques com processadors de textos (openoffice i microsoft office), programes matemàtics típics de pràctiques (maple, matlab i spss) i es comprobava el seu correcte i fluid funcionament.

S'hi tenim això en compte, les solucions que fan servir la paravirtualització són les que més rendiment tenen, com a mínim a nivell teòric, ja que la seva solució ha d'executar menys codi degut a que no fa l'emulació de tot el sistema. Es per això que contra càrregas de treball més dures, Xen, UML i KVM haurien de ser més ràpides, almenys a nivell teòric. En qualsevol cas, en les proves que s'han fet, en totes les solucions el resultat va ser satisfactori sense que notéssim molta diferencia entre executar en màquina real o virtual, amb petites excepcions quan entrava en joc l'apartat gràfic, on, en general, totes les màquines virtuals es mostren incapaces d'oferir rendiments a l'alçada de la màquina real. Aquest punt és important ja que cap solució es capaç de donar-nos màquines virtuals amb acceleració 3D i fins i tot depenent quina resolució volem posar en 2D ens veurem limitats tant en disponibilitat de la mateixa com en velocitat.

Seguidament comentarem la fiabilitat. VMWare és la que millor s'ha comportat en totes les proves i les seves fallades han estat mínimes. A prop trobem VirtualBox que tampoc ha donat cap problema durant les proves fetes. Seguidament trobaríem a Xen que paravirtualitzant Linux es presenta molt estable, però quan ha de fer virtualització amb hvm, es a dir, virtualitzant un Windows XP, per exemple, ha deixat veura una mica més d'inestabilitat. Finalment tenim a KVM i UML en aquest ordre. KVM ha mostrat bastant inestabilitat, però que no podem ni tant sols culpar a KVM ja que el kernel on es va provar era una “release candidate“ d'ubuntu i no podríem assegurar que no

tingues problemes el mateix kernel. Com a últim classificat trobem a UML que realment donava molts problemes, penjades continuades i corrupcions del fitxer que contenia el disc dur virtual.

Finalment intentarem veure l'últim apartat, les facilitats que mencionavem per l'administrador: d'instal·lació, configuració i manteniment. La classificació es torna a repetir. VMWare torna a guanyar gràcies a una consola d'administració gràfica que fins i tot es pot connectar a ordinadors remots (VMWare Server). Seguit de prop per VirtualBox que també té aquesta facilitat tot i que no pot fer-ho de manera remota.Després trobariem Xen que té una mica menys madur el tema tot i que per cli (command line interface) es pot fer de tot, les seves solucions de gui són bastant més primitives. En quant a KVM i UML sobretot en el cas de l'últim resulta tremendament complex de manegar i sobretot d'instal·lar ja que es tracta d'aplicar un pegat al kernel (no existeix versió precompilada) que en el nostre cas el propi pegat tenia un error i per tant, vam haver de possar un pegat al propi pegat resultant tot en un grau de complexitat molt gran.

Per facilitar la comprensió i resumir les experiencies comentades amb els diferents productes reflexarem tots aquests punts en una taula, donant una puntuació del 0 al 10 a cadascun dels apartats (10 millor).

	PREU	SUPPORT	PRESTACIONS	ESTABILITAT	FACILITAT	TOTAL
VirtualBox	8	7	7	7	8	37
VMWare	4	9	7,5	8	8,5	37
Xen	8	8	8	7	7	38
UML	10	2	4	3	2	21
KVM	9	3	5	4,5	4	25,5

En el resultat, més que els números podem fer dos grans anàlisis: VirtualBox,VMWare i Xen són les tres solucions més vàlides amb molta diferència amb UML i KVM. L'altre gran conclusió que es pot treure es que les tres solucions virtualitzadores capdavanteres estan molt igualades i que potser

el tema econòmic és el que més ha pesat en la decisió En tot cas l'ha solució escollida ha estat la guanyadora per punts Xen.

Descripció de la Solució

Un cop amb la decisió pressa sobre fer servir Xen, passem a descriure com s'ha concretat la solució.

1-)Instal·lació de Xen.

Després de fer moltes proves amb diverses distribucions (OpenSuse i Ubuntu) i diferents maneres d'instal·lació de Xen (paquets precompilats de diferents origens i compilació de les fonts) es va determinar que la millor opció seria instal·lar la distribució Ubuntu ja que així es mantenia la homogeneïtat amb tots els ordinadors de les aules i això simplifica el tema d'administració. El fet de quedar-nos amb Ubuntu ens obliga a no fer servir el Xen que porta empaquetat de serie, tot i que seria lo desitjable per simplicitat. Això es degut a que tant en la versió Dapper, com en la actual Feisty, Ubuntu té greus problemes amb la virtualització. En Dapper no es podia virtualitzar més de 2 màquines, a més de la del Dom0, es a dir, la que té el control per llençar la resta de màquines virtuals, sense desactivar l'hyperthreading, i amb aquest desactivat tan sols 3. Si es posaven més màquines s'incorria en una “race condition” dins el kernel que automàticament resetejava l'ordinador. En Feisty, els problemes han vingut pel tema de hvm ja que quant s'intentava virtualitzar Windows XP i mostrar-lo per pantalla es quedava la màquina virtual penjada.

Per aquestes raons, es va optar per fer servir els paquets de xen que s'han posat a disposició del públic a: http://cedric.gabriello.fr/ubuntu. Amb aquest paquets podem tenir a disposició Xen 3.1 compilat per Ubuntu Feisty i sense cap dels problemes que dona els oficials d'Ubuntu.

2-)Descripció de les Màquines virtuals.

Com ja s'ha comentat, cada màquina real dispondrà de 3 màquines virtuals: VirtualMultimedia (vm), VirtualWindows (vw) i VirtualLinux (vl).

2.1-)VirtualMultimedia (vm)

La màquina vm serà l'encarregada de servir de suport pel professor quan faci classe. Bàsicament esta enfocada a fer presentacions amb powerpoints, pdf's i a reproduir algun vídeo. El sistema operatiu encarregat de fer aquestes tasques serà Windows XP on s'instalaran tots els programes necessaris per fer les tasques descrites. El disc dur d'aquesta màquina serà un fitxer que es trobarà a la arrel del Dom0. Això tot i que a priori pot fer una mica més lent l'accés a disc ens facilita la recuperació d'un desastre, ja que al mateix directori del fitxer es troba un altre anomenat vm_backup, que en cas de que el primer quedes en un estat no funcional, podriem copiar vm_backup sobre vm tenint d'aquesta manera la imatge de virtualmultimedia altre cop totalment funcional.

Altre part important es que aquesta màquina s'ha de veure directament per la sortida de VGA, que en condicions normals anirà a parar a un projector. Per poder fer això, en un primer moment es va intentar fent ús del propi servidor de vnc de xen, però presentava problemes, concretament que presentava dos punters del ratolí i es desincronitzaven entre ells.Per solventar això es va instal·lar un servidor propi al windows xp que es connectava mitjançant un script i un password en un fitxer. Les proves indicaven que no anava malament, fins que vam provar a reproduir un vídeo o un flash. Amb vnc no es tenia un refresc suficientment bo com per poder veure amb fluïdesa les imatges. Això ens va obligar a donar la sortida fent ús de les llibreries SDL. L'ús d'aquestes llibreries va augmentar les prestacions de vídeo notablement i ara per ara ja es pot veure vídeos (per exemple s'ha provat amb un mp4), vídeos de youtube (flash) i tot a una resolució de 1024x768 amb una profunditat de 24 bits. El principal problema que presentava SDL i que va ser el motiu pel qual no es va fer ús en primera instància, es que el ratolí no funcionava, ja que no hi havia sincronització entre el mouse de l'aplicació i el mouse de les X, degut a que no carregaven un manager de finestres. Això es

va solucionar posant la següent linea a l'arxiu de configuració de la màquina virtual: *usbdevice= 'tablet'*. En l'script següent podem veura l'arxiu de configuració de Xen per arrencar aquesta màquina:

```
disk= ['file:/srv/xen/images/vm.img,ioemu:hda,w']
memory = 256
vcpus = 1
builder = 'hvm'
device_model = '/usr/lib/xen-ioemu-3.1/bin/qemu-dm'
kernel = '/usr/lib/xen-ioemu-3.1/boot/hvmloader'
name = 'vm'
vif = [ 'type=ioemu,mac=00:16:3e:6a:26:72' ]
stdvga = 0
sdl = 1
audio=1
soundhw='all'
usbdevice= 'tablet'
usb = 1
localtime = 0
on_poweroff = 'restart'
on_reboot = 'restart'
on_crash = 'restart'
boot = 'c'
```

Fitxer configuració VM

Un cop ja s'ha definit la propia màquina virtual, cal crear tot allò que l'envolta i que permet, desde que sigui llençada de manera automàtica fins a conectar-li dispositius USB.

Per poder llençar automàticament vm al iniciar l'ordinador vam crear dos scripts. El primer script s'encarrega de llençar les X que executan el segon script, que és l'encarregat d'arrencar la vm dins l'entorn de les X. No es va voler fer servir cap gestor de finestres per que no es feia cap ús d'ell (només es llençava una finestra) i així s'estalviava memòria RAM.

El primer script, denominat Vx, tot just començar destruix qualsevol instància que hagués quedat de execucions prèvies que haguessin quedat penjades, sense control ni resposta. Seguidament llença “contra les X” l'script vm. Això es repeteix de manera indefinida. D'aquesta manera tot i que es

vulgui sortir de les X's (per exemple fent ús de la combinació ctrl+alt+backspace) es tornarà a llençar de manera immediata la màquina virtual. Aquesta combinació de teclas, també es pot aprofitar com a sistema per reiniciar el windows virtual.

```
#!/bin/bash
while true
do
        xm destroy vm
        xinit /usr/local/bin/vm
        xm destroy vm
done
```

Script Vx

El segon script és que el s'encarrega de llençar la màquina virtual dins les X i fer un bucle “infinit”. El motiu d'aquest segon bucle és degut a que quan es llença la màquina virtual, la finestra on surt es asíncrona, llavors l'execució de l'script continua sense esperar la finalització de la màquina virtual i no tenim temps de treballar amb ella que ja tornaria a reiniciar-se degut al primer script.

```
#!/bin/bash
#export SDL_VIDEO_X11_DGAMOUSE=0
/usr/sbin/xm create -c /etc/xen/vm
while true
do
sleep 1000d
done
```

Script vm.

L'altre problema que es va haver de sol·lucionar que afectava directament a vm és l'ús dels coneguts dispositius pen-drives. Xen no te una manera fàcil i transparent de passar un dispositiu com l'USB a una màquina virtual. Es per això que quan connectem un pen-drive a la màquina, l'únic s.o. que detecta el nou dispositiu és el Linux del Dom0. La majoria dels professors porten les seves presentacions en dispositius d'aquest tipus i això ens obligava a buscar alguna solució. Aquesta va venir donada de la següent manera: primer es van afegir alguns arxius a la màquina Dom0 per modificar el

comportament del programa UDEV i d'aquesta manera poder controlar el que fa el sistema quan es punxa algun dispositiu d'emmagatzemament a l'usb. Els arxius afegits són:

```
KERNEL=="sdb1", ACTION=="add", ENV{dispo}="%k", RUN+="/usr/local/bin/add", OPTIONS+="last_rule"
KERNEL=="sdb1", ACTION=="remove", ENV{dispo}="%k", RUN+="/usr/local/bin/remove", OPTIONS+="last_rule"
KERNEL=="sdc1", ACTION=="add", ENV{dispo}="%k", RUN+="/usr/local/bin/add", OPTIONS+="last_rule"
KERNEL=="sdc1", ACTION=="remove", ENV{dispo}="%k", RUN+="/usr/local/bin/remove", OPTIONS+="last_rule"
```

Fitxer 03-usb-storage.rules

El que provoquen aquestes modificacions a UDEV es que quan identifica que s'ha punxat algun dispositiu físic i el kernel l'ha otorgat el dispositiu lògic /dev/sdb, /dev/sdb1, /dev/sdc o /dev/sdc1 crida al corresponent script. En el cas de /dev/sdb1 o /dev/sdc1 l'script que entra en joc és el següent:

```
#!/bin/bash
case "$dispo" in
"sdb1")
        umount /media/pen1 2>> /tmp/hora
        mount -o sync,uid=65534,gid=65534 /dev/sdb1 /media/pen1 2>> /tmp/dispodebug
        ;;
        "sdc1")
        umount /media/pen2 2>> /tmp/hora
        mount -o sync,uid=65534,gid=65534 /dev/sdc1 /media/pen2 2>> /tmp/dispodebug
        ;;
        esac
```

Fitxer /usr/local/bin/add

El que farà l'script és muntar el dispositiu dins d'una carpeta prefixada /media/pen1 o /media/pen2. És important ressaltar la forma en que es monta el dispositiu: en primer lloc és fan amb un usuari i grup concret (nobody) perquè d'aquesta manera l'usuari que fa servir el vm pugui escriure. L'altre cosa important a ressaltar és l'ús d'un muntatge sync, es a dir, sincronitzat. Això és fa per evitar una possible pèrdua de dades fent ús de l'usb. Si la persona que fa ús dels dispositius USB grava alguna dada, aquesta, tal i com es rep es grava, mode sincronitzat, en contra del que passaria si fos en mode asíncron que és el S.O. que emmagatzema l'informació i en els moments que no fa altres feines

va gravant poc a poc l'informació. Si es treu el pen-drive en el moment en que es trova entre gravació i gravació perdrem les dades i podríem deixar corrupte tot el sistema d'arxius del pen-drive.

Quan el que es detecta des de UDEV és l'extracció dels dispositius /dev/sdb, /dev/sdb1, /dev/sdc o /dev/sdc1 el que es crida es un altre script. En el cas de /dev/sdb1 s'executaria el següent script:

```
#!/bin/bash
case "$dispo" in
        "sdb1")
        /etc/init.d/samba stop
        sleep 2
        killall -9 smbd
        sleep 2
        umount /media/pen1 2>> /tmp/dispodebug
        /etc/init.d/samba start 2>> /tmp/dispodebug
        ;;
        "sdc1")
        /etc/init.d/samba stop
        killall -9 smbd
        sleep 2
        umount /media/pen2 2>> /tmp/dispodebug
        /etc/init.d/samba start
        ;;
```

Script sub

L'script s'assegura que no hi hagi cap programa fent servir el dispositiu, per poder desmuntar-lo correctament. En aquest moment, normalment el servei que estarà fent servir el dispositiu es SAMBA i per això es para abans de desmuntar-lo, per després tornar-lo a activar.

Per altre banda, com ja hem vist, el directori /media/pen1 i /media/pen2 es comparteix amb vm, amb SAMBA. La configuració d'aquest servei correspon a la següent configuració:

```
[pendrive2]
hosts allow=158,109,203,26/24
```

```
hosts deny = 0.0.0.0/0
comment = Pen Drive 2
writable = yes
public = yes
path = /media/pen2
guest ok = yes
```

Part de l'arxiu de configuració de SAMBA

Aquí podem veure que la carpeta /media/pen2 només es comparteix amb la ip 158.109.203.26 que és la que té vm i amb la resta es denega l'accés. Això es fa per preservar la privacitat dels continguts del pen-drive i que només sigui accesible a vm.

S'ha d'aclarir que hem vist exemples de com es comporta si, per exemple, es punxes un pen-drive i aquest fos assignat a /dev/sdb1 o /dev/sdc1. No s'ha commentat el cas de que s'assignes a /dev/sdb o /dev/sdc, pel simple motiu que el comportament és anàlog i els scripts iguals, amb excepció, es clar, dels nom. En la mateixa casuistica ens trobem alhora de l'extracció del pen-drive.

2.2-)VirtualWindows

Aquesta màquina està orientada a ser accedida via xarxa. Per poder donar aquest servei en cadascun d'aquests windows, s'ha habilitat la opció de connexió remota, que ens permet fer ús del protocol RDP.

En un primer moment, es va intentar fer una copia exacta dels continguts d'una imatge del windows d'un ordinador d'un aula. Aquesta imatge, el primer cop que arrencava feia tot un seguit de canvis a la configuració del maquinari ja que la instal·lació original es sobre una màquina APD (Pentium IV,512 MB RAM).Windows XP detectava aquest canvi e instal·lava el nous drivers, pel nou maquinari. Quan tornàvem a iniciar la màquina, per causes que encara es desconeixen, Windows XP funcionava extremadament lent i tot i no estar desenvolupant cap tasca tal i com reflexava l'administrador de tasques, el tant

per cent de CPU que feia servir la màquina, en cap cas baixava del 12% i amb qualsevol operació per simple que fos, feia pujar l'ús de CPU fins al 100% durant bastant de temps. Finalment es va optar per fer una imatge completament nova i posar des de cero tot el programari que es desitjés.

La configuració de la màquina virtual vw és la següent:

```
disk= ['phy:/dev/sda3,ioemu:hda,w']
memory = 512
vcpus = 1
builder = 'hvm'
device_model = '/usr/lib/xen-ioemu-3.1/bin/qemu-dm'
kernel = '/usr/lib/xen-ioemu-3.1/boot/hvmloader'
name = 'vw'
vif = [ 'type=ioemu,mac=00:16:3e:6a:26:70' ]
sdl = 0
audio=0
boot = 'c'
usb = 0
localtime = 1
on_poweroff = 'restart'
on_reboot = 'restart'
on_crash = 'restart'
```

2.3-)VirtualLinux

Aquesta imatge va ser la que menys feina va suposar, ja que es va seguir també el procediment de fer una copia exacte de la partició d'un ordinador de les aules, fins la partició d'on arrencaria aquesta màquina virtual. En aquest cas, afortunadament, no hi va haver cap problema i tant sols fent els ajustaments mínims, la màquina va arrencar a la perfecció.

La configuració de la màquina virtual vl és la següent:

```
disk= ['phy:/dev/sda5,sda1,w','phy:/dev/sda6,sda2,w']
memory = 512
vcpus = 1
builder = 'linux'
kernel = '/boot/vmlinuz-2.6.19-4-generic'
ramdisk = '/boot/initrd.img-2.6.19-4-generic'
root = '/dev/sda1 ro'
```

```
name = 'vl'
boot = 'c'
vif = [ 'type=ioemu,mac=00:16:3e:6a:26:71' ]
localtime = 1
on_poweroff = 'restart'
on_reboot = 'restart'
on_crash = 'restart'
```

Fitxer Configuració VL

3-)Integració dins el sistema de desplegament d'imatges

L'últim apartat a resoldre era el de passar d'un ordinador model, a una imatge del mateix que s'autoconfigures per tots els ordinadors on es posaria. Tot i que el maquinari és el mateix i això permet fer la imatge, ens havíem d'enfrontar bàsicament a 2 grans problemes. Per un costat les interfícies de xarxa virtuals s'havien configurat amb una MAC estàtica, es a dir, s'havia escrit el seu valor als fitxers de configuració de cadascuna de les màquines virtuals. Això provocava que si es feia una copia exacta de la màquina, tindríem 21 màquines virtuals amb la mateixa MAC, i per tant, el DHCP els i donaria la mateixa IP, deixant inservible les màquines a nivell de connexió.

L'altre problema era la configuració del SAMBA. Degut que les carpetes que es compartien via SAMBA amb la màquina vm han d'estar protegides contra accessos de màquines que no siguin les virtuals. Això feia que haguéssim de coneixer la ip de la vm abans de posar-la en marxa per poder modificar l'arxiu de configuració smb.conf on es talla l'accés a les carpetes compartides, deixant entrar només a una IP en concret i denegant aquest privilegi a la resta.

Això es va solucionar de la següent manera: es va crear una regla per generar les IP's de les màquines virtuals i després via un script, que s'executaria només el primer cop que arranques la màquina real, es treuria tota la informació necessària per fer el canvis a l'arxiu de configuració. La regla per calcular les MAC's de les màquines virtuals és molt senzilla El tres primers bytes es deixen amb la identificació de Xen 00:16:3E. En el primer dels tres

següents bytes es codifica el tercer byte de la pròpia IP de la màquina real, es a dir, si la màquina té una IP 158.109.202.70, la tercera seria CA que correspon a la codificació del 202 en hexadecimal. El segon byte es faria igual que en el tercer. L'últim estarà destinat a distingir les diferents màquines virtuals que contingui la màquina real, per exemple 01 per la màquina vm, 02 per vw i 03 per vl. Com a exemple. si la màquina real té assignada una IP 158.109.202.70 , les MAC generades són 00:16:3E:CA:46:03 per vl, 00:16:3E:CA:46:01 per vm i 00:16:3E:CA:46:02 per vw.

Un cop ja tenim això resolt, el segón problema, que consistia en sapiguer la IP de vm abans d'arrancarla es resum en fer consultes al dhcp, suplantant la mac de vm,i aplicar els resultats obtinguts al fitxer de configuració que toca. Quan ja es tenen les dades, es torna a posar la mac original. Això es resol amb el següent script d'inici,que s'executa durant la primera arrancada de la màquina i només una vegada.

```
#!/bin/bash
firstboot=/usr/local/bin/fb

#IP & MAC of HOST
iphost=`ifconfig | grep 158.109 | cut -d ':' -f 2 | cut -d ' ' -f 1`
machost=`ifconfig | grep ^eth0 | cut -d ' ' -f 11`
ip3=`echo $iphost | cut -d '.' -f 3 `
ip4=`echo $iphost | cut -d '.' -f 4 `
ip3h=`printf "%02X\n" $ip3 `
ip4h=`printf "%02X\n" $ip4 `

#IP OF VM
ifconfig eth0 down
ifconfig eth0 hw ether 00:16:3e:$ip3h:$ip4h:01
ifconfig eth0 up
dhclient eth0 2>/dev/null
ipvm=`ifconfig | grep 158.109 | cut -d ':' -f 2 | cut -d ' ' -f 1`
ifconfig eth0 down
ifconfig eth0 hw ether $machost
ifconfig eth0 up
dhclient eth0 2>/dev/null


#Check if it's the first boot
if [ -e $firstboot ];then
   #MAC to VM
```

```
   echo vif = [ \'type=ioemu,mac=00:16:3e:${ip3h}:${ip4h}:01\' ] >> /etc/xen/vm
   #MAC to VW
   echo vif = [ \'type=ioemu,mac=00:16:3e:${ip3h}:${ip4h}:02\' ] >> /etc/xen/vw

   #MAC to VL
   echo vif = [ \'mac=00:16:3e:${ip3h}:${ip4h}:03\' ] >> /etc/xen/vl

   #SAMBA CONFIGURATION
   echo -e "[pendrive1]\nhosts allow=${ipvm}/24\nhosts deny = 0.0.0.0/0\n\
comment = Pen Drive 1\nwritable = yes\npublic = yes\n\
path = "/media/pen1"\nguest ok = yes" >> /etc/samba/smb.conf

   echo -e "[pendrive2]\nhosts allow=${ipvm}/24\nhosts deny = 0.0.0.0/0\n\
comment = Pen Drive 2\nwritable = yes\npublic = yes\n\
path = "/media/pen2"\nguest ok = yes" >> /etc/samba/smb.conf

fi

rm /usr/local/bin/fb
```

Script Start.

Capítol 2.Aplicació Virtual Ring

Descripció General.

Que es vol aconseguir

Un cop s'han posat les bases per tenir les màquines virtuals funcionant, s'ha de resoldre el problema de la connexió a les mateixes. Es va pensar que aprofitant un applet de l'empresa NoMachine, de lliure distribució i ús (no programari lliure), podríem oferir accés a les màquines, mitjançant un aplicatiu web, sense fer que els alumnes haguessin d'instal·lar un nou programa, almenys explícitament.

Alhora, un altre problema que es vol solucionar es el creixent volum d'aplicacions web que fan coses relativament senzilles, però que cadascuna d'elles té la seva pròpia url, s'ha de tornar a introduir el username i el password cada cop que s'entra a una diferent, tenen un aspecte no homogeni,etc. Per donar una solució a tot això, el que s'ha volgut crear és un framework on es donin cabuda a totes les aplicacions, ja sigui redissenyant-les, o bé posant-les dins directament sense gaires retocs al codi original, sempre que això sigui possible.

Creació d'un Framework

El mètode que es va seguir per aconseguir aquest objectiu va ser el de programar una aplicació que resolgués les necessitats de connexió contra les màquines virtuals però que al mateix temps fos programada de la manera més modular possible i amb tecnologia que fos lo suficientment potent perquè tots

els components es puguin reutilitzar en totes les altres aplicacions.

Per aquest motiu, es va fer el dissenys del que podria ser una macro aplicació, que controles totes les dades i entitats que actualment es gestionen per programari separat. Aquest disseny, havia de ser prou ampli i genèric per servir de marc, ja que no es faria una implementació del mateix, només ajudaria a decidir com es dissenyaria les aplicacions més senzilles de manera que fossin reaprofitables.

Anàlisis

Anàlisis de la Macro Aplicació.

Diagrama de casos d'ús.

En el següents diagrames podem veure els diagrames de casos d'us per alumne. S'ha de tornar a dir que en cap cas corresponen a diagrames d'ús de la aplicació presentada, sinó que s'han fet servir com a ajuda a l'hora de pendre decisions, per poder tenir una visió general, per després, donar pas a una aplicació més petita i concreta.

El diagrama presentat és el de casos d'ús de l'alumne:

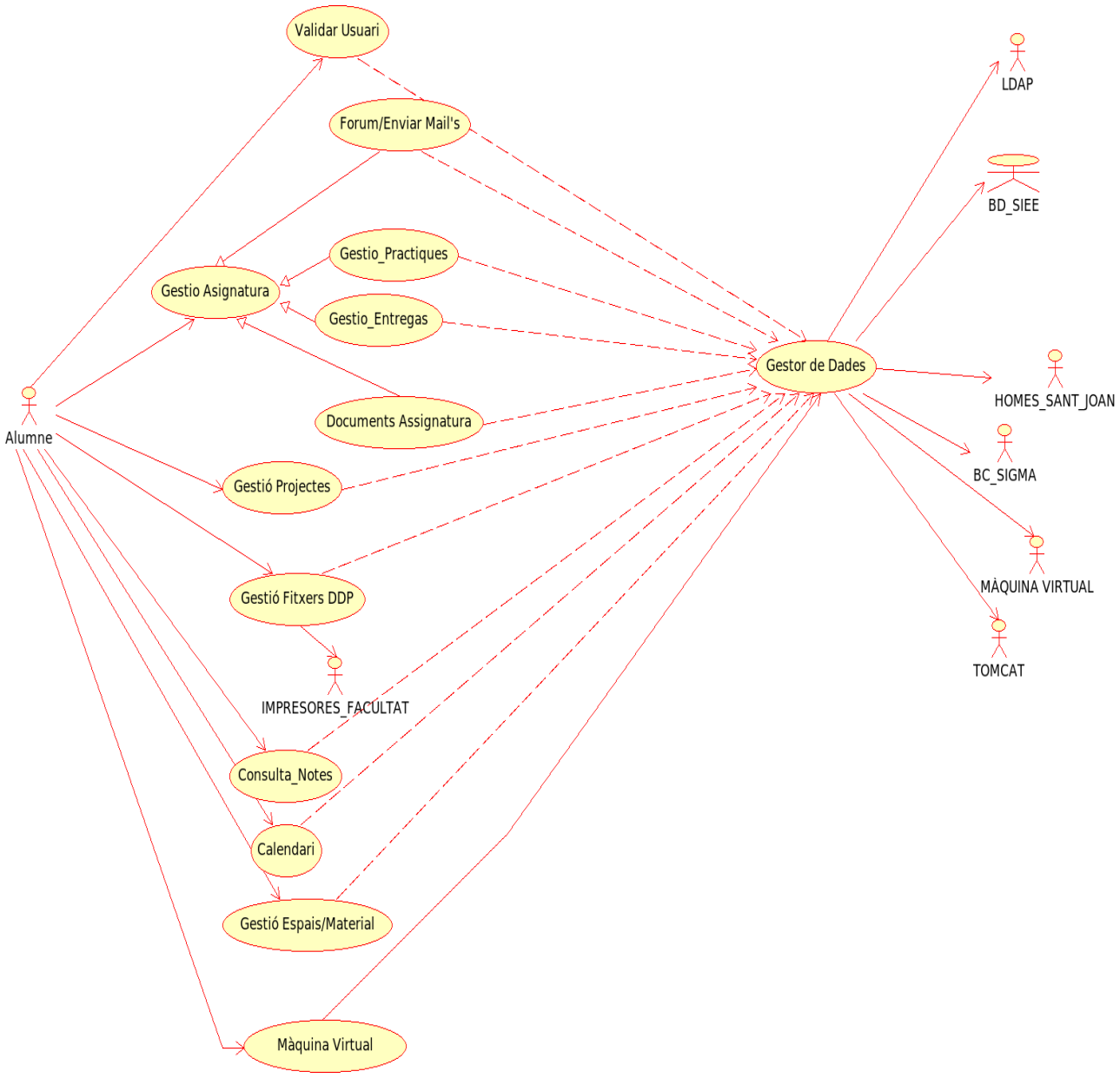


Figura 2.1.Diagrama Casos Alumne MacroAplicació

En ell podem veure les principals accions que voldríem que un alumne pogués fer: Validar-se, Gestionar una assignatura, Gestionar la presentació de projectes de final de carrera, fer ús del seu disc dur personal i accedir a les màquines virtuals. D'aquest grup la gestió d'una assignatura i la gestió de projectes es fan actualment amb dos aplicatius que es troben en producció i amb èxit. És per aquest motiu que s'ha decidit posposar el seu íntegrament dins el projecte, donant prioritat a altres aspectes molt més urgents per fer,

com seria la gestió del disc dur personal per web, en lloc de fer-ho com habitualment s'ha fet per FTP i la connexió a les màquines virtuals.

Existeixen altres cassos d'ús que seria convenient tenir integrats dins l'aplicació, com ara la consulta de notes i la gestió integrada de les dates d'examen automàticament. Aquest punt, però, no es immediat de realitzar ja que els servidors de bases de dades que contenen aquesta informació no són propietat del SIEE, són del SI (Servei d'informàtica) i l'accés esta molt limitat en el millor dels cassos, normalment degut a polítiques de seguretat.

Finalment queda el cas d'ús de la gestió d'espais i materials. Aquest cas, es vol implementar en un termini mig, després de fer l'accés a les màquines virtuals i als discs durs personals. Aquest cas d'ús ens permetria poder fer una reserva de qualsevol material o de qualsevol espai disponible dins la ETSE. Tot i ser aquest la predisposició amb la que es treballarà de començament es possible que a l'hora d'implementar-lo el material i l'espai es limitin als que s'ofereixen al SIEE per motius de competències.

Com es pot veure, a més a més del propi alumne, existeixen altres actors, que són molt importants a l'hora de la decisió de disseny del projecte. Tot la resta dels actors, són backends on s'emmagatzemen dades. Cadascun d'aquests actors, dona peu a un tros de codi en java que serà el que permeti 'dialogar' amb ell. En el cas de que vulguem validar-nos haurem de consultar a ldap per autoritzar o desautoritzar l'entrada. Es a dir que per aconseguir la modularitat que volem al nostre codi, cadascun d'aquests actors haurà de tenir un codi totalment independent i compatible entre ell per permetre crear la aplicació aglutinadora que anem anunciant.

El diagrama de casos d'ús del professor és pràcticament el mateix però dins la seva implementació es on hauríem de trobar les diferencies ja que un professors gestiona de manera diferent les assignatures a com ho faria un alumne.

Diagrames entitat-relació

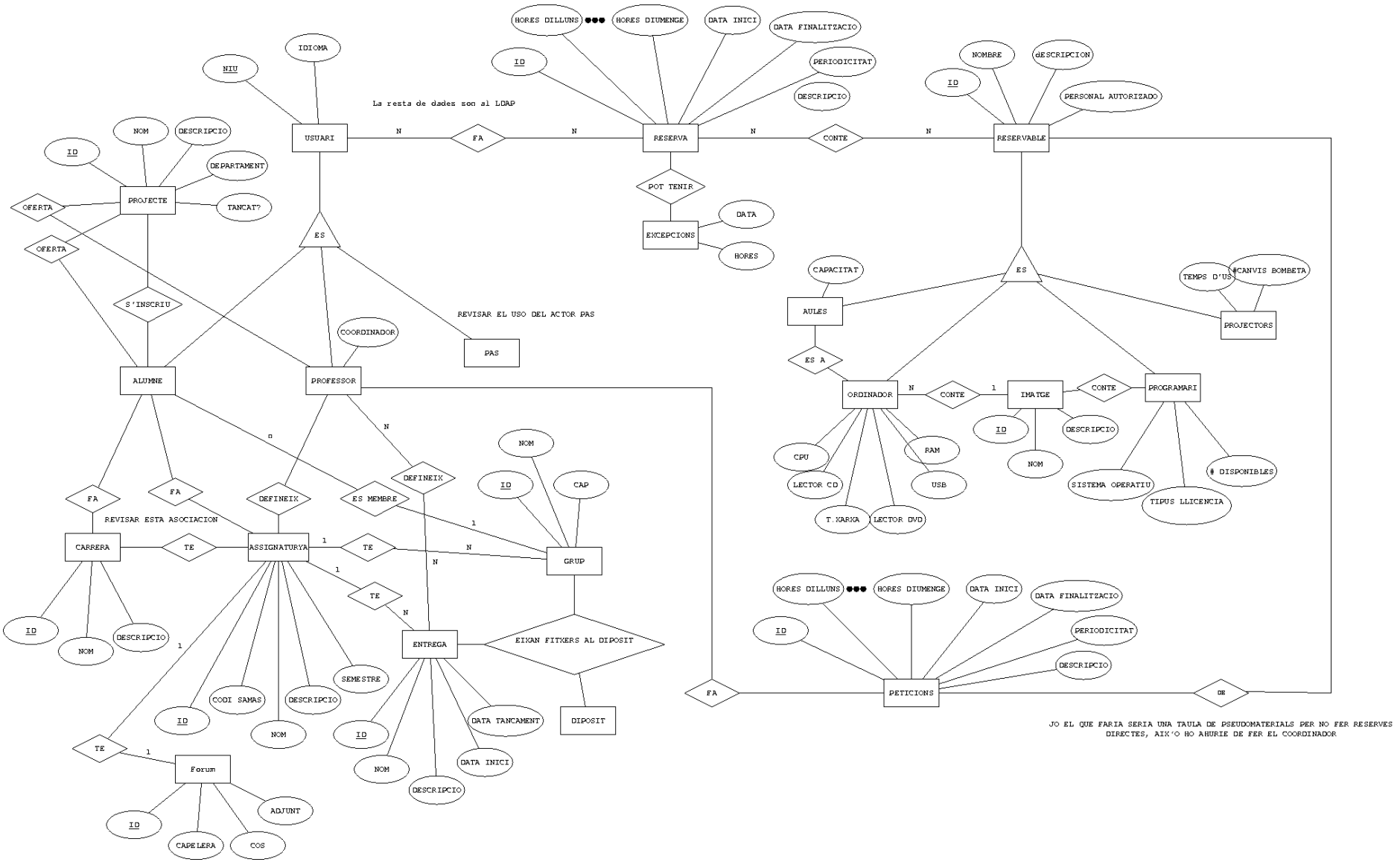

HORES DILLUNS
HORES DIUMENGE
DATA INICI
DATA FINALITZACIO
ID
PERIODICITAT
DESCRIPCIO
IDIOMA
NIU
La resta de dades son al LDAP
USUARI
N
FA
N
RESERVA
N
CONTE
N
RESERVABLE
NOMBRE
dESCRIPCION
ID
PERSONAL AUTORIZADO
NOM
DESCRIPCIO
ID
DEPARTAMENT
PROJECTE
TANCAT?
OFERTA
OFERTA
POT TENIR
EXCEPCIONS
DATA
HORES
ES
S'INSCRIU
REVISAR EL USO DEL ACTOR PAS
COORDINADOR
PAS
CAPACITAT
ES
TEMPS D'US
#CANVIS BOMBETA
AULES
PROJECTORS
ES A
ALUMNE
PROFESSOR
ORDINADOR
N
CONTE
1
IMATGE
CONTE
PROGRAMARI
n
N
DEFINEIX
ES MEMBRE
DEFINEIX
NOM
ID
CAP
ID
DESCRIPCIO
NOM
CPU
RAM
LECTOR CD
USB
T.KARKA
LECTOR DVD
SISTEMA OPERATIU
DISPONIBLES
TIPUS LLICENCIA
FA
FA
REVISAR ESTA ASOCIACION
CARRERA
TE
ASSIGNATURYA
1
TE
N
GRUP
1
N
TE
N
ID
NOM
DESCRIPCIO
ENTREGA
EIXAN FITXERS AL DIPOSIT
HORES DILLUNS
HORES DIUMENGE
DATA INICI
DATA FINALITZACIO
ID
PERIODICITAT
DESCRIPCIO
SEMESTRE
CODI SAMAS
DESCRIPCIO
ID
NOM
ID
DATA TANCAMENT
DIPOSIT
NOM
DATA INICI
DESCRIPCIO
TE
1
Forum
ID
ADJUNT
CAPELERA
COS
FA
PETICIONS
DE
JO EL QUE FARIA SERIA UNA TAULA DE PSEUDOMATERIALS PER NO FER RESERVES DIRECTES, AIX'O HO AHURIE DE FER EL COORDINADOR

Figura 2.2.Diagrama Entitat-Relació MacroAplicació.

Dins el diagrama podem trobar 4 grans entitats: usuari, reserva, assignatura i reservable. Com a usuaris podem trobar tant els alumnes com els professors. Tots dos es troben vinculats per l'entitat que agrupa les assignatures, tot i que els professors i els alumnes mantenen una relació diferent amb la entitat assignatura. Per la seva part, un alumne manté una relació amb una entitat projecte que no és un altre cosa que un projecte final de carrera. Aquesta entitat ens ajudaria a poder oferir i donar tots els serveis relacionats amb la gestió de projectes de final de carrera. Evidentment, aquesta entitat també es troba relacionada amb un professor, qui serà el director de projectes. D'altre banda, l'alumne també té relació amb una carrera, que és la que realitza que al mateix temps té assignatures on també queda relacionat l'alumne. Amb aquesta doble relació podem sapiguer que carrera cursa un determinat alumne i les assignatures que n'hi han en ella, per tant, podem simplificar opcions en el moment de presentar dades ja que no treurem dades que corresponguin a altre carreres. Les assignatures tenen entregues definides pel professor. Això facilitaria la gestió de cada assignatura que és del que s'encarrega ara per ara el psg.

Finalment, podem veurà les entitats reserva i reservable. La principal funció d'una reserva es que podem definir un temps en el que un reservable, que és una figura que conté qualsevol objecte susceptible de ser reservat, com ara un projector, un aula, etc. es trobi d'alguna manera vinculat a un usuari de manera exclusiva.

Aplicació connexió màquines virtuals.

A partir de la creació i estudi dels anteriors diagrames, es van pendre tot una serie de decissions sobre com es concretaria la implementació de la primera de totes les futures aplicacions, l'aplicatiu per poder connectar contra una màquina virtual. L'estudi de disseny que es va fer, va ser el següent.

Diagrama de casos d'ús.

El diagrama de casos d'ús es extremadament sencill i està extret integrament del que ja hem comentat anteriorment.

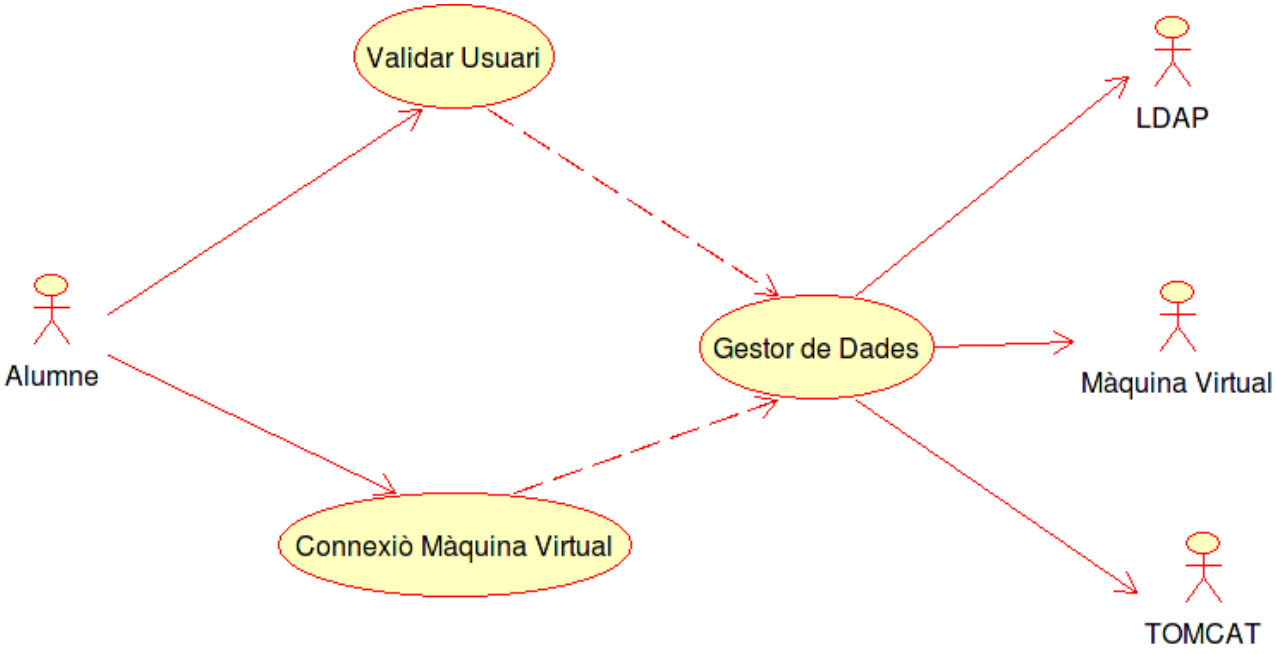


Figura 2.3.Diagrama de Casos D'ús aplicació presentada.

L'alume podrà validar-se contra l'aplicatiu, que farà servir l'arbre de ldap per això. Un cop estigui validat, Podrà connectar-se contra una màquina virtual.Per fer això el programa es recolzarà en el tomcat, per validar el perfil i finalment reenviant-lo cap a una màquina virtual concreta.

Requeriments Funcionals

Els usuaris haurien de poder connectar-se fent ús del mateix nom d'usuari i password que fan servir per identificar-se a la intranet de la uab (correu, consulta d'expedient,etc.)

El sistema ha de distingir automàticament entre professors i alumnes.

El sistema ha de presentar un menú d'opcions en funció del perfil de

l'usuari (alumne,professor o administrador), per seleccionar alguna de les opcions que se li ofereixen, com per exemple la de connectar-se a una màquina virtual.

Un cop s'hagi destriat quin dels serveis oferits de màquina virtual es vol fer ús: màquina virtual amb linux,windows, només aplicatiu,etc., l'aplicació automàticament decidirà contra quina de les múltiples màquines virtuals disponibles enviarà l'alumne. Aquesta decisió s'ha de pendre en funció d'alguns factors com ara ocupació de les màquines reals i virtuals i el número de sesions obertes per l'usuari.

L'aplicació ha de ser multilingüe. En funció de la configuració de la llista d'idiomes preferits del navegador l'aplicació retornarà tots els textes en el idioma favorit, en funció dels que tingui l'aplicació evidentment.

Requeriments No Funcionals

Totes les connexions han d'estar degudament autenticades i xifrades.

L'aplicació ha de funcionar de manera correcta, al menys, en els dos navegadors amb més quota de mercat Internet Explorer 6 i Mozilla Firefox 1.5.

El sistema s'ha d'adaptar a les limitacions d'amplada de banda que presentaràn els futurs usuaris. Considerarem com amplada de banda normal, una connexio ADSL doméstica mitjana, que podem traduir aproximadament en 1 Mb/s de baixada i 256 Kb/s de pujada.

Arquitectures de disseny emprades

La següent arquitectura de disseny és la que s'ha seleccionat per fer

màximament modular el programa resultant.

MVC (model view controler).

El principal objectiu que tracta d'aconseguir aquesta arquitectura és separar en tres capes ben diferenciades la part de programació que resol el model de dades que s'utilitza, la part que presenta aquestes dades a l'usuari i la part que gestiona les crides entre totes les parts i coordina la seva acció. A continuació veurem més detalladament cadascuna d'aquestes parts mitjançant la figura 2.4:

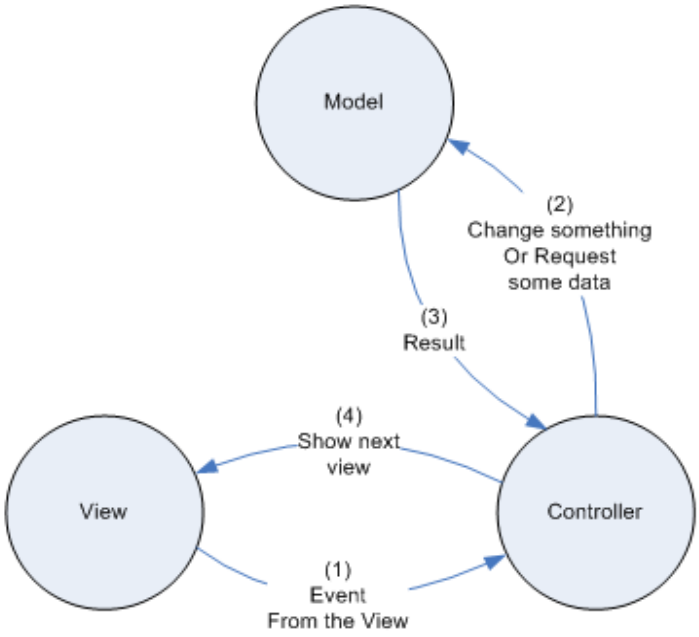


Figura 2.4.Esquema funcionament MVC

Com podem veurà en el diagrama, tot comença amb una primera view,és a dir, comença amb una primera pantalla on es presenten les primeres informacions per l'usuari. Es podria començar per la banda del controlador o del model fins i tot, però per simplificar la situació començarem amb la view. Un cop l'usuari ha vist la informació, normalment interactua amb la pantalla que se li presenta i genera algun event: per exemple escull alguna de les opcions que se li proposen i fa click en ella. Aquest event es rebut pel

controlador, qui s'encarregarà de decidir quin és el tractament adequat a l'event en qüestió. Normalment farem alguna petició al model, per actualitzar dades d'acord amb el que s'ha modificat a la vista o bé per treure informació que després farà servir la pròpia vista. Un cop hem obtingut el resultat de les dades, normalment es passen les dades obtingudes (si es que hi ha alguna) a la vista, la qual serà l'encarregada de generar la nova pantalla a partir de les indicacions del controlador.

Aquest comportament admet modificacions i, per exemple, segons la implementació del model mvc, també es permet que la part de dades faci crides a la part del view, per informar-li de que alguna dada ha sofert modificacions i que això hauria de provocar canvis en la presentació de la mateixa.

Tot aquest esquema, en el cas que ens toca ho podem interpretar com que la part del view, es la que genera tot el codi html i css que s'envia al navegador de l'usuari. Aquesta part queda resolta per Velocity. La part del controlador queda en mans d'struts que és un framework dins el model de servlet i es qui s'encarrega de cridar cadascuna de les parts implicades per resoldre les peticions dels usuaris. En l'apartat del model no es va voler fer servir alguns dels múltiples frameworks que també existeixen ja que les necessitats eren molt bàsiques com per justificar l'ús d'un dels complexes frameworks que existeixen almenys de moment. En lloc d'això s'han fet algunes solucions a mida amb Java directament com la connexió a LDAP.

Tecnologies emprades.

Java 6.0

Java és un llenguatge que es fa públic al 1995 (java 1.0) i que es basa en el paradigma de la programació imperativa orientada a objectes. També s'ha de remarcar que es fortament tipat i estàtic.

Actualment és un dels llenguatges més emprats. Una de les seves principals característiques es que al tractar-se d'un llenguatge interpretat, això ha permès fer-lo multiplataforma de manera senzilla. Un codi escrit en Java, només s'ha de compilar, per obtenir el que s'anomena bytecode i que posteriorment una màquina virtual de java s'encarregarà d'interpretar. Es per això que només cal que tinguem una màquina virtual de java per la nostre plataforma i qualsevol programa compilat de java (bytecode) funcionarà dins la nostra màquina.

Té una forta orientació cap a Internet (tot i que està capacitat per gairebé qualsevol altre tasca com gràfics 2d/3d, só ,etc.) i prova d'això són les nombroses aplicacions i frameworks escrits en java que fan tant de servidor web, com facilitan la tasca de la generació dinàmica de les mateixes.

Actualment es troba en un procés d'alliberament pel qual, Sun Microsystems ha alliberat gran part del codi font del llenguatge i s'espera que abans del final de 2007 es trobi completament alliberat i sota una llicència GPL

Struts V2.0.

Struts és un framework per la creació de aplicacions web basades en java, en concret fent ús del denominat java EE (enterprise edition). Aquest framework facilita el disseny fent ús de l'arquitectura MVC. Va ser dissenyat per Craig McClanahan i finalment donat a la fundació Apache on es troba actualment integrat dins el projecte Apache Jakarta Project i és coneix amb el nom de Jakarta Struts.

La seva llicència és la Apache 2.0 license que es pot considerar open source, però que es incompatible amb la llicència GPL2.0, tot i que s'espera que sigui compatible amb la GPL v3.0, d'acord amb l'ultim esborrany d'aquesta.

Velocity 1.5

Velocity es el que es coneix com un motor de plantilles. Esta escrit amb Java i proveïx d'un llenguatge propi capaç de referir-se als objectes de Java. Al fer-se servir juntament amb Struts, Velocity s'encarrega de resoldre l'apartat del view dins la terna MVC, aportant-nos aquesta capacitat de referir-se objectes java, podent fer crides als mateixos,obtenint dades i integrant-les dins la plantilla que s'ha creat per determinada plana web.

Es totalment compatible amb altres solucions, potser més típiques, dins l'apartat de view, com per exemple JSP. En aquest cas, la plantilla podria donar com a resultat un jsp que tornaria a ser interpretat pel servlet de Java. Velocity pot generar moltes més sortides com XML, PostScript, SQL,etc.

La seva llicència al igual que Struts és la Apache 2.0 license.

Scriptaculous.

Scriptaculous és un framework de treball per ECMAScript. Es troba construït sobre un altre framework anomenat Prototype, també d'ECMAScript. La seva principal tasca és la de facilitar la programació de efectes visuals i interacció amb l'usuari dins els navegadors que implementin ECMAScript, com per exemple Mozilla amb javascript.

CSS.

El CSS (Cascade Style Sheets) és un llenguatge que es fa servir per definir l'estil, o dit d'una altre manera, l'aspecte gràfic d'un document escrit en llenguatge de marques com ara l'html. Fent ús del CSS podem separar la creació d'una plana web de l'aspecte que té i posteriorment introduir canvis amb javascript.

Eclipse

Eclipse és un IDE (Integrated Development Environment) escrit en Java, que permet desenvolupar programes en multitud de llenguatges com ara Java, C, C++,etc. El desenvolupament amb eclipse facilita molt la creació de servlets ja que té una integració molt bona amb el desenvolupament web i dona moltes facilitats per programar amb struts,etc.

La seva llicència és: Eclipse Public License, que és considerada una llicencia d'open source.

Umbrello

Umbrello és una eina de creació de diagrames UML que ens ha ajudat a la creació del disseny conceptual de l'aplicació.

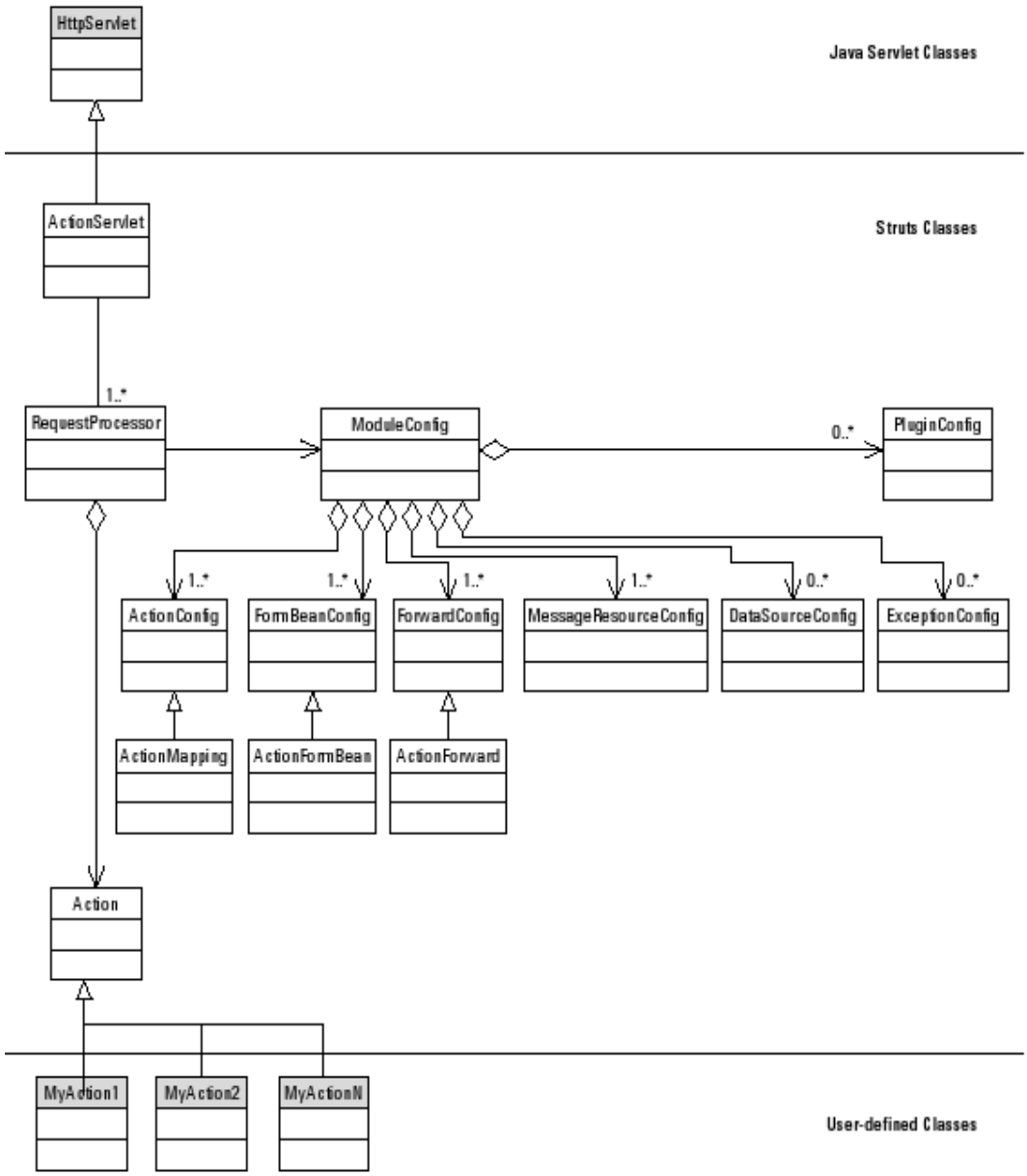
Descripció d'implementació

Primerament veurem breument com funciona Struts, per poder ubicar la feina de programació de manera correcta. Després es passarà a analitzar de manera general i breu cadascun dels mòduls que s'han implantat per separat, oferint cadascun d'ells una funcionalitat diferent i que en el seu conjunt ens donaran l'aplicació web que volíem

Esquema de funcionament d'Struts.

Per poder comprendre com s'ha fet la implementació, s'ha de tenir clar com funciona Struts, ja que es qui dona la base de codi sobre la que treballarem i així podrem ubicar millor el codi del projecte. Ens recolzarem sobre el següent diagrama per explicar el funcionament:

Figura 2.5.Diagrama UML de les principals classes de Struts.



Quan arriba una petició al servidor web, aquest deriva la mateixa cap al servlet de Java, en concret contra la classe HttpServlet. Aquesta classe es troba extesa per ActionServlet que ja forma part,sent la primera en entrar en acció, de la família de classes de Struts. Bàsicament, ActionServlet, s'encarrega de inicialitzar totes les classes restants del diagrama, que han de contenir dades desde l'inici i que són als arxius de configuració. Es per això que la classe ActionServlet llegeix diversos arxius de configuració, entre ells el principal struts-config.xml, per seguidament introduir aquestes dades a les diferents classes que inicia.

La següent classe en entrar en acció és RequestProcessor. Aquesta classe és l'encarregada de gestionar realment tota la part del controller i és el motor d'struts. Quan arriba una petició, s'encarrega de consultar a les classes pertinents, per sapiguer cap a on a de derivar la petició en funció de la consulta i de recopilar tota la informació necessària provinent del view per posar a disposició de la Action, que veurem més endavant.

La classe ModuleConfig i totes les seves derivades són les que emmagatzemen tota la informació de configuració del controlador i és on es recolza bàsicament RequestProcessor per pendre les decisions.

Finalment trobem la classe Action. Aquesta classe serà de la que heretarem per fer les nostres Actions. Les nostres Actions no són més que el que volem que faci la nostra aplicació en funció d'algunes peticions. Per exemple, quan algú intenti validar-se, arrivarà gràcias a Struts i als fitxers de configuració que haurem preparat expressament, a una action que s'encarregarà de validar o no l'usuari.

Codificació del projecte.

Fitxer Struts-config.xml

Mitjançant aquest fitxer, Struts es capaç de sapiguer relacionar cadascuna de les peticions amb la seva corresponent action. Aquest fitxer esta escrit en xml i en el nostre aplicatiu és el següent:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<!DOCTYPE struts-config PUBLIC
        "-//Apache Software Foundation//DTD Struts Configuration 1.1//EN"
        "http://jakarta.apache.org/struts/dtds/struts-config_1_1.dtd">

<!-- This is the Struts configuration file for the Login application -->
<struts-config>
```

```
<!--       <controller> <set-property processorClass="login.LoginRequestProcessor"/> </controller> -->


  <!-- ========== Form Bean Definitions =================================== -->
  <form-beans>
    <form-bean name="loginForm" type="login.Forms.LoginForm"/>
  </form-beans>

      <!-- ========== Action Mapping Definitions ============================== -->
      <action-mappings>

      <action path="/login" type = "login.Actions.LoginAction" name = "loginForm"
         scope = "session"
         input = "/Pages/login/login.vm"
         validate = "true">
           <forward name="failure" path="/Pages/login/errorlogin.vm"/>
           <forward name="success" path="/Pages/login/initial.vm"/>
      </action>

     <!--
      <action path="/virtualmachine" name="VirtualMachineForm" type="virtualmachine.Actions.VirtualMachineAction"
scope="session" validate="false" roles="student">
      <forward name="go" path="/alumne/info.jsp" />
      </action>
      -->


       <action path="/virtualmachine" type = "virtualmachine.Actions.VirtualMachineAction"
         scope = "session"
         input = "/Pages/login/login.vm"
         validate = "true" roles="student teacher">
         <forward name="go" path="/Pages/Applications/VirtualMachine/VirtualMachine.vm"/>
      </action>

       <action path="/virtualmachine2" type = "virtualmachine.Actions.VirtualMachineAction2"
         scope = "session"
         input = "/Pages/Applications/VirtualMachine/VirtualMachine.vm"
         validate = "true" roles="student teacher">
         <forward name="go" path="/Pages/Applications/VirtualMachine/VirtualMachineApplet.vm"/>
      </action>



       <action path="/print" type = "virtualmachine.Actions.VirtualMachineAction"
         scope = "session"
         input = "/Pages/login/login.vm"
         validate = "false" roles="student teacher">
         <forward name="go" path="/Pages/Applications/Printer/upload.vm"/>
```

```
      </action>

       <action path="/printcontrol" type = "printer.Actions.PrinterAction"
         scope = "session"
         input = "/Pages/login/login.vm"
         validate = "false" roles="student teacher">
         <forward name="go" path="/Pages/Applications/Printer/printcontrol.vm"/>
       </action>

    </action-mappings>

    <controller processorClass="login.LoginRequestProcessor"/>

  <!-- ========== Message Resources Definitions ========================== -->
   <message-resources null="false" parameter="ApplicationResources"/>


        <!-- ================================================ Plug Ins Configuration -->

<!-- ========================================================= Tiles plugin -->

  <plug-in className="org.apache.struts.tiles.TilesPlugin" >
    <!-- Path to XML definition file -->
    <set-property property="definitions-config"
                  value="/WEB-INF/tiles-defs.xml" />
    <!-- Set Module-awareness to true -->
    <set-property property="moduleAware" value="true" />
  </plug-in>

  <plug-in className="login.StartUp" />

</struts-config>
```

Fitxer Struts-config.xml

Lo primer que definim en aquest fitxer és les beans que farem servir, en aquest cas la bean per fer la comunicació de les dades d'usuari (uid i password) a l'aplicatiu. Això ho podem veurà a la linea form-bean name="loginForm" type="login.Forms.LoginForm"/>.D'aquesta manera Struts ja sap que té una bean i com ara veurem, després l'informarem de quan usar-la. A continuació fem el mapeig entre pàgines demanades i pagines que es serveixen i l'action associat. si mirem l'acció que té determinada quan es demana la pàgina login.do (l'extensió no es posa, és implícita) que és troba en la linea que comença per <action path="/login" type = "login.Actions.LoginAction" name = "loginForm" i següents fins que es

tanca amb </action>. Mitjançant la linea de action el que fem és relacionar la demanda de la pàgina login.do amb la action login.Actions.LoginAction que és la classe que explicarem en el modul de login i que gestiona l'ingrés a l'aplicatiu. Al mateix temps, també relacionem aquesta pàgina amb el seu corresponent form per passar les dades de la view cap a l'action. Finalment, també informem de que ha de fer quan acabi l'acció, en aquest cas mitjançant dues accions de forward que hem anomenat success i failure que són les següents en cas de que la validació hagi tingut èxit o hagi fracassat respectivament. També podem indicar que segons sigui el rol de la persona que sol·liciti la pàgina li otorgui o denegui el permís d'accedir. Això es fa amb un plugin per Struts anomenat Tiles i la següent instrucció dins la definició de la relació de la pàgina amb el action i el bean: roles="student teacher".

La següent part és molt important ja que modifiquem el comportament normal d'Struts. En la linea on posa: <controller processorClass="login.LoginRequestProcessor"/>, el que estem informant es que en lloc de fer servir el RequestProcessor que el propi Struts porta per defecte farem servir un fet específicament per aquesta aplicació, basat en el d'Struts i amb característiques del plugin Tiles,que veurem més endavant.

Finalment la última part que val la pena comentar és en la que definim el que s'anomena message-resources: <message-resources null="false" parameter="ApplicationResources"/>. Aquesta definició indica on es troben els fitxers de message-resources que són els encarregats de contenir el text que es presenta en la web a l'usuari. Mitjançant aquest sistema aconseguim que l'aplicació sigui multilingüe ja que el servlet detecta automàticament la llista de prioritats en idioma que es configura a la majoria de navegadors i si té algun d'aquest retorna automàticament la pàgina en aquell idioma. En cas de no tenir cap dels seleccionats es donarà el que s'hagi posat per defecte, en aquest cas el català. Per aconseguir tot això només hem de posar el mateix fitxer, ApplicationResources.properties, però acabat amb un guió baix i les lletres que identifican a cada país, es per espanya, ca per català (ApplicationResources_ca.properties), etc.

LoginRequestProcessor

Com ja s'ha comentat, no s'ha fet servir el RequestProcessor habitual d'Struts. El motiu d'aquest canvi és fàcil: implementar un sistema de seguretat que eviti l'accés a les pàgines sense que no s'estigui logat al sistema i sense que sigui el perfil pertinent per veurà la pàgina en qüestió. Per fer això s'ha implementat un nou RequestProcessor anomenat LoginRequestProcessor que el que fa es heretar de TilesRequestProcessor que és el plugin que comentàvem en el fitxer struts-config.

```
package login;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.apache.struts.action.ActionMapping;
import org.apache.struts.tiles.TilesRequestProcessor;

import login.Forms.*;

public class LoginRequestProcessor extends TilesRequestProcessor {

	protected boolean processRoles(HttpServletRequest request,
			HttpServletResponse response, ActionMapping mapping) {
		String roles = mapping.getRoles();
		if (roles == null){
			return true; // No roles for this request
					// acces granted
			}

		HttpSession session = request.getSession();
		if (session!=null){
			LoginForm user= (LoginForm) session.getAttribute("user");
			if (user!=null){
				int profile = user.getProfile();
				if (roles.contains(Constants.PROFILE_DESCRIPTIONS[profile])) {
					return true;
				}else{
					return false; //with the profile of this User
							//we can't enter.
				}
```

```
                        }else{
                                return false;       //user=null. This error NEVER
                                                                //succes.
                        }
                }else{
                        return false; //The user isn't loged in.
                        }
        }
}
```

Classe LoginRequestProcessor

En aquesta classe podem veura com recollim els roles que poden accedir a la pàgina en la linea `String roles = mapping.getRoles();`. si no hi ha cap role definit al fitxer Struts-config.xml, es que es d'accés lliure, es a dir, per tothom encara que no estigui validat. Després mirem el roles de la sessió, que recordem es troba dins el perfil guardat com una instància de la classe LoginForm. Si coincideixen amb qualsevol dels rols permesos continuarem endavant, sinó generarem un error.

Modul de Login

El mòdul de login tracta de resoldre la necessitat de validar un usuari o no, en funció de si el binomi nom d'usuari i password és correcta. Recordem que aquest nom d'usuari i passwords han de ser els mateixos que els usuaris fan servir per validar-se contra el correu institucional de la universitat. Per portar a bon terme aquest punt, el que s'ha fet és que el servlet sigui capaç d'atacar a un servidor de LDAP que es troba disponible gràcies al projecte Sant Joan.

El primer fitxer que veurem és el de LoginForm.java. Aquesta classe és una bean que hereta de la classe ActionForm. Serveix bàsicament per dipositar tres dades per comunicar-se entre la view, el controller i el model, aquestes dades són: uid (identificador d'usuari), password, profile (perfil d'usuari que pot ser alumne, professor...).

```
package login.Forms;

	import javax.servlet.http.HttpServletRequest;
	import org.apache.struts.action.ActionError;
	import org.apache.struts.action.ActionErrors;
	import org.apache.struts.action.ActionForm;
	import org.apache.struts.action.ActionMapping;


	public class LoginForm extends ActionForm {

		private String uid;
		private String password;
		private int profile;


		public void reset(ActionMapping mapping, HttpServletRequest request){
			uid="";
			password="";
		  }

		public String getPassword() {
			  return password;
			}
		public String getUid() {
			return uid;
			}
		public int getProfile() {
			return profile;
			}
		public void setPassword(String string) {
			password = string;
			}
		public void setUid(String id) {
			uid = id;
			}
		public void setProfile(int id) {
			profile = id;
			}

	}
```

Classe LoginForm.java

Després tenim la classe UserBean.java. Aquesta classe és l'encarregada de fer les consultes contra l'arbre LDAP per validar els usuaris, quan es crida al

seu mètode.

```
package login.model;

import javax.naming.directory.DirContext;


public class UserBean {


	public int validateUser(String userName, String password)
	{
		int profile=login.Constants.UNKNOW_PROFILE;

		try {
			//Try top connect to Ldap and search the user
			Hashtable env = new Hashtable(11);
			env.put(Context.INITIAL_CONTEXT_FACTORY , "com.sun.jndi.ldap.LdapCtxFactory");
			env.put(Context.PROVIDER_URL, login.Constants.LDAP_SERVER);
			env.put("com.sun.jndi.ldap.connect.pool","true");

			DirContext ctx= new InitialDirContext(env);
			SearchControls ctls = new SearchControls();
			ctls.setSearchScope(SearchControls.SUBTREE_SCOPE);
			ctls.setReturningObjFlag(true);
			ctx = (DirContext) (new InitialDirContext (env));
			NamingEnumeration answer = ctx.search("o=sids","(uid=" + userName  + ")", ctls);

			//Find a set of user, try to match user/password
			while (answer.hasMore()) {
				SearchResult sr = (SearchResult) answer.next();
				env.put(Context.SECURITY_CREDENTIALS, password);
				String DN = sr.getName() + ",o=sids";
				env.put(Context.SECURITY_PRINCIPAL, DN);
				try {
					DirContext ctx2 = new InitialDirContext(env);
					ctx2.close();
					//Password and user match.Find the profile of the user
					if (DN.contains(login.Constants.LDAP_OPERATORS_TREE)) {
						profile = login.Constants.ADMINISTRATOR_PROFILE;
					} else {
						profile = getAttributes(sr.getAttributes());
					}

				}catch (Exception e) {
					// parella password / usuari incorrectes
				}
			}
			ctx.close();
```

```
                } catch (NamingException e) {
                        e.printStackTrace();
                }

                return profile;


        }


        private int getAttributes  (Attributes attrs){
                int profile = login.Constants.UNKNOW_PROFILE;
                if (attrs == null) {
                //          System.out.println("No te atributs");
                } else {

                        try {
                                if (!attrs.get("cp").toString().equals("cp: 0")) {
                                        profile = login.Constants.TEACHER_PROFILE;
                                } else {
                                        profile = login.Constants.STUDENT_PROFILE;
                                }

                                /* for (NamingEnumeration unenum = attrs.getAll(); unenum
                                                .hasMore();) {
                                        Attribute attrib = (Attribute) unenum.next();
                                         System.out.println("----- " + "ATRIBUT :" + attrib.getID());
                                          for (NamingEnumeration e = attrib.getAll(); e.hasMore();)
                                                System.out.println("\t\t       = " + e.next());

                                }*/

                        } catch (NullPointerException npe) {}


                }
                return profile;

        }

}
```

Classe UserBean

Com podem veure, obrim una connexió contra un arbre ldap amb les classes que ens proporcionen el mateix java 6 que es troben dins de javax.naming.directory.DirContext. Hem de notar també que aquesta connexió

es fa en mode de pool, és a dir, que cadascuna de les consultes que es fan contra l'arbre de java no obra una nova connexió sinó que la pròpia màquina virtual de Java gestiona la obertura i quan es torna a fer una nova consulta, es reaprofita,en cas de poder-se, la connexió original. En cas de que el password i el usuari siguin correctes, recuperarem el seu perfil (professor, alumne...) que es el que retornarem.

LoginAction

Finalment veurem la classe LoginAction qui es l'encarregada d'efectuar l'operació, recolzant-se en les dos classes anteriorment explicades, de validació d'usuari.

```
package login.Actions;
import login.Forms.*;
import login.model.*;


import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.apache.struts.action.Action;
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

public class LoginAction extends Action {

	public ActionForward execute( ActionMapping mapping,ActionForm form,HttpServletRequest
request,HttpServletResponse response) throws Exception
	{
		// create a new LoginBean with valid users in it
		UserBean ub = new UserBean();
		// check to see if this user/password combination are valid
		LoginForm lf= ((LoginForm)form);
		int profile= ub.validateUser( lf.getUid(),lf.getPassword() );
		if(profile !=0){
			HttpSession session=request.getSession();
			lf.setProfile(profile);
			session.setAttribute("user",lf);
			request.setAttribute("user",profile);
```

```
                    return (mapping.findForward("success"));
                    }
                else      // username/password not validated
                    {
                    //create ActionError and save in the request
                    ActionErrors errors = new ActionErrors();
                    ActionError error = new ActionError("error.login.invalid");
                        errors.add("login",error);
                        saveErrors(request,errors);
                        request.setAttribute("profile", profile);
                        return (mapping.findForward("failure"));
                    }
                }
}
```

Classe LoginAction

Com podem veure, el que fem es intentar recuperar el perfil del usuari, que té les seves dades dipositades dins una instància de la clase LoginForm, recolzant-nos en el mètode ValidateUser de la classe UserBean, també anteriorment explicada. Si tot va bé guardarem el LoginForm amb el perfil actualitzat dins la sessió perquè es pugui recuperar en qualsevol moment que l'usuari faci ús de l'aplicatiu no tingui que tornar a identificar-se.

Modul de Màquines Virtuals

Per implementar la connexió a les màquines virtuals, bàsicament es realitza tota la feina desde la classe VirtualMachineAction2.

```
package virtualmachine.Actions;

import virtualmachine.Forms.*;
//import virtualmachine.model.*;


import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import login.Forms.LoginForm;
import login.model.UserBean;

import org.apache.struts.action.Action;
```

```
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

import login.StartUp;

public class VirtualMachineAction2 extends Action{

	public ActionForward execute( ActionMapping mapping,ActionForm form,HttpServletRequest
request,HttpServletResponse response) throws Exception
	{

		VirtualMachineForm ub = new VirtualMachineForm();
		ub = (VirtualMachineForm) form;
		String uid = ((LoginForm) request.getSession().getAttribute("user")).getUid();
		System.out.println("USUARI DINS LA SESSIO ANALITZADA"+uid);
		//loop for find if user is log now
		for (String [] people : login.StartUp.pip){
			if ( uid.equals(people[0]) ){

				System.out.println("el tio" + uid + "que en el vector es:"+ people[0] +" tiene la
maquina" + people[1]);
				return (mapping.findForward("go"));
			}
		}
		System.out.println("atributo vm=" + ub.getVm() );

		switch ( ub.getVm() ){
		case 1:  //Complet Windows Desktop
			//We need to choose a PC and generate de session.nxs
			System.out.println("CASE 1");
			for (String[] people2 : login.StartUp.pip){
				System.out.println("DINS EL FOR DE CASE1");
				if (people2[0] == null){
					people2[0]=uid;
					people2[1]= FreeMachine(1);
					System.out.println("LI DONO el PC: "+ people2[1] );
					request.setAttribute("ip",people2[1]);
					break;
					}
				}
			break;
		case 2: //Complet Linux Desktop
			System.out.println("CASE 2");
			//We need to choose a PC and generate de session.nxs
			break;
		case 3: //Maple aplication only
			System.out.println("CASE 3");
```

```
                //We need to choose a PC and generate de session.nxs
                break;
        case 4: //Matlab aplication only
                System.out.println("CASE 4");
                //We need to choose a PC and generate de session.nxs
                break;
        case 5: //Autocad Only
                System.out.println("CASE 5");
                //We need to choose a PC and generate de session.nxs
                break;
        default:
                System.out.println("CASE DEFAULT");
                //Error Option
                break;

        }

                return (mapping.findForward("go"));
                }
//      else    // username/password not validated
        //      {
                //create ActionError and save in the request
        //      ActionErrors errors = new ActionErrors();
                //ActionError error = new ActionError("error.login.invalid");
                //      errors.add("login",error);
                //      saveErrors(request,errors);
                //      request.setAttribute("profile", profile);
                //      return (mapping.findForward("failure"));
        //      return (mapping.findForward("failure"));
                //}

private String FreeMachine(int kindofmachine){
        //1 Windows Machine
        //2 Linux Machine

        String ips;
        boolean inpool=false;
        int ip;

        for (ip=login.Constants.INITIAL_IP_POOL; ip< login.Constants.FINAL_IP_POOL; ip=ip+3){
                System.out.println("IP analitzat:" + ip);

                switch (kindofmachine){
                case 1:
                        ip=ip+1;
                        break;
                case 2:
                        ip=ip+2;
                        break;
                }
```

```
                for(int i=0;i<login.Constants.CONCURRENT_PEOPLE_IN_POOL;i++){
                        System.out.println("i:"+i);
                        ips = String.valueOf(ip);
                        if (ips.equals(login.StartUp.pip[i][1])){
                                //i=login.Constants.CONCURRENT_PEOPLE_IN_POOL;
                                inpool=true;
                                break;
                        }
                }
                if (!inpool){
                        //ip=login.Constants.FINAL_IP_POOL;
                        break;
                }

                inpool=false;
                switch (kindofmachine){
                case 1:
                        ip=ip-1;
                        break;
                case 2:
                        ip=ip-2;
                        break;
                }
        }
        ips=String.valueOf(ip);
        return ips;
}

        }
```

Classe VirtualMachineAction2

La nostre action, es recolza en una estructura molt sencilla anomenada pip (de people in pool) i que és un vector de dues dimensions. La primera dimensió correspon amb el màxim número de persones que poden estar connectades. En un primer moment, per fer proves, es permetran un màxim de 30 persones. Cada entrada d'aquest vector té un altre vector de dues entrades. En la primera, identifiquem l'usuari amb el seu uid. En la segona el PC que li hem donat. L'array, quan s'arrenca la aplicació s'inicialitza a zero.

Amb aquesta estructura al cap, podem començar a analitzar el codi. Primer de tot ens preocuparem de mirar que l'usuari que ens acaba de

demanar una màquina virtual no tingui una previament assignada. Això és fa al bucle que hi ha tot just després del comentari "//loop for find if user is log now". En ell, recorrem tot el bucle per trobar als usuaris als que ja hem donat una màquina virtual i comparem el seu uid amb el del usuari que ens acaba de demanar aquest servei. Si no té una màquina assignada passem a assignar-li una. Ara el que ens queda es assignar-li una màquina en funció del que s'hagi demanat: escriptori de windows, de linux, aplicatiu... Per fer això s'ha decidit aïllar aquesta lògica en una altre funció, FreeMachine. Quant FreeMachine ens retorni la ip seleccionada, tot seguit s'explicarà el seu funcionament, el que farem és passar a la view aquesta informació, que construirà una pàgina web de connexiò que apuntarà cap a un arxiu de configuració previament creat on s'especificarà la ip de la màquina.

La funció FreeMachine rep com a parametre la opció de l'usuari referent al tipus de màquina virtual que vol i retorna la ip de la màquina seleccionada. Per aconseguir-ho fem un doble bucle. En el primer bucle recorrerem totes les ip's del pool de màquines virtuals. En el segon, que es troba dins el primer, el que fem es recorre de nou l'estructura pip en busca de la ip que hem seleccionat al primer bucle. Si la ip no es troba dins pip vol dir que podem assignar la ip, sino continuarem amb el bucle de ip's per comprovar la disponibilitat de la següent. Un cop tinguem una ip lliure, retornarem aquesta.

Capítol 3.Conclusions

Punts aconseguits

Si separem el projectes en dues parts, la de sistema on es configurava una maquina host on hi coexistien 3 màquines virtuals i la de disseny i programació on es creava una aplicació web per poder accedir a les màquines virtuals, veurem que els punt aconseguits amb respecte els proposats són:

En la part de sistemes podem parlar de que amb respecta als requeriments funcionals que totes tres màquines virtuals han estat portades a terma amb relatiu èxit ja que totes ofereixen, en major o menor grau, totes les funcionalitats que se li demanaven, tant en l'apartat d'aplicatius per alumnes, com en velocitat per reproduir vídeo, com per manegar dispositius usb,etc. En quant al requeriment no funcional de que tot això funciones sobre el Dell Optiplex es van solventar tots els problemes que van aparèixer amb controladors de dispositius i ara per ara funciona de manera correcta.

En quant a la part de disseny i programació s'han aconseguit també molts punts: els usuaris es poden autenticar amb el mateix niu i password que fan servir al correu electrònic i altres serveis de la universitat. Es fa distinció entre els diferents perfils que es troben a LDAP (professor, alumne i administrador) tot i que encara només es una distinció interna ja que a nivell de la web no es mostra cap diferència. S'ha creat un menú que de moment mostra 3 opcions tot i que la única realment desenvolupada es la de connectar a les màquines virtuals. També s'ha aconseguit connectar contra les màquines virtuals fent ús del modul de màquina virtual i l'applet de NomMachine. El multilingüisme esta plenament aconseguit i depenent de les preferències del navegador el servlet automàticament canviarà el text que mostra a la pàgina web d'un idioma a un altre. Actualment es troba en 3 idiomes: català, castellà i

anglès tot i que introduir nous idiomes es a nivell tècnic extremadament senzill gràcies al sistema seguit per portar a terma aquest requeriment

Si mirem una mica als requeriments no funcionals que ens havíem plantejat veiem que les connexions contra l'aplicatiu web s'han aconseguit xifrar i que la web té un certificat, tot i que es un signat per nosaltres mateixos i no té gaire validesa. Funciona correctament amb Firefox 1.5 i Internet Explorer però per aquest últim s'han de crear uns css propis perquè tot i funcionar, la pàgina web no es veu correctament. En quant a l'amplada de banda necessària per poder connectar-se a les màquines virtuals, s'ha intentat configurar per adaptar-lo a les necessitats i en les proves realitzades , el seu comportament ha estat satisfactori, tot i que el seu funcionament real continua sent un incògnita fins que no es desplegui tot el sistema, entri en producció i es rebi el feedback dels usuaris.

Punts per aconseguir

En quant a les màquines virtuals, són pocs els punts per aconseguit que han quedat per fer. L'únic una mica més notable és el d'aconseguir un sistema millor per punxar els pen-drives i que surtin a la màquina virtual sense cap intervenció manual. Malauradament per aconseguir millorar aquest punt, haurem d'esperar millores en el propi sistema de virtualització, es a dir, en Xen.

En quant a l'aplicatiu web, el principal punt que encara no s'ha aconseguit ha estat el balanceig de carrega. En un primer moment, es va confiar en el balanceig de carrega que porta freenx, però degut a que no era gaire confiable ja que portava errors en el codi que vam haver de sol·lucionar nosaltres i que no podiem d'aquesta manera obtenir estadístiques es va decidir implementar nosaltres mateixos el sistema de balanceig. Es per aquest motiu que el sistema encara no es troba molt avançat i el punt a aconseguir és el de millorar l'algorisme de carrega, que ara per ara es un round-robbin.

També es troba en desenvolupament més funcionalitats dins el menú de

selecció a part de les màquines virtuals. En concret s'ha fet una prova de concepte per permetre a l'usuari enviar treballs d'impressió amb formats concrets, pdf,ps,gif,txt,etc. contra la impressora OCE de la ETSE. Altre funcionalitat que encara no ha passat del pur concepte en un paper es el de poder manegar els fitxers dels emmagatzemament personal que ofereix el SIEE als alumnes des de la web, tot i que ja té la seva icona a l'aplicatiu.

Conclusió

Com a conclusió final podem dir que la dimensió del projecta ha estat molt gran per tant sols una persona. El projecte tenia 2 parts molt diferenciades que cadascuna tenia una dificultat bastant elevada. En la part de sistema, la tecnologia de virtualització encara es troba en una etapa on els canvis es succeïxen a gran velocitat i que no tot funciona amb la senzillesa que seria recomanable. En quant a la tecnologia per obtenir escriptoris remots, al tractar-se d'una versió completament lliure, desenvolupada per un únic home tot i funcionar correctament, tampoc té totes les funcionalitats i facilitats que ofereixen altres programaris. Tot això, juntament amb tota la bateria de proves dels diferents sistemes de virtualització han fet que s'haguessis de dedicar nombroses hores.

En l'apartat de l'aplicatiu web, trobem que es troba encara una mica massa immadur i que s'ha de continuar treballant en ell per poder oferir una pàgina web molt millor. En aquest sentit, el gran conjunt de noves tecnologies sobre el que es sustenta ha fet que tingui una bona base per continuar desenvolupant sobre l'aplicatiu, però alhora ha fet que es requerís d'uns estudis previs en totes aquestes tecnologies que també han retrasat molt el desenvolupament.

Tot i aquest gran volum de treball i hores dedicades, creiem que el resultat final és totalment satisfactori i que compleix el principal objectiu amb escreix que era assentar unes bones basses de tot aquest projecte per continuar desenvolupant el mateix després d'aquest projecte final de carrera dins el SIEE.

Vies d'ampliació i continuïtat.

Les primeras vies de continuitat d'aquest projecte és la maduració del mateix, en especial amb respecte a l'apartat de l'aplicatiu web. En aquest aplicatiu s'han de millorar l'aspecte visual de la plana web, la documentació técnica, els comentaris dins el propi còdi,etc. Ara per ara tenim un aplicatiu petit i funcional, però tenint en compte el creixement que se li vol donar, s'ha de deixar acabat amb una qualitat molt més gran per tenir una base plenament solida i d'això tracta la maduresa de la que parlem.

D'altre banda en el futur, un cop s'hagi madurat el projecte inicial, es continuaran creant aplicacions, entre ellas, ja s'ha parlat o fins i tot s'han fet proves conceptuals de: impressió desde la web (com ja s'ha comentat), manipulació de fitxers de l'espai compartit,etc.

BIBLIOGRAFÍA

WWW(titulo,autores,url,ultima fecha visita)

[1]Descripció de IVT-X http://www.intel.com/technology/itj/2006/v10i3/1-hardware/5-architecture.htm

[2]Descripció arquitectura VirtualBox: http://www.virtualbox.org/wiki/VirtualBox_architecture

[3]Descripció de Productes Vmware i algunes característiques
http://es.wikipedia.org/wiki/VMware
http://www.vmware.com/elqNow/elqRedir.htm?ref=http://www.vmware.com/pdf/virtualization.pdf
http://www.vmware.com/download/eula/server.html

[4]XEN:
http://www.cl.cam.ac.uk/research/srg/netos/xen/architecture.html

Llibres

[5]JLearnin Java, Jonathan Knudsen i Patrick Niemeyer,0-596-00873-2,May 2001.

[6]JavaScript:
David Flanagan,2006, 978-0-59-610199-2,August 2006,JavaScript: The Definitive Guide

[7]Struts.

Mike Robinson i Ellen Finkelstein, 2004, 0-7645-5957-5, 2004, Jakarta Struts for dummies.

[8] Memoria final de carrera:Pràcticas, Sesiones y Grupos, Carlos Jimenez Leal, Septembre, 2006.

Index de Figures.

Resum
Actualment, el servei d'informàtica de l'escola d'enginyeries (SIEE) s'enfronta a dos problemes:l'augment del nombre d'alumnes, mantenint el mateix número d'ordinadors per fer les pràctiques i, d'altre banda, el també creixent nombre d'aplicacions que s'han desenvolupat i es desenvolupen per resoldre les necessitats generades pels mateixos alumnes. Aquest projecte neix amb la voluntat de solucionar aquests problemes, creant per un costat un aula de màquines virtuals i per altre banda crear un aplicatiu web, que servirà de framework i contenidor de futures aplicacions, on es pugui connectar de manera senzilla amb les màquines virtuals.

Resumen
Actualmente, el servicio de informàtica de la escuela de ingenierias (SIEE) se enfrenta a dos problemas: el aumento del número de alumnos, manteniendo el mismo número de ordenadores para hacer las pràcticas y, por otro lado, el también creciente número de aplicaciones que se han desarrollado y se desarrollan para resolver las necesidades generadas por estos mismos alumnos. Este proyecto nace con la voluntad de solucionar estos problemas, creando por un lado un aula de máquinas virtuales y por otra parte crear una aplicación web, que servirá de framework y contenedor de futuras aplicaciones, donde se pueda conectar de manera sencilla con las máquinas virtuales.

Abstract
Now, the computer service of the engineering school (SIEE) faced two problems: the increase of the number of the students keeping the same number of computers to do the practise and, for the other side, the increase of the number of applications developed and that are developing to resolve the needs generated by the students. This project born with the wish of solve this problems, make a classroom of virtuals machines and for other side make a web application, that use it of framework and container of future application, where the student can connect in a simply way with the virtual machines.