EDUCAWEB.COM
MILLORA DEL RENDIMENT DE LA PLATAFORMA

ADMINISTRACIÓ DE XARXES I SISTEMES OPERATIUS

Autor: Xavier Aspas Bayón
Consultor: Miguel Martín Mateo
Tutor: Jordi Sabaté Estopà

23 de Desembre de 2012

Resum del projecte

L'empresa Educaonline gestiona sota la marca Educaweb portals de continguts d'orientació acadèmica i professional a Catalunya, Espanya, Itàlia, Mèxic i Brasil, i preveu seguir creixent obrint serveis a nous països.

Actualment tots aquests serveis són servits des d'una plataforma situada a la ciutat de Barcelona, a on aquesta presenta una bona connectivitat amb Europa oferint temps de resposta òptims però s'ha detectat que els usuaris americans tenen uns temps de resposta molt més dolents. D'altra banda amb el creixement de visitants i nous portals les necessitats de processament continuen augmentant i posen més estrès als servidors d'aplicacions.

Així doncs el projecte pretén millorar els temps de resposta del servidors i reduir les necessitats de processament dels mateixos per tal que siguin més eficients i ràpids entregant les peticions.

Un estudi detallat de les pàgines servides als visitants revela un elevat nombre de repeticions en les pàgines consultades, a tall d'exemple un dia *normal* a un dels portals es serveixen al voltant de 40.000 pàgines de les que només n'hi ha 15.000 que siguin diferents, per tant tota la resta de peticions podria reaprofitar-se per evitar estressar els servidors web.

El temps de càrrega d'una plana es divideix en el temps de processament i el temps de transferència, per tant que aquest projecte intentarà reduir ambdós temps.

Per a la reducció del temps de processament es planteja la instal·lació d'un HTTP Proxy Cache, en concret el programari lliure Varnish Cache que permet aprofitar crides passades i servir-les directament sense haver-les de tornar a processar.

Pel que fa a la reducció del temps de transferència l'objectiu és col·locar aquests Caches a servidors propers al públic objectiu de cada servei. Per exemple desplegant un node als Estats Units per atendre els usuaris de americans.

El producte resultat d'aquest projecte ha de ser un servidor virtual amb el Varnish Cache instal·lat i escoltant les peticions, retornant les respostes si les té o demanant-les al node central en cas de que no les tingui i guardant una còpia per a la propera petició repetida.

Índex

1. Introducció i Objectius

L'empresa Educaonline desenvolupa i gestiona el conjunt de pàgines web i portals que actuen sota la marca Educaweb. L'activitat principal d'Educaonline és l'orientació acadèmica i professional, tant presencial com online. La seva activitat comença l'any 1994 però no introdueix el serveis online fins l'any 1998 quan crea Educaweb.com, un portal web dirigit a recollir tots els materials d'orientació disponibles per a l'activitat presencial i posar-los a disposició dels usuaris mitjançant la xarxa.

A la seva primera versió Educaweb.com era un simple directori de recursos educatius, centres i cursos disponibles a l'estat espanyol, amb l'objectiu de donar suport a l'activitat presencial i donar a conèixer la marca, però sense un objectiu comercial concret.

Amb el pas del temps el portal es va convertir en un punt de referència important pels professionals del sector i va anar assolint un nombre important de visites que va fer creure que s'hi podia treure un rendiment econòmic, al principi en forma de publicitat inserida entre els continguts totalment gratuïts que oferia el portal.

Ben aviat es va fer evident que el manteniment del portal era insostenible sense un model de negoci clar ja que els visitants anaven augmentant i calia mantenir el contingut actualitzat i els recursos econòmics obtinguts amb el model publicitari eren insuficients.

La solució va arribar de la mà d'un nou producte, la generació de contactes pels centres educatius o més comunament coneguts com a cupons o *leads* en anglès. Els centres educatius tenien la seva oferta de cursos publicada al portal i els usuaris que hi accedien podien sol·licitar informació al centre directament des del mateix. El model de negoci consistia en rebre una comissió per a cada potencial estudiant enviat al centre, aquest és el model que encara avui suposa la principal font d'ingressos de l'activitat del portal.

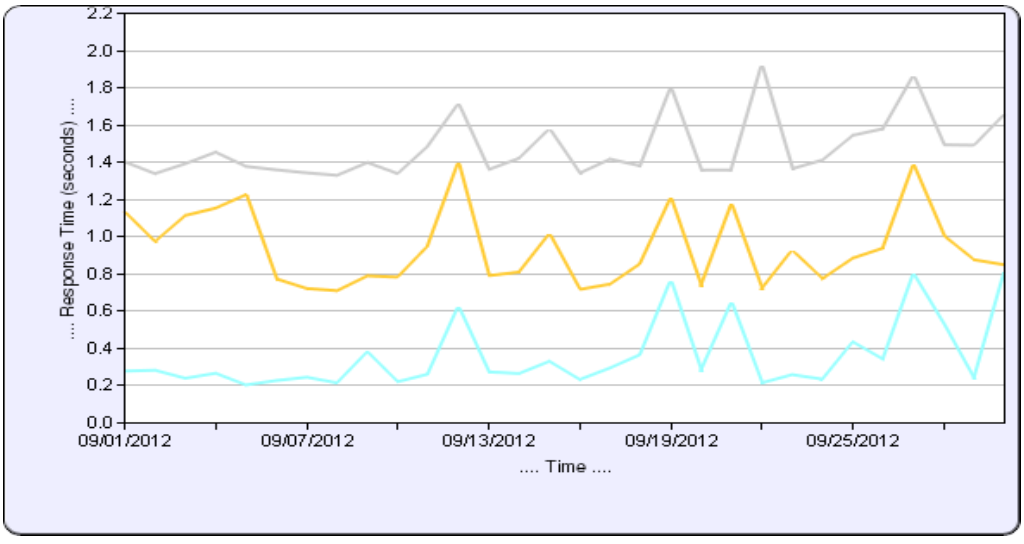
Aquest és un model sostenible si el volum de centres, cursos i visitants és elevat ja que els ingressos per sol·licitud són relativament baixos. Per això l'empresa va decidir crear nous continguts i serveis, sempre gratuïts per l' usuari visitant, que incrementessin la captació de nous visitants cap al portal, principalment via motors de cerca. Un exemple d'aquests serveis són nous webs específics d'orientació com Qestudio, tot un conjunt de *newsletters* d'informació i orientació periòdics, així como el test d'interessos professionals GR, i la versió catalana Educaweb.cat.

Gràcies a aquesta estratègia es va aconseguir un augment significatiu dels visitants i avui en dia el conjunt de productes Educaweb atrau al voltant de 6 milions d'usuaris anuals al seus portals.

L'evolució següent és clara, la internacionalització del model, i des de 2010 s'aposta decididament per la replicació de portals internacionals utilitzant el mateix model d'Educaweb.com, la captació de visitants, centres i cursos. Actualment s'han creat les versions per Mèxic (Educaweb.mx), Brasil (Educaweb.br.com) i Itàlia (Educaweb.it) i hi ha diversos països en fase de planificació i preparació.

Degut al creixement en nombre de productes i serveis i a l'increment de visitants ha estat necessari adaptar i millorar la infraestructura per donar un servei adequat als usuaris cosa que s'ha anat produint periòdicament.

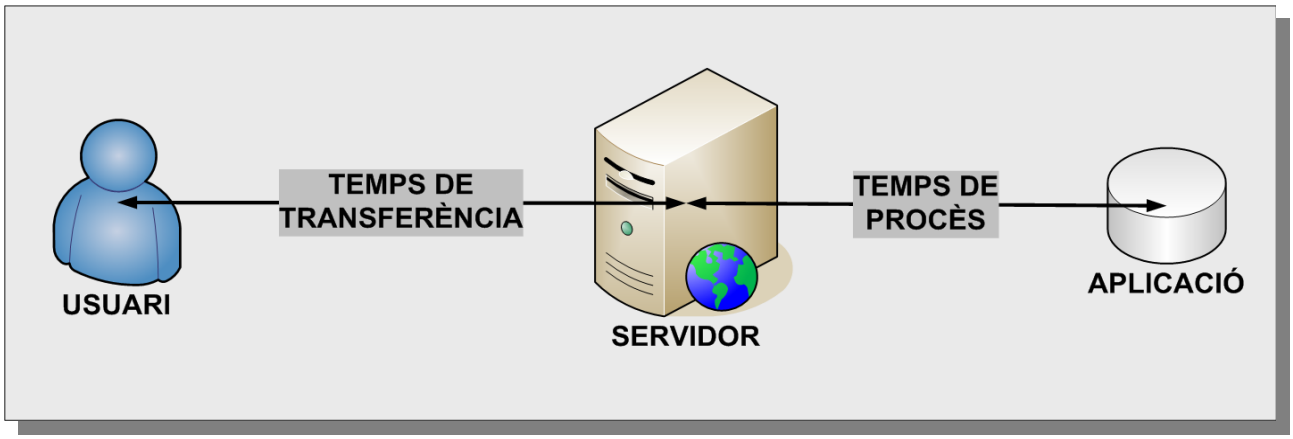
Actualment tots aquests serveis són servits des d'una plataforma situada a la ciutat de Barcelona, a on aquesta presenta una bona connectivitat amb l'estat Espanyol oferint temps de resposta òptims pels visitants de l'estat i Europa, però s'ha detectat que els usuaris americans tenen uns temps de resposta molt més dolents tal i com es pot veure, a través de l'eina de monitorització Alertsite[1], el següent gràfic mostra el temps en segons en rebre el document HTML de la pàgina principal del website www.educaweb.com.



Estudis realitzats per les empreses més importants del sector, com Amazon, Google i Microsoft conclouen que el temps de resposta és crític per una optimització de la satisfacció de l'usuari i el rendiment econòmic del servei. A tall d'exemple, Amazon va comprovar que cada 100ms addicionals de temps d'espera en la càrrega de la pàgina reduïen les vendes en un 1%.[2]

1 Website Performance Monitoring – AlertSite (http://smartbear.com/products/web-monitoring/website-monitoring)
2 The Psychology of Web Performance (http://www.websiteoptimization.com/speed/tweak/psychology-web-performance)

Els elements principals del temps de resposta d'una pàgina són el temps de processament i el temps de transferència de la pàgina com es pot veure al següent gràfic.



El temps de transferència depèn de la **qualitat de les comunicacions** i de la ***distancia*** entre l'usuari i el servidor en nombre de salts que ha de realitzar la comunicació.

El temps de proces depèn de la **càrrega del servidor** i de la **qualitat de l'aplicació**.

Els objectius del projecte són:

- Acostar geogràficament els continguts dels serveis als usuaris internacionals per millorar la distancia entre usuari i servidor.
- Minimitzar la càrrega del servidor dinàmic.

2. Estudi de viabilitat

2.1 Necessitats i requeriments del client

Durant els 2 darrers anys Educaweb ha fet el salt internacional oferint serveis a països llatinoamericans, com Mèxic i Brasil. El fet d'oferir serveis a usuaris d'aquests països ha posat en evidència la pobre connectivitat entre els servidors localitzats a Barcelona i els usuaris que visiten la web des d'aquests països a on la seva experiència d'usuari és més pobre que les dels visitants de l'estat espanyol.

D'altra banda la proliferació de portals ha significat un augment significatiu en les necessitats d'emmagatzemament i processament de la plataforma actual que podria fins i tot penalitzar els usuaris més propers.

El client desitja millorar el rendiment general del sistema per suportar l'augment de visitants i a l'hora acostar físicament el servei als nous visitants del continent americà per tal de que la seva experiència sigui el més semblant a la que tenen els actuals visitants espanyols.

Degut al dinamisme del negoci i la poca disponibilitat en hores de programació que no siguin per la creació de noves funcionalitats o millora de les actuals un requisit molt important imposat pel client és que l'afectació de la solució proposada s'adapti el màxim possible a l'estructura actual afectant el mínim possible a l'aplicació en funcionament. En una situació ideal el sistema s'ha de pensar com una capa per sobre de l'actual que sigui transparent per les aplicacions actualment en funcionament.

A més cal tenir molt present que ens trobem davant d'un servei que funciona 24x7 els 365 dies de l'any i que no es pot permetre una aturada per tant caldrà posar-ho en producció de forma transparent pels usuaris sense deixar de donar servei en cap moment.

2.2 Anàlisi de la situació actual

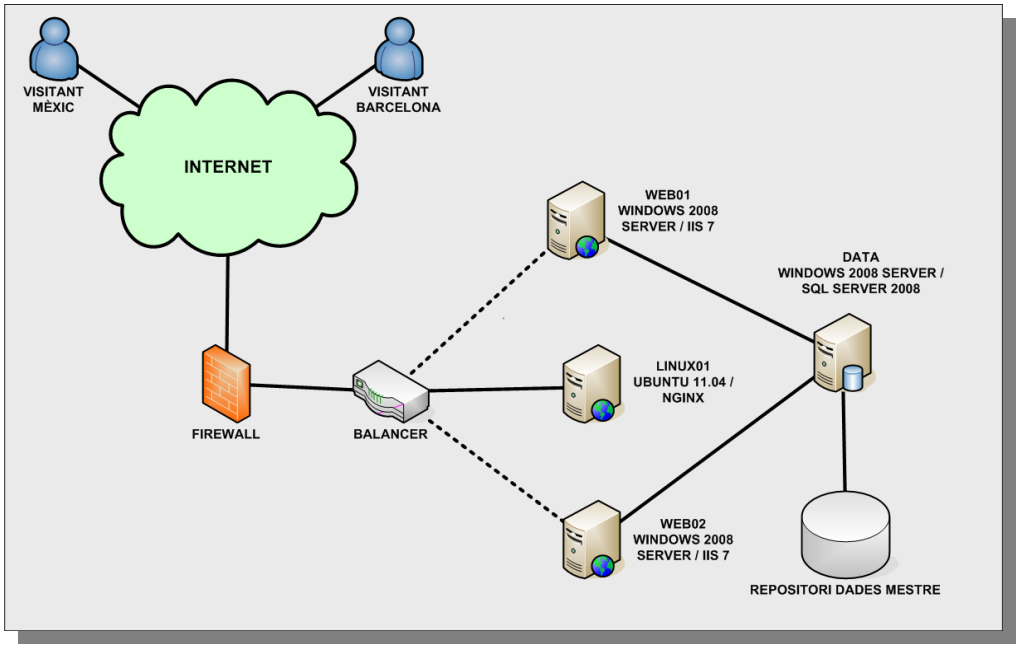
Plataforma Tecnològica

Actualment la plataforma tecnològica d'Educaweb té la següent estructura:

- Situada a un CPD a la ciutat de Barcelona.
- Connectivitat síncrona de 6Mbps
- Balancejador de peticions HTTP.
- Firewall.
- Servidors físics amb software de virtualització VMWare
- Servidors virtuals:

    - **DATA** : Servidor de Base de Dades Windows 2008 Server / SQL Server 2008 Standard Edition
    - **WEB01**: Servidor Web pel contingut dinàmic Windows 2008 Server / IIS 7.5
    - **WEB02**: Servidor Web pel contingut dinàmic Windows 2008 Server / IIS 7.5
    - **LINUX01**: Servidors Web pel contingut estàtic Linux Ubuntu 11.04 LTS / NGINX

El següent diagrama mostra l'estructura anterior:



Aplicació

Les aplicacions d'Educaweb estan desenvolupades en C# amb el Framework ASP.NET 4.0. pel que fa al contingut dinàmic i una plataforma basada en servidors Linux i servidor web NGINX[3] per a servir contingut estàtic.

El repositori de fitxers principal es troba situat al Servidor DATA i es replica en una sola direcció a la resta de servidors via el programa Mirror Folder.[4]

Temps de resposta

Els temps de resposta actuals mesurats amb l'eina de monitorització Alersite per carregar el document HTML de la pàgina principal són els següents :

Response Time and Throughput Breakdown							
Location	**Throughput Kbytes/sec**	**Total Response Time**	**DNS Time**	**Connect Time**	**Redirect Time**	**First Byte**	**Content Download**
Madrid, ES	633	0.3467	0.0074	0.0194	0.0000	0.2800	0.0399
Sao Paulo, BR	35	1.4813	0.0287	0.2395	0.0000	0.4967	0.7163
New York, US	71	0.8551	0.0219	0.1289	0.0000	0.3490	0.3554
Total / Avg	**247**	**0.8943**	**0.0193**	**0.1293**	**0.0000**	**0.3752**	**0.3705**

3 NGINX (http://nginx.org/)
4 MirrorFolder: A real-time folder synchronization and backup software (http://www.techsoftpl.com/backup/)

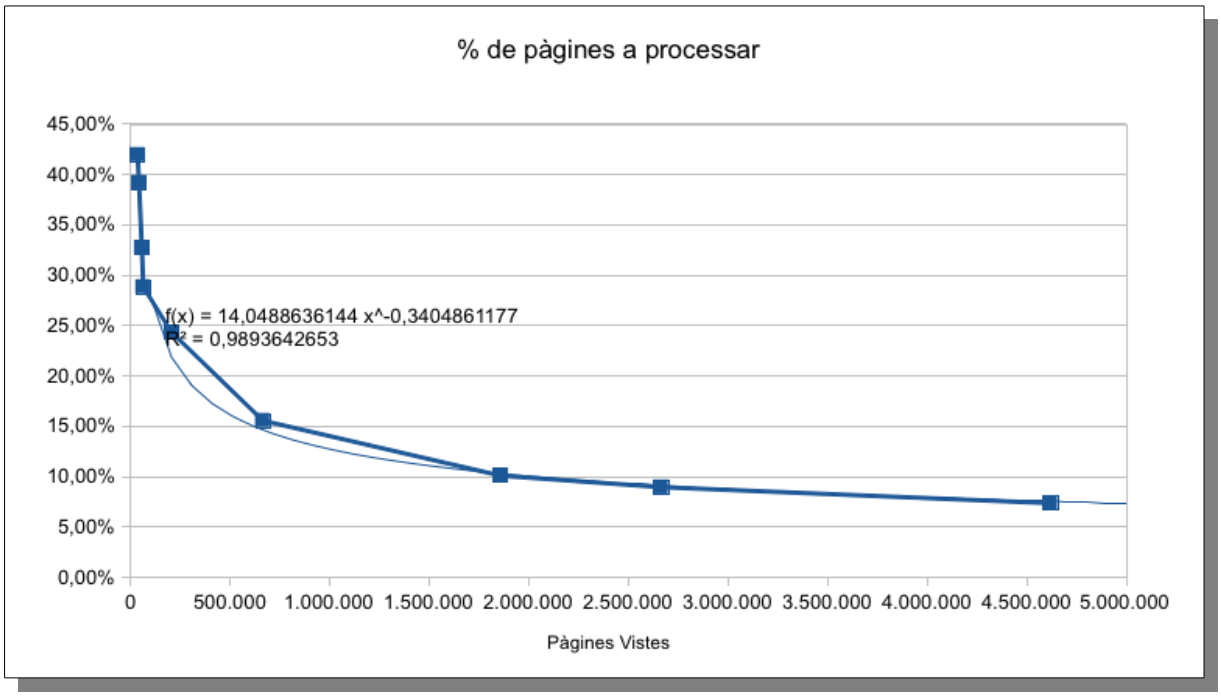
Càrrega de processament de l'aplicació

Actualment el portal Educaweb.com està servint una mitjana de 40.000 pàgines diàries als usuaris segons indica Google Analytics, a això cal sumar-li totes les pàgines que són servides als motors de cerca com Google Bot i altres. Cada plana d'aquestes suposa una crida a l'aplicació amb la conseqüent càrrega de procés per l'accés a la base de dades i el render de la pàgina.

A la següent taula tenim diversos exemples de períodes diversos de pàgines que es serveixen comparat amb les pàgines diferents que són servides, per exemple un dia de baixa audiència es serveixen 17.992 pàgines però realment només 8.219 són diferents.

Pàgines	Pàgines Úniques	%
17.992	8.219	45,68%
33.733	14.159	41,97%
40.000	15.684	39,21%
54.843	17.962	32,75%
61.522	17.746	28,84%
204.027	49.657	24,34%
663.002	103.208	15,57%
1.857.409	187.685	10,10%
2.664.334	238.860	8,97%
4.619.921	340.207	7,36%

Si agafem diversos períodes diaris, setmanals, mensuals, trimestrals, etc., veiem que quantes més pàgines servides major repetició en les pàgines, en concret segueixin una funció com la que veiem al gràfic següent a on s'observa que quan s'incrementa el número de pàgines augmenta la repetició, cosa que ens porta a la conclusió que tot aquest temps de procés està desaprofitat per a les planes reaprofitables que són moltes com es veurà més endavant.



2.3 Definició de requeriments del sistema

El sistema a implementar ha de complir els següents requeriments, entre parèntesis valorem la importància del requeriment:

1. **Reduir el temps de càrrega de les pàgines. (10)**

   Cal millorar l'experiència de l'usuari millorant el temps en que triga un usuari a veure la pàgina completament carregada des del moment de sol·licitud. El que anomenem *Total Response Time* a la taula anterior.

2. **Minimitzar les necessitats de processament de l'aplicació per permetre l'increment d'usuaris. (10)**

   Segons les estadístiques extretes amb l'eina Google Analytics i vistes en el quadre de % de repetició de pàgines un dia tipus es podria reduir al 40% el nombre de pàgines a processar degut a la repetició de crides a pàgines dinàmiques recurrents durant un mateix dia.

3. **Que sigui una solució fàcilment escalable. (8)**

   La solució ha de permetre créixer en nombre de serveis i usuaris amb facilitat.

4. **Que sigui transparent a l'aplicació de contingut dinàmic. (8)**

   És considera molt important que no calgui gaire desenvolupament específic a l'aplicació de contingut dinàmic, per tant caldrà buscar una solució tant transparent com sigui possible pels programadors.

5. **La solució ha de posar-se en producció sense aturades de servei pels usuaris actuals. (7)**

   Al tractar-se d'un servei 24x7x365 hem d'aconseguir muntar la solució amb el mínim d'interrupció del servei, si és possible no hauria d'haver-hi cap interrupció i caldrà valorar el cost de la mateixa en cas de ser imprescindible.

6. **Que el seu manteniment sigui escalable. (7)**

   És molt important que el creixement de la solució no suposi grans costos ni complexitat en el seu manteniment.

7. **Mantenir els costos el més acotats possibles, utilitzant eines de programari lliure sempre que sigui possible. (10)**

   Degut a que es preveu un creixement important en nombre de serveis durant els propers anys la solució hauria de tenir uns costos el més reduïts possibles, idealment una ampliació de serveis no hauria de suposar un cost linealment proporcional al cost de la primera instal·lació i hauria d'anar-se reduint a mesura que augmenta la seva dimensió.

8. **Que compleixi amb la legislació vigent sense gaires dificultats. (6)**

   El fet que ens trobem en un entorn internacional pot suposar problemes d'incompatibilitats legals especialment pel que fa a la Llei de Protecció de Dades, caldrà tenir en compte aquest requeriment ja que pot suposar costos legals ocults si no optem per una solució general que eviti la problemàtica legal.

2.4 Estudi d'alternatives de solució

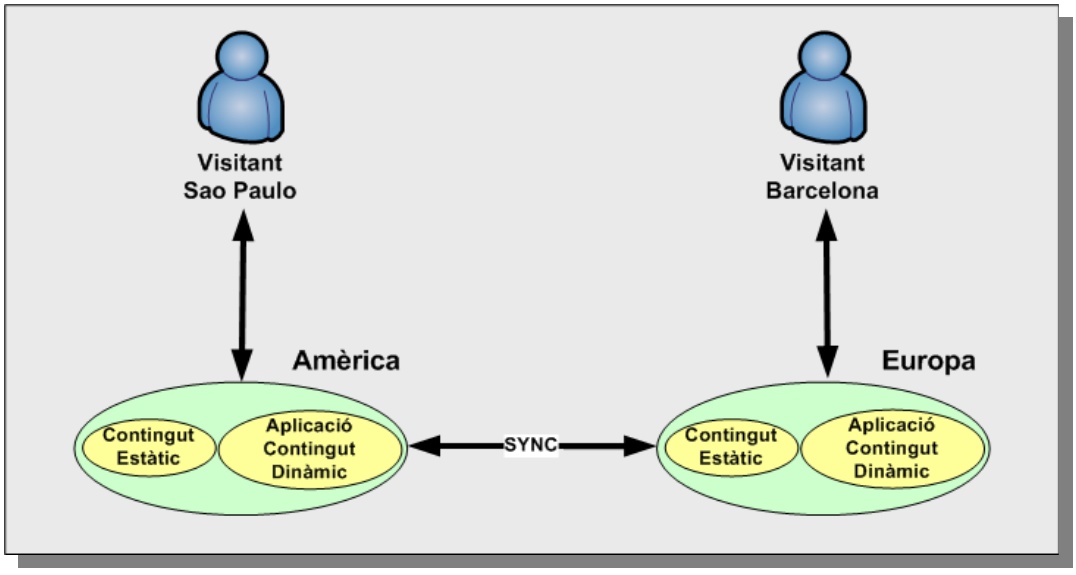
Tal i com hem vist la distancia entre els visitants i servidor web és un factor clau en el temps de resposta del servidor i la càrrega de les pàgines. Per això estudiarem alternatives que permetin l'acostament del servidor web als usuaris i la disminució de la càrrega de procés del(s) servidor(s) de contingut dinàmic.

Nodes Remots Actius (Alternativa 1)

Creació de Nodes Remots Actius amb replicació de la plataforma completa: sincronització de fitxers estàtics, aplicació localitzada al node remot i replicació de la base de dades.

> **Avantatges:** Ofereix la capacitat de servir el contingut d'altres nodes si un d'ells cau ja que ens permet replicar el node de contingut dinàmic i la base de dades. El fet de dividir ens diversos nodes les necessitats de procés disminuirien les necessitats de procés del node actual, millorant el temps de procés.
>
> **Inconvenients:** Aquesta alternativa suposa uns costos elevats ja que implica la replicació completa de tota la plataforma, per tant els *hostings* remots hauran de ser potents i cars, a més tenint en compte que la solució actual està desenvolupada en tecnologies propietàries caldria adquirir més llicències i algunes de les actuals no serien vàlides ja que per implementar un Clúster de Bases de Dades amb SQL Server cal disposar de la versions Enterprise molt més costoses. El fet de replicar la base de dades també té implicacions legals ja que estem realitzant una transferència internacional de dades personals.

El gràfic següent mostra l'esquema de la solució:

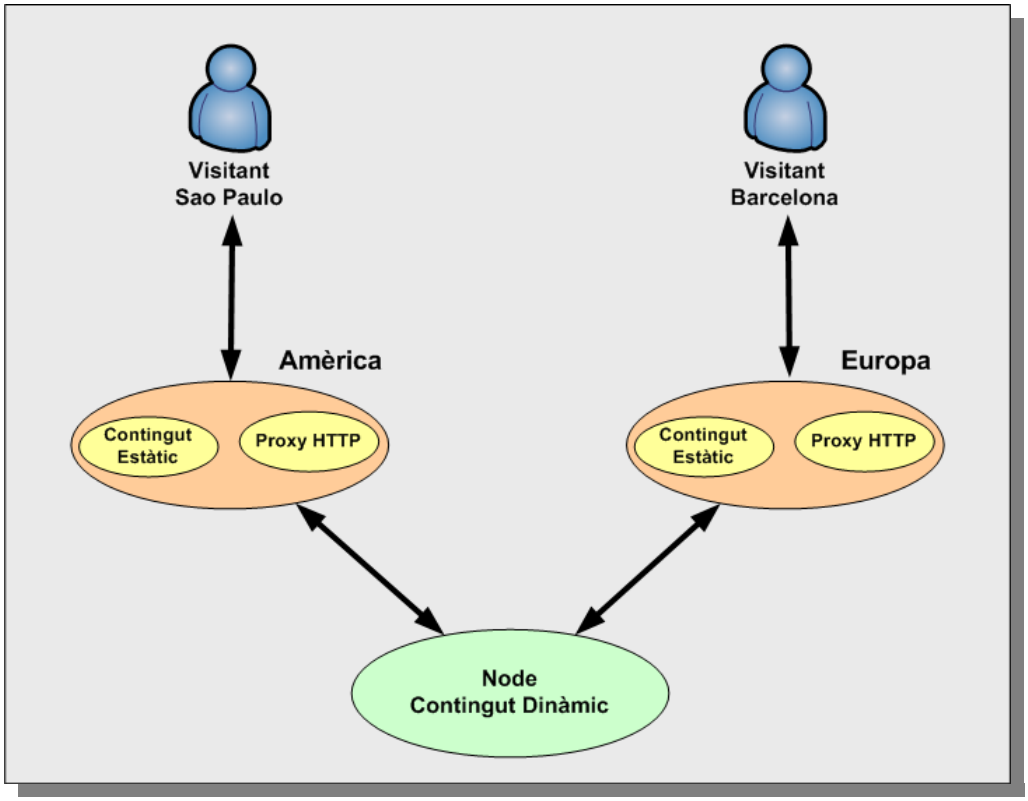


Nodes Externs Lleugers (Alternativa 2)

Creació Nodes Externs que situarem a una distancia propera dels usuaris amb una replicada del contingut estàtic i un sistema que permeti la recuperació eficient del contingut dinàmic mitjançant un HTTP Cache.

> **Avantatges:** Els nodes remots són molt més lleugers que el node dinàmics i les necessitats de *hosting* seran molt menors. La implementació pot ser molt transparent a l'aplicació de contingut dinàmic actual. Hi ha solucions de programari lliure sense cost de les llicències. El contingut en Cache no tornarà a processar-se pel que alliberarem el node mestre de càrrega de procés. Des d'un punt de vista legal no transferim dades de caràcter personal i ens permet limitar-nos a la legislació espanyola.
>
> **Inconvenients:** Seguim disposant d'únic node dinàmic i per tant un únic punt de fallada per tot el sistema però aquest és un problema que actualment ja tenim.

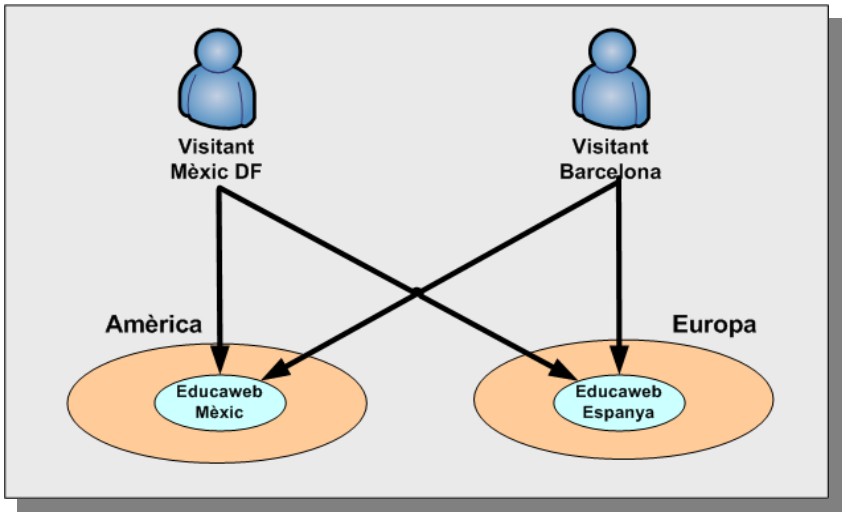
L'esquema següent mostra l'esquema de la solució:



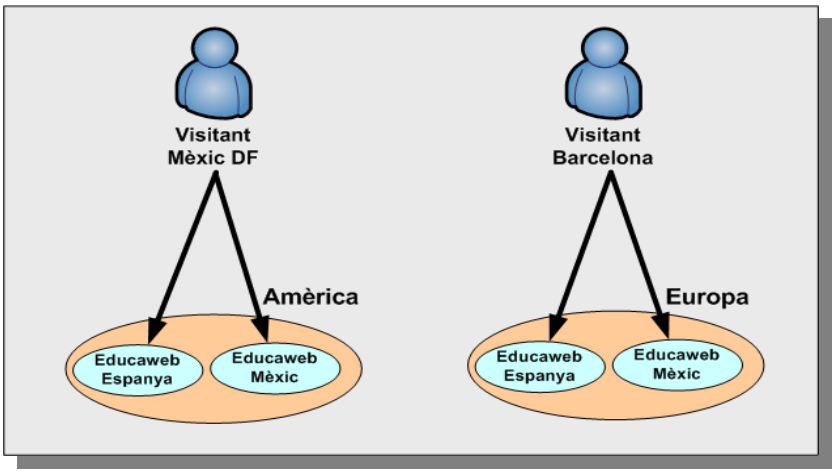
Consideracions addicionals

En ambdues alternatives caldrà definir si volem orientar la solució al servei o l'usuari que visita la web.

Orientada al servei: Cada node servei només els continguts del seu àmbit geogràfic i el fet d'accedir a un node o altre no depèn de la localització del visitant si no del servei que està accedint. Així doncs un usuari mexicà que volgués accedir a un contingut d'Educaweb.com (Espanya) acabaria accedint als servidors localitzats a Barcelona i només es beneficiaria de la proximitat del servei quan volgués accedir a Educaweb.mx que es trobaria a una ubicació més propera.



Orientada a la ubicació del usuari: Cada node serveix tots els continguts per a tots els serveis i un sistema de GeoDNS s'ha d'encarregar d'enviar les peticions de l'usuari al servidor adequat en funció de la seva localització. En aquest cas un usuari mexicà que volgués visitar Educaweb.com ho faria contra un servidor localitzat més a prop, ja que el sistema de GeoDNS li serviria la IP del servidor més proper.



2.5 Valoració i elecció de les possibles solucions

A la taula següent veiem una comparativa de l'acompliment dels requeriments del projecte segons les alternatives proposades.

Requeriment	Nodes Actius (Alternativa 1)	Nodes Lleugers (Alternativa 2)	Orientació a Servei	Orientada a Ubicació
1. Temps de resposta	Millorar degut a la major proximitat entre el visitant i el node.		Els usuaris creuats no aprofiten la proximitat dels nodes.	Tots els usuaris aprofiten la proximitat dels nodes.
2. Càrrega servidor dinàmic	Es redueix degut a la distribució de peticions entre els diferents nodes dinàmics.	Es redueix degut a l'aprofitament de les peticions en Cache.	Menor degut a la major utilització del Cache	Major degut a la menor utilització del Cache
3. Escalabilitat	Alta però augmentant la complexitat del sistema degut a l'augment de nodes al clúster de base de dades.	Alta ja que la proliferació de nodes es tan fàcil com desplegar una nova màquina virtual en un nou punt de la xarxa.	Menor ja que l'augment d'usuaris recau sobre un sol servidor	Major ja que l'augment d'usuaris pot distribuir-se segons la seva ubicació
4. Transparència	Baixa ja que s'hauran de tenir en compte els problemes típics dels sistemes distribuïts.	Cal fer certs retocs a les aplicacions per fer que més continguts siguin *cachejables* i els programadors hauran de tenir-ho present sempre.	Menor ja que els programadors han de tenir present on es publicarà el contingut i treballar les URLs.	Major, tots els nodes són iguals però té implicacions de sistemes a la configuració del GeoDNS.
5. Disponibilitat	Alta ja que ofereix un sistema de replicació de tot el sistema que pot ser utilitzat en cas de caiguda d'algun node dinàmic.	La mateixa que actualment amb certa millora que poden oferir els sistemes de Cache quan segueixen servint contingut encara que el node dinàmic estigui caigut si el tenen a la seva memòria.	Menor ja que si falla el node del servei tots els visitants deixaran de tenir-hi accés.	Major ja que en cas de fallida d'un node tota la resta de visitants d'altres nodes segueixen tenint accés.
6. Manteniment	Complex degut a la replicació de tota la plataforma especialment de la base de dades.	Baix ja que la solució la conformen màquines virtuals totalment idèntiques entre nodes.	Les configuracions del nodes són lleugerament diferents.	Cal mantenir el GeoDNS.
7. Costos	Alts degut a plataformes més potents i les llicències de programari propietari.	Baixos degut a plataformes menys potents i ús de programari lliure.	Menors ja que la potencia dels serveis de hosting no cal que sigui tan elevada.	Majors pel servei de hosting i els serveis de GeoDNS.
8. Legals	Hi ha transferència de dades personals.	No hi ha transferència de dades personals.	No té implicacions legals.	No té implicacions legals.

Compleix Compleix parcialment No compleix

Solució a implementar

De les anàlisis anteriors es conclou que la solució de **Nodes Externs Lleugers (Alternativa 2)** és la que més s' adapta als requeriments especificats ja que compleix total o parcialment tots els requeriments mentre que l'alternativa 1 incompleix 4 dels requeriments del projecte.

Pel que fa a l'orientació a servei o ubicació la decisió no és tant evident ja que si bé sembla que l'orientació a ubicació ens ofereix un millor encaix amb els requeriments la diferència no es substancial i cal que tinguem en compte la configuració i gestió del servei de GeoDNS. El sistema que es vol implementar tenen un nivell de complexitat força elevat per afegir-li un nou repte d'implementar un sistema de GeoDNS, i tenint en compte que es fàcilment implementable a posteriori s'ha decidit que en aquest projecte s'implementarà la **solució orientada a servei**.

Per tal de reduir els costos al màxim i donat que l'estructura actual ja disposa d'un servidor de contingut estàtic l'objectiu del projecte serà construir un Node Extern a partir del servidor de contingut estàtic.

Programari

La solució final del projecte constarà d'una màquina virtual que ha de ser fàcilment replicable cada cop que es vulgui desplegar un nou Node Lleuger i que tindrà instal·lat el següent programari:

Sistema Operatiu: El sistema operatiu dels Nodes Externs queda definit pel sistema actualment en funcionament que utilitza **Ubuntu Server 12.04 LTS**[5].

Servidor Web: A l'igual que amb el sistema operatiu es conservarà **NGINX** pel contingut estàtic, un servidor web de programari lliure molt lleuger optimitzat per servir contingut estàtic molt eficientment.

Programari de sincronització: Actualment el sistema utilitza per a la sincronització d'arxius el programari Mirror Folder**,** aquest ha demostrat ésser força eficient en la sincronització d'arxius entre ordinadors connectats a la mateixa xarxa local però no sembla molt eficient per sincronitzar arxius a través de la xarxa. Una alternativa de Programari lliure alternatiu podria ser **RSYNC**[6] que serà estudiat com a alternativa per aquest projecte.

5 Ubuntu (http://www.ubuntu.com/)
6 RSYNC (http://rsync.samba.org/)

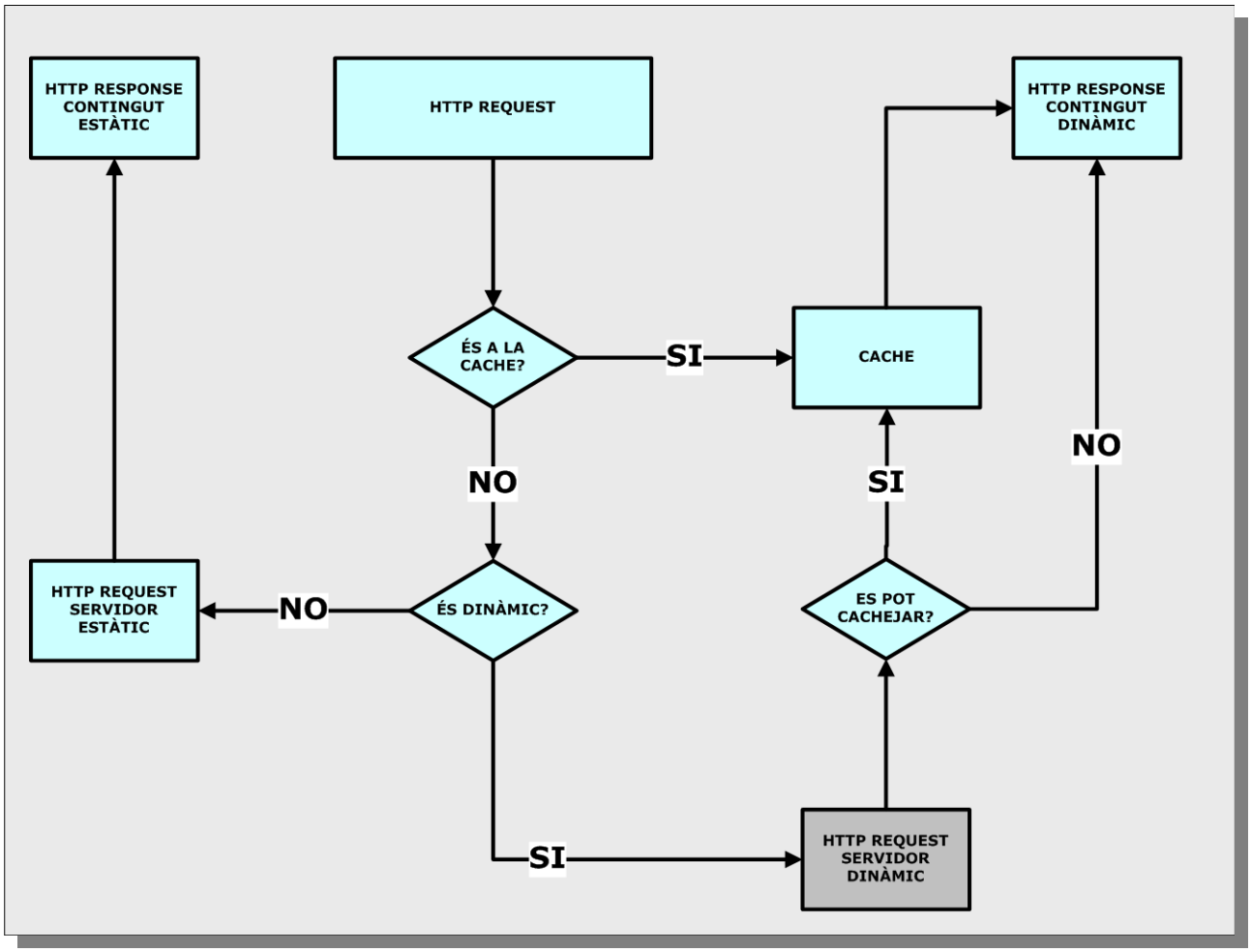
Servidor HTTP Cache: Un servidor HTTP Cache és un programari que permet situar-se com a frontal d'una aplicació web escoltant les peticions i sol·licitar-les al servidor web d'aplicació si no disposa d'una còpia vàlida de la pàgina, en cas de que ja disposi d'aquesta copia vàlida allibera al servidor de contingut dinàmic de la feina d'haver de tornar a generar la pàgina servint la pàgina que ja té a la seva memòria. En aquests moments la millor alternativa de programari lliure per oferir aquesta funcionalitat és **Varnish Cache**[7].

7 Varnish Cache (https://www.varnish-cache.org/)

3. Anàlisi del sistema

3.1 Definició del sistema

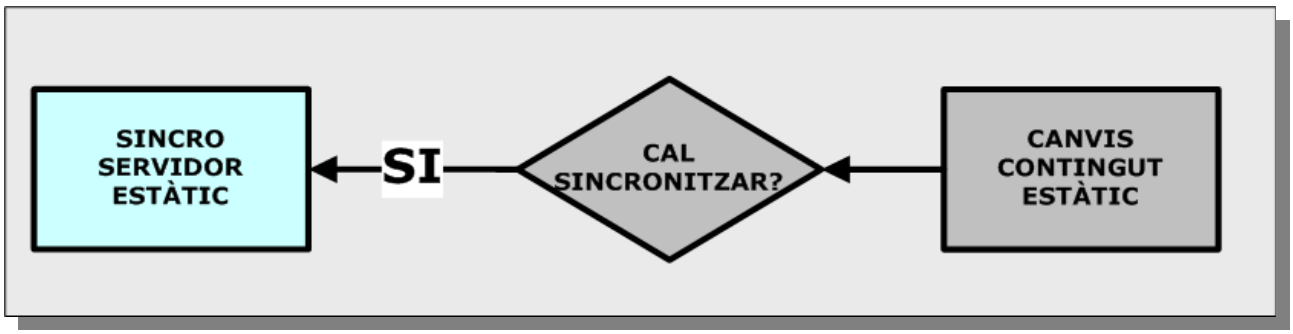
El sistema que volem implementar constarà d'una màquina virtual anomenada Node Extern Lleuger (NEL) que podrà ser posada en funcionament amb un esforç molt petit sempre que es desitgi desplegar una nova ubicació. Aquesta màquina actuarà com a servidor Web de les peticions que li arribin seguint el diagrama següent:



Tots els elements en blau són elements o processos situats al Node Extern Lleuger, en canvi l'element en gris és un servidor extern al que farem peticions.

És interessant observar que al diagrama proposat no afegim els elements estàtics a la memòria cau cosa que també seria viable. Aquesta decisió ha estat presa per minimitzar els requeriments de maquinari dels NELs ja que el programari de Cache que s'utilitzarà treballa en memòria i els elements

estàtics acostumen a ser pesats cosa que augmentaria les necessitats de memòria dels NEL considerablement.
El fet que la memòria cau no tracti els elements estàtics provoca que haguem de sincronitzar els mateixos en temps reals amb el repositori del Node Mestre. Això serà relativament fàcil d'aconseguir gràcies a que la replicació serà unidireccional, del node mestre als nodes lleugers i a que la freqüència d'actualització del contingut estàtic és força baixa cosa que permetrà disposar d'un procés prou eficient de replicació en temps real. El diagrama actual mostra el procés de sincronització.



Al repositori de continguts principals caldrà executar un procés que sincronitzi els canvis en temps real sempre que sigui necessari. Aquest procés s'executarà per tants NELs com tinguem al sistema. En cas de que el sistema creixi molt podríem realitzar una sincronització en cascada per alliberar al repositori principal de la feina de sincronització de tots els NELs que posaria en perill el temps real.

3.2 Requeriments exactes del projecte

- **Necessitats de processament.** El servidor de contingut dinàmic ha de processar menys pàgines de les que actualment està processant. Cal reutilitzar tot el contingut dinàmic que sigui possible sense tornar-lo a generar. Com s'ha vist anteriorment és possible **reduir les necessitats de processament de pàgines d'un dia tipus en un 60%**. Per això utilitzarem el programari Varnish HTTP Cache.

- **Millora del temps de transferència.** La distancia entre els visitants i la ubicació física dels serveis és clau pel que fa al temps de transferència necessari per obtenir una pàgina, per tant un requeriment bàsic és la millora del rati de transferència del contingut. Segons les dades recollides és possible **multiplicar per 10 o més la velocitat de transferència** acostant el servei als usuaris contractant serveis d'hospedatge amb bona connectivitat pels usuaris locals.

- **Sincronització.** El contingut estàtic ha de ser **sincronitzat en temps real** entre el repositori principal situat al node mestre i els NEL. El servei de sincronització correrà sobre protocol segur SSH amb usuari sense password que haurà de tenir molt restringits els privilegis.

- **Programació.** Cal que les modificacions a l'aplicació de contingut dinàmic siguin les mínimes possibles però la orientació a servei proposada requerirà **modificacions en les URLs de contingut estàtic** que actualment apunten totes al mateix conjunt de servidors.

  D'altra banda hi ha un conjunt d'estadístiques que s'estan enregistrant al servidor dinàmic en el moment de la generació de la pàgina per evitar aquest problema **caldrà programar un servei web d'estadístiques i un script de client** per continuar-les enregistrant a les pàgines servides des de la memòria cau.

- **Disponibilitat.** Els serveis s'ofereixen 24x7x365 per això cal planificar molt bé la posada en producció per minimitzar els temps d'aturada de servei, idealment **no s'hauria de produir cap aturada del servei** apreciable pels usuaris. Degut a l'orientació a servei que hem decidit caldrà fer un canvi de DNS dels serveis que passaran a servir-se des d'altres ubicacions, per evitar tallades de servei caldrà que el node mestre continuï essent actiu durant la replicació dels DNS.

- **Monitorització.** D'altra banda el fet d'utilitzar memòries cau per servir els continguts pot fer transparents fallades del servidor dinàmic i cal preveure aquesta contingència **monitoritzant els NEL i el servidor dinàmic.**

- **Seguretat.** La infraestructura actual pertany tota a la mateixa xarxa local i hi ha establertes mesures de seguretat com tallafocs que la protegeixen de l'exterior. Amb la introducció dels NEL aquests quedaran fora del perímetre de protecció actual i **caldrà establir els mecanismes de seguretat necessaris** per protegir-los de forma eficient. Només hauran de ser accessibles els serveis web pels usuaris i la interacció de sincronització i recuperació de contingut dinàmic entre els NEL, els servidors de contingut dinàmic i el repositori de contingut estàtic amb l'utilització d'un tallafocs als NEL.

- **Llicències.** Degut a la probable proliferació dels NELs el **sistema s'ha de basar en programari lliure i en llicències tan poc restrictives** en la mesura que sigui possible per evitar un creixement insostenible dels costos.

- **Legals.** La replicació de continguts a servidors de fora de l'estat espanyol pot tenir implicacions legals que s'han d'evitar, per això cal que **el sistema no transfereixi dades personals** i quedin emmagatzemades als NEL.

4. Disseny del sistema

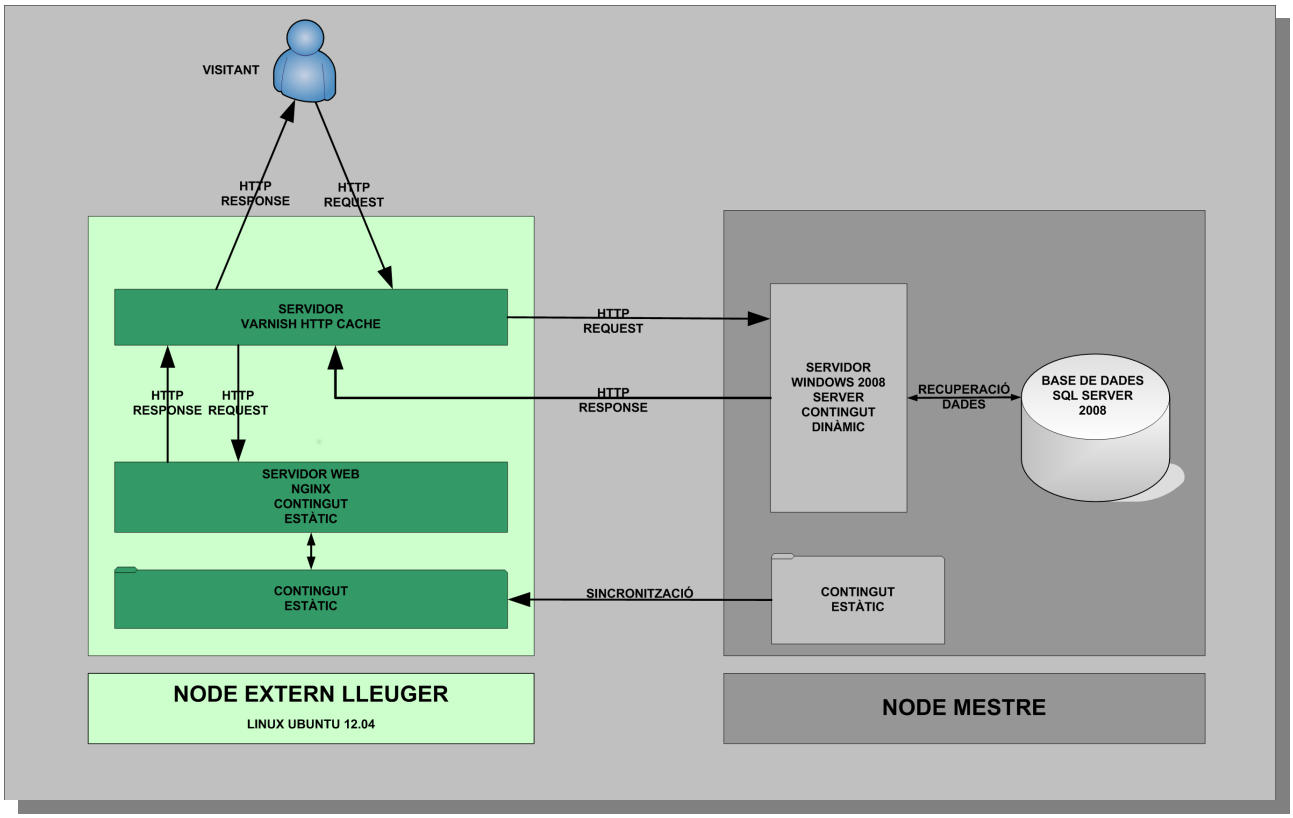
4.1. Arquitectura

Tal i com s'ha indicat en capitols anteriors el sistema constarà bàsicament de 2 elements principals:

El Node Mestre basat en l'arquitectura actual que és l'encarregat de servir el contingut dinàmic i d'emmagatzemar el repositor de contingut estàtic (imatges, fulls d'estil, scripts de client, etc.).

El Node Extern Lleuger que serà un servidor localitzat fora de la xarxa local i sovint en una ubicació remota que contindrà un servidor web pel contingut estàtic i un servidor de HTTP Cache que serà l'encarregat de servir les peticions als visitants.

El següent diagrama mostra els diferents elements de l'estructura de l'arquitectura proposada. En verd es mostren les parts que són objecte de desenvolupament d'aquest projecte



4.2. Subsistemes

4.2.1. Sincronització d'arxius estàtics

Aquest subsistema consistirà en la definició dels elements que interven en el procés de sincronització de fitxes de contingut estàtic entre el node mestre i el node lleuger.

Les característiques principals d'aquest subsistema són:

- **Sincronització unidireccional de creació, modificació i esborrat d'arxius.** Per aquest tasca s'ha triat el programari **RSYNC** inclòs a la distribució de Linux del servidor que actual té el repositori de fitxers estàtics.

- **Temps real**. Utilitzarem el *daemon* **LSYNCD** que controla els esdeveniment de l'API *inotify*[8] de l'arbre de directoris local i executa RSYNC quan cal fer una actualització.

- **Comunicació segura entre nodes.** Per tal de muntar un sistema automàtic i no haver de deixar una porta oberta entre els dos servidors implementarem la comunicació via protocol **SSH** entre els dos nodes. Per això caldrà crear un usuari amb privilegis limitats que permeti l'autentificació sense paraula clau (*passwordless login*).

4.2.2. Frontal Web

Aquest serà el subsistema encarregat d'atendre totes les peticions HTTP i resoldre que cal fer amb elles. Per aquesta tasca hem seleccionat el accelerador **HTTP Varnish Cache**, un programari especificament dissenyat per accelerar el procés de servir elements web. Les tasques principals d'aquest sistema seran:

- **Escoltar els ports web** exposats del servidor i decidir que cal fer amb cada petició.

- **Redirigir les peticions** al servidor adequat quan no li sigui possible processar la petició.

- **Gestionar la memòria cau** de peticions ja resoltes.

- **Retornar la resposta** des de la memòria cau o bé redirigir la resposta dels sistemes interns segons sigui el cas.

8 inotify - monitoring file system events (http://www.kernel.org/doc/man-pages/online/pages/man7/inotify.7.html)

4.2.3. Servidor de contingut estàtic

S'encarregarà de servir el contingut estàtic previament sincronitzat pel sistema de sincronització d'arxius estàtics. Utilitzarem el servidor web **NGINX** que atendrà les peticions que li faci el Frontal Web.

4.2.4. Servidor de contingut dinàmic

Aquest subsistema ja està implementat i és el responsable de crear les respostes de contingut dinàmic, principalment procedent de la base de dades i retornar-les en forma de pàgina html per ser servides. Les principals tasques d'aquest subsistema són:

- **Escoltar els ports definits** pels diferents serveis esperant peticions normalment en format HTTP Request.
- **Enviar les peticions a la capa d'aplicació** que l'hora les enviarà a les capes de dades per obtenir la informació necessària per generar el contingut dinàmic o per realitzar les accions indicades a la petició al servidor de bases de dades.
- **Renderitzar el continguts** en un format apte per ser servit, generalment HTML, tot i que en alguns casos pot tractar-se de documents XML o altres, com per exemple els serveis web.
- **Retornar la resposta** en forma de HTTP Response.

4.2.5. Aplicació de contingut dinàmic

Tot i que aquest sistema no és part del desenvolupament d'aquest projecte l'incloem degut a que és una part clau en tot el procés i serà necessari prendre algunes decisions sobre el seu comportament que poden implicar certes modificacions en l'aplicació per tal de ser més susceptible a ésser utilitzada a la memòria cau.

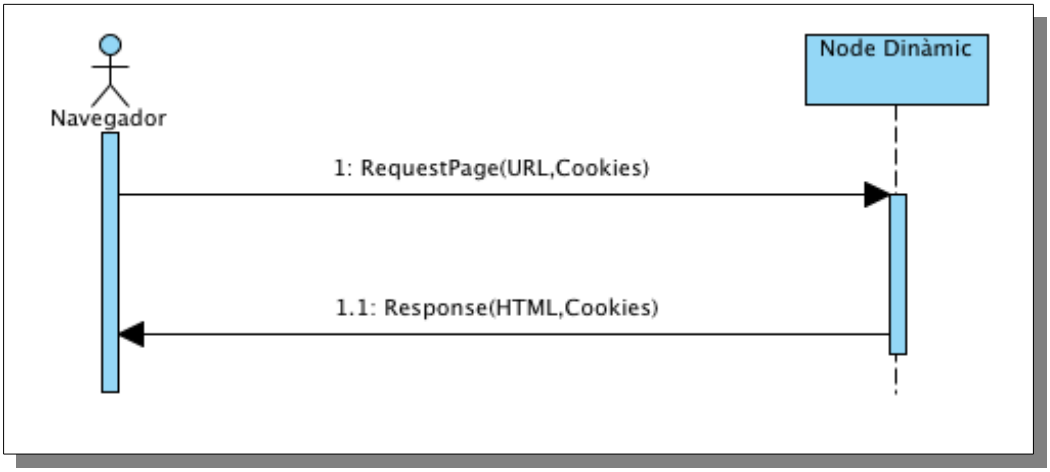
Un dels requeriments del projecte demana que aquesta solució sigui el màxim de transparent amb l'aplicació actual però després d'un acurat anàlisi de la mateixa s'ha arribat a la conclusió que seran necessàries algunes modificacions.

Traçabilitat, estadístiques i registre d'esdeveniments

L'aplicació de contingut dinàmic tal i com està implementada a banda de servir el contingut sol·licitat realitzar tasques de *log* per guardar la traçabilitat per modificar el comportament de certs processos i a l'hora disposar d'informació estadística del comportament dels usuaris. Aquests processos utilitzen

Cookies per tal de tenir la traçabilitat de la sessió, degut a aquesta característica cap de les planes HTML podrien ser emmagatzemades a la memòria ja que si que ho fem perdríem el *log* d'aquestes.

La seqüència actual d'una crida HTTP Request per una pàgina és la següent:



Per tal d'aconseguir que les planes siguin susceptibles a entrar a la memòria cal desvincular el procés de registre d'estadístiques i traçabilitat del procés de generació del contingut.

En concret s'hauran de programar:

- Un script *Javascript* que acompanyarà la plana i s'executarà de forma asíncrona que gestionarà les *Cookies*.

- Un *Webservice* que rebi crides del script anterior amb la informació de l'activitat que s'ha produït per enregistrar-la.

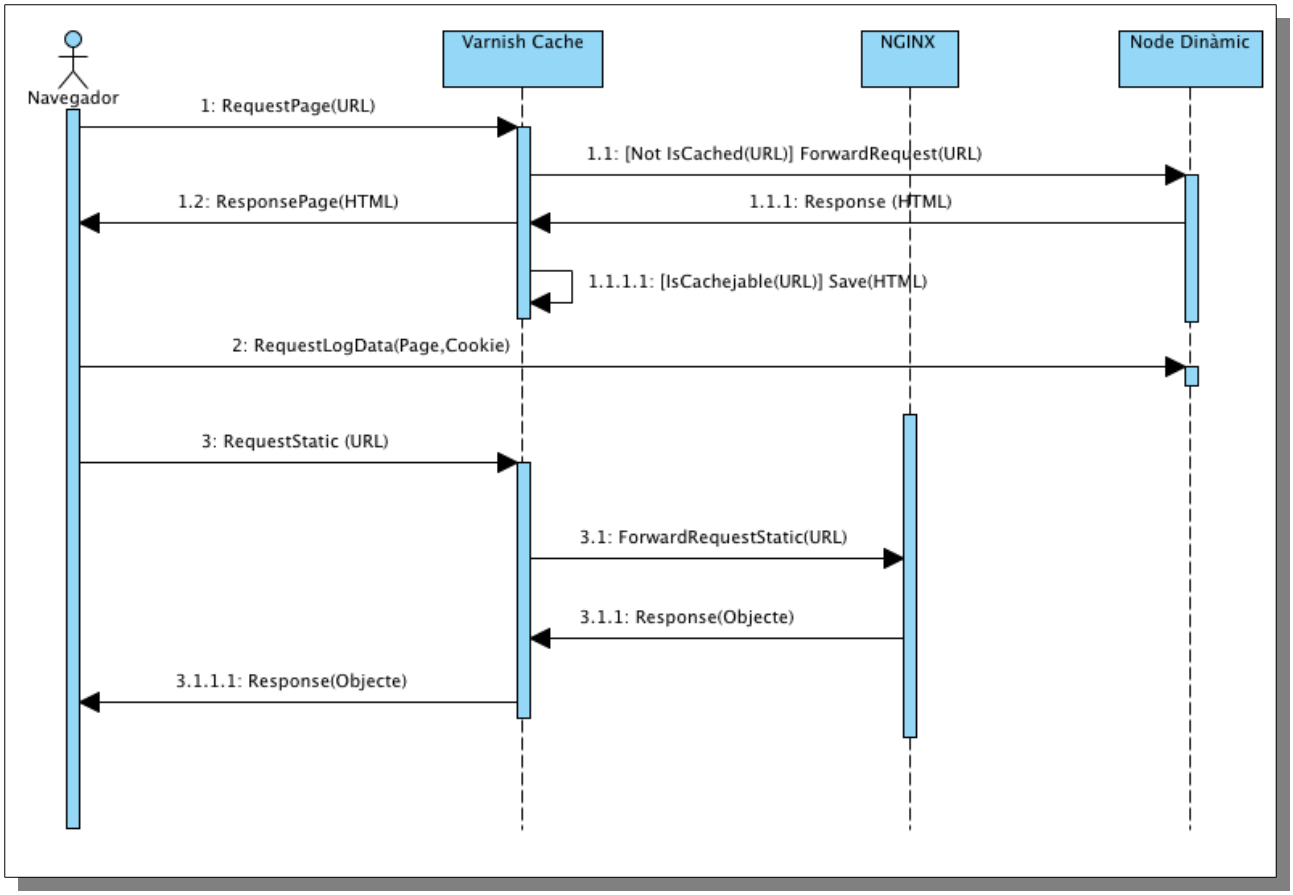
URLs de contingut estàtic

Les planes servides des de l'aplicació de contingut dinàmic incorporen les referències a la ubicació del contingut estàtic. A l'aplicació actual tot el contingut estàtic apunta a un servidor d'estàtics que es troba al node mestre. Caldrà modificar l'aplicació per tal de que les URLs de contingut estàtic apuntin a un domini diferenciat segons el servei que s'estigui visitant.

Per exemple una plana del domini www.educaweb.mx ha de servir el contingut estàtic des del servidor més proper i per això cal que les URLs del mateix apuntin a un domini o subdomini diferent a les URLs del servei www.educaweb.com.

4.3. Cas d'ús

El cas d'ús que tractarem és el de la petició d'una plana web i els elements que l'acompanyen. A continuació mostrem el diagrama de seqüència UML que caldrà considerar.



És important observar que al diagrama anterior avaluem diverses possibilitats.

- HTTP Request d'un contingut dinàmic que no estan a la memòria cau però pot estar-hi.
- HTTP Request d'un contingut dinàmic que no estan a la memòria cau però NO pot estar-hi.
- HTTP Request d'un contingut dinàmic que ja es troba a la memòria cau.
- HTTP Request d'un contingut estàtic.

4.4. Pla de proves

A continuació definim el pla de proves que desenvoluparem per comprovar el correctament funcionament del sistema. L'objectiu d'aquest pla de proves serà la construcció de diversos *scripts* que permetin realitzar una validació automatitzada dels diversos elements que composen el sistema.

4.4.1. Proves unitàries

Sincronització correcta:
Creació d'un conjunt d'arxius al repositori dinàmic i comprovació que aquests siguin sincronitzats correctament i mesura del temps entre l'escriptura i la disponibilitat de l'arxiu al node remot.

Validació de la memòria cau:

- HTTP Request d'una pàgina que no es troba a la memòria cau i que és susceptible de ser-hi, comprovació del processament de la mateixa al servidor dinàmic.
- HTTP Request d'una plana que es troba a la memòria cau i comprovació de que aquesta no s'ha processat al servidor dinàmic.
- HTTP Request d'una plana que no es troba a la memòria cau i que no és susceptible de ser-hi, comprovació de que aquesta s'ha processat al servidor dinàmic.
- HTTP Request d'un element de contingut estàtic i comprovació que s'ha recuperat del servidor local.

4.4.2. Proves d'integració

- Intent d'accés al sistema via SSH amb usuari diferent al usuari de sincronització.
- Intent d'accés al sistema via SSH amb l'usuari de sincronització des d'una IP no autoritzada.
- Intent d'accés al sistema via SSH amb l'usuari de sincronització des d'una IP autoritzada.

4.4.3. Proves de sistema

- Temps de càrrega de les planes abans del desplegament del sistema i posteriorment.
- Número de planes processades abans i després de la posada en marxa de la plataforma.
- Velocitat mitjana de càrrega de les planes.

4.4.4. Proves d'implantació

- Canvis de DNS
- Monitorització de serveis

5. Desenvolupament

Abans de l'inici del desenvolupament s'ha redactat l'informe de requeriments del projecte que hem vist al capítol 3 d'aquesta memòria. Aquest informe ha permès la creació del pla de treball que definirà les tasques de desenvolupament com es mostra a continuació.

5.1. Pla de treball

El desenvolupament del projecte s'executarà segons la següent planificació:

	Name	Start	Finish
1	Informe de requeriments	9/17/12 8:00 AM	9/28/12 5:00 PM
2	Redacció Pla de Treball	9/29/12 8:00 AM	10/1/12 1:00 AM
3	Instal·lació entorn de proves	10/1/12 1:00 AM	10/7/12 5:00 PM
4	**Fase 1 : Instal·lació Base**	**10/7/12 5:00 PM**	**11/3/12 5:00 PM**
5	Selecció del programari	10/7/12 5:00 PM	10/13/12 5:00 PM
6	Instal·lació del programari de sincronització	10/14/12 5:00 PM	10/20/12 5:00 PM
7	Instal·lació del programari de Cache	10/18/12 2:00 AM	10/28/12 8:00 PM
8	Proves de funcionament i rendiment	10/29/12 6:00 PM	11/3/12 5:00 PM
9	**Fase 2 : Configuració**	**11/4/12 5:00 PM**	**11/25/12 5:00 PM**
10	Especificacions tècniques	11/4/12 5:00 PM	11/7/12 5:00 PM
11	Modificacions de l'aplicació	11/8/12 5:00 PM	11/18/12 5:00 PM
12	Configuració acurada sincronització	11/8/12 5:00 PM	11/18/12 6:00 PM
13	Configuració acurada Cache	11/8/12 8:00 PM	11/18/12 4:00 PM
14	Proves de funcionament i rendiment	11/18/12 6:00 PM	11/25/12 5:00 PM
15	**Entorn de producció**	**11/26/12 5:00 PM**	**12/5/12 5:00 PM**
16	Instal·lació en entorn real	11/26/12 5:00 PM	11/28/12 5:00 PM
17	Proves de funcionament	11/28/12 5:00 PM	12/5/12 5:00 PM

Pel desenvolupament de la solució plantegem una execució en 2 fases, a la primera fase construirem el sistema complet amb les configuracions per defecte sense tenir en compte les particularitats del sistema ni tots els requeriments de seguretat i avaluarem el seu funcionament. A la segona fase realitzarem les configuracions específiques de cada element i afegirem els requeriments de seguretat.

5.2. Instal·lació de l'entorn de proves / desenvolupament

Per tal de disposar d'una plataforma el més propera possible a la de producció replicarem el sistema final en forma de màquines virtuals utilitzant el software gratuït Oracle VirtualBox 4.2.1 (GNU GPLv2). Les màquines seran les següents:

Node Extern (NE)	
Hostname	node01
IP	192.168.1.201
Sistema Operatiu	Ubuntu Server 12.04 LTS (GNU GPL i altres llicències lliures)
Frontal Web	Varnish Cache 3.0.2 (BSD)
Servidor Web	NGINX 1.1.19 (BSD 2-Clause-like)
Firewall	IPTABLES 1.4.12 (GNU GPL)
SSL	OpenSSL 1.0.1 (Apache License 1.0 i 4-clause BSD License)
Descripció:	Servidor que actuarà com a Node Extern o sigui com a Frontal Web de les peticions als serveis que tingui assignats a través de la configuració dels registres DNS dels diferents dominis. Servirà tant contingut dinàmic a través del Varnish Cache com el contingut estàtic que li proveirà el servidor NGINX. També disposarà d'una replica en temps real del repositori de contingut estàtic.

Node Repositori de contingut estàtic (ST)	
Hostname	static
IP	192.168.1.200
Sistema Operatiu	Ubuntu Server 12.04 LTS (GNU GPL i altres llicències lliures)
Sincronització	RSYNC 3.0.9 (GNU GPLv3)
Sincronització realtime	LSYNCD 2.04 (GPLv2 i posteriors)
Firewall	IPTABLES 1.4.12 (GNU GPL)
SSL	OpenSSL 1.0.1 (Apache License 1.0 i 4-clause BSD License)
Descripció:	Servidor que actuarà com a repositori de contingut estàtic i que sincronitzarà en temps real els diferents nodes. Per fer-ho utilitzarà els programes LSYNCD i RSYNC.

Node dinàmic (DY)	
Hostname	web
IP	192.168.1.100
Sistema Operatiu	Windows 2008 Server (Microsoft EULA)
Servidor Web	IIS 7.5 (Microsoft EULA)
Framework d'aplicació	ASP.NET 4.0 (Microsoft EULA)
Servidor Bases de Dades	SQL Server 2008 (Microsoft EULA)
Descripció:	Servidor que conté l'aplicació de contingut dinàmic i el servidor de bases de dades. És el responsable d'atendre totes les peticions que li facin els nodes externs i retornar el resultats per ser enviats als clients.

Client	
Hostname	pc
IP	x.x.x.x
Sistema Operatiu	Mac OS X 10.7.4 (Apple EULA)
Navegador	Mozilla Firefox 16.02 (MPL 2.0)
Descripció:	Ordinador de sobretaula que actua com a client, realitzant peticions de pàgines a les URLs de prova.

A l'annex 1 podeu veure una relació de tot el programari utilitzat, l'adreça web oficial del producte i un enllaç a la llicència del mateix.

5.3. Fase 1. Instal·lació base

En aquesta fase construirem un entorn funcional molt simple que permetrà comprovar el funcionament de tots els elements i la seva interconnexió sense entrar en les particularitats del nostres sistema, especialment les del node dinàmic. A continuació mostrem una relació del conjunt de tasques que es duran a terme en aquesta fase, entre parèntesis s'indica a quins servidors es realitzaran aquestes accions:

Instal·lació o actualització d'aplicacions

- Instal·lació o actualització de RSYNC (ST).
- Instal·lació o actualització de OPEN SSL (ST, NE).
- Instal·lació o actualització de IPTABLES (ST, NE).
- Instal·lació de LSYNCD (ST).
- Instal·lació de Varnish Cache (NE).

Configuracions locals

- Creació de l'usuari de sincronització (ST, NE)
- Modificació de permisos de les carpetes de sincronització (ST, NE).
- Creació de les claus pública i privada de l'usuari de sincronització (ST).
- Importació de la clau pública de l'usuari de sincronització (NE).

Configuració dels serveis

- Configuració bàsica i inici de RSYNC i LSYNCD (ST)
- Modificació del port d'escolta de NGINX (NE)
- Configuració bàsica i inici de Varnish Cache (NE)

Proves de funcionament (ST,NE,DY)

- Codificació dels scripts de prova
- Execució dels scripts
- Avaluació dels resultats

5.4. Fase 2. Configuració

L'objectiu d'aquesta fase és generar totes les configuracions específiques per el correctament funcionament de la solució. Així doncs haurem de tornar sobre tots els elements instal·lats a la fase anterior i anar fent els afinaments necessaris per tal de complir tots els requeriments.

5.4.1. Modificacions a l'aplicació de contingut dinàmic

Per garantir que no perdem funcionalitats amb el procés de *cache* i ens permeti maximitzar el nombre de d'elements susceptibles de ésser servits de la memòria cau cal revisar quines tasques realitzem a la banda del servidor que podríem traslladar a la banda client.

- **URLs de contingut estàtic:** Quan es construeixen les URLs de contingut estàtic a les planes de contingut dinàmic (les imatges per exemple) totes aquestes apunten al mateix subdomini de contingut estàtic static.educaweb.com. També s'ha observat que el contingut estàtic servit com a subdomini d'un domini que porta Cookies acostuma a porta aquestes mateixes Cookies a les seves crides. Per aquest motiu es comprarà un nou domini per servir el contingut estàtic amb el que s'escriuen les URLs i servir-lo amb un subdomini específic per a cada aplicació.

- **Traçabilitat de les visites:** L'aplicació porta una traça de l'activitat dels usuaris (*log*) que es perdrà si afegim la funcionalitat de memòria cau, per això caldrà traslladar aquesta funcionalitat a la banda del client, probablement via un *Javascript* que permeti enregistrar aquest *log* de forma dinàmica i asíncrona a on el tractament de la *Cookie* de sessió serà suportat per aquest *script* de client. Per això es crearà un Servei Web que enregistri les estadístiques i es cridarà aquest servei des d'un script asíncron situat a les planes servides.

5.4.2. Elements estàtics

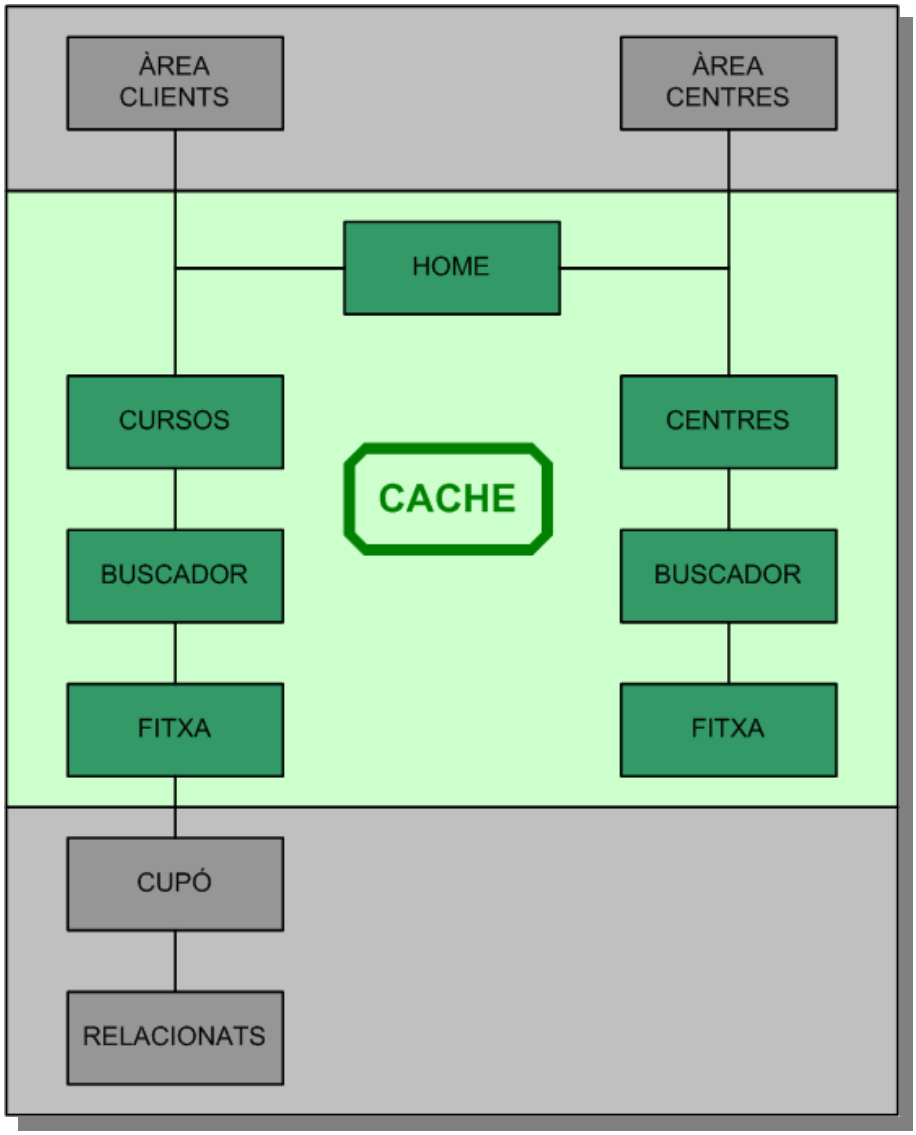
Es consideraran elements dinàmics els següents tipus d'arxius, en cas de que canviïn el procés de sincronització s'encarregarà de mantenir-les actualitzades entre els nodes.

- **Imatges:** Totes les imatges que incorporen les pàgines són contingut estàtic. Les extensions més habituals són: .jpg .gif .png .bmp
- **Javascript:** Scripts de client, amb extensió .js
- **Fulls d'estil:** Els arxius CSS que acompanyen les pàgines, tenen extensió .css.
- **Documents:** Altres documents que es serveixen com fitxers, les extensions més habituals són: .pdf .doc .docx .odt

5.4.3. Elements dinàmics

Totes les pàgines processades per la capa d'aplicació, en general tots els continguts processats en ASP.NET que utilitzen l'accés a la base de dades per a la seva construcció.

A continuació podem veure un esquema dels blocs de pàgines dinàmiques que es generen a l'aplicació. En verd marquem els elements amb possibilitats d'ésser emmagatzemats a la memòria cau i en gris els elements que depenen fortament de la sessió i que no podran ser emmagatzemats a la memòria cau.



A banda de les pàgines web que es serveixen a l'usuari dins de les pròpies planes hi ha elements AJAX que executen crides al servidor. Aquestes en la majoria de casos recuperen continguts dinàmics que no susceptibles d'ésser guardats a la memòria cau.

Per últim les pàgines d'errors 4XX i 5XX també caldrà tractar-les ja que podríem tenir problemes de caigudes puntuals del servidor dinàmic i que un cop aquest es recuperés el cache continués servint la pàgina d'error.

Per identificar **les pàgines NO cachejables** de les que no utilitzarem la següent lògica:

- Les pàgines de l'Àrea de clients i l'Àrea de centres que sempre comencen amb l'estructura: /client o /user.

- Les pàgines Cupó i Relacionats es composen de 3 pàgines .aspx /buscador/informacion.aspx, /buscador/relacionados.aspx i /buscador/relacionados2.aspx.

- Les crides AJAX que es produeixen dins d'aquestes pàgines que són realitzades amb pàgines .ASPX o .ASHX.

- Les pàgines d'errors amb codis de resposta 4XX o 5XX.

És possible que un anàlisi més detallat de les lògica anterior ens permetés aprofitar més exhaustivament les planes dinàmiques però complicarien en excés la lògica d'elecció i seria una font potencial de problemes pel que ens limitarem a aquesta lògica per tant **tota la resta de pàgines són cachejables**.

5.4.4. Temps d'expiració

Un cop decidides les planes que són candidates a ésser cachejades caldrà definir quin és el seu temps d'expiració. Degut a que l'aplicació d'Educaweb és força dinàmica i els seus continguts varien constantment no es considera prudent establir un temps d'expiració pel contingut dinàmic superior a 1 dia, així doncs **s'estableix una regla general per a totes les planes d'expiració de 24 hores**, moment en el que seran tornades a sol·licitar al servidor dinàmic.

Evidentment és possible definir excepcions a aquesta regla en un posterior afinament i donar un temps d'expiració més o menys llarg per algunes pàgines específiques, aquest afinament serà un procés de manteniment i experimentació continua un cop el sistema estigui en funcionament.

5.4.5. Cookies

A la seva configuració per defecte Varnish tracta totes les peticions que portin Cookies com a peticions no-cachejables. Això seria un greu problema si no fessim cap tractament ja que totes les peticions de contingut dinàmic que fa la

nostra petició porten Cookies, per tant caldrà que fem un anàlisi detallat de quin tractarem en farem.

A continuació mostrem una capçalera de crida HTTP i les Cookies que incorpora per analitzar-les i definir el comportament de Varnish.

```
Cookie: BuscadorEducaweb=54519732;
__utma=257518303.1345701189.1331226791.1353690482.1353774240.30;
__utmz=257518303.1353690482.29.7.utmcsr=google|utmccn=(organic)|
utmcmd=organic|utmctr=(not%20provided); __unam=aab56d-135f34cd34a-
5325f7b8-49; edwt-22=1; ASP.NET_SessionId=2zvg25xjfuhmz5fxb3feo0ts;
HA_EDUCAWEB_CK_1=web01; __utmb=257518303.2.10.1353774240;
__utmc=257518303
```

__utma __utmz __unam __utmb

Són de Google Analytics per mantenir estadístiques sobre l'ús del lloc web. Els scripts client de GA les utilitzen en la seva comunicació amb els servidors de GA, podem ignorar-les.

BuscadorEducaweb

Identificador únic d'ordinador utilitzat pel seguiment de navegació de l'usuari. Aquest identificador haurà de traslladar-se al script de registre de la navegació definit a les modificacions de l'aplicació de contingut dinàmic, pel que desapareixerà i podem desestimar-lo.

edwt-XX

En la millora de l'experiència d'usuari, Educaweb realitza experiments amb diferents versions de les pàgines, aquesta cookies registra quina versió de la pàgina està veient l'usuari pel que caldrà tractar-la especialment. Una possible solució serà l'ús de l'atribut Vary a la capçalera http, amb l'ús de d'aquest atribut el Cache guarda les diferents versions i serveix la corresponent. Aquestes cookies no sempre són presents a les crides ja que només hi apareixen quan hi ha un experiment en curs.

ASP.NET_SessionId

És un identificador de sessió d'ASP.NET, en general no és necessari enviar-la per a totes les peticions que ja han estat classificades com a cachejables, per a la resta cal enviar-la per mantenir les dades de sessió.

HA_EDUCAWEB_CK_1

L'aplicació dinàmica disposa d'un balancejador de càrrega i aquesta cookie permet mantenir els usuaris dins del mateix servidor durant tota la sessió. És important mantenir les peticions de sessions de pàgines no cachejables dins del mateix servidor ja que si no ho féssim ens trobaríem amb comportament no desitjat, per a les pàgines cachejables aquesta cookie no és important tot i que la podem utilitzar per realitzar el balanceig de càrrega entre els diferents frontals de l'aplicació dinàmica.

5.4.6. Mida de la memòria cau

Un cop definides les planes que poden entrar dins la memòria cau i el seu temps d'expiració cal preveure la mida de la memòria cau per que permeti emmagatzemar el màxim nombre de planes possibles i així maximitzar el benefici. Cal tenir present que Varnish enmagatzema tot el cache en memòria per tant aquest serà un punt bàsic. Per tal de calcular les necessitats de memòria s'ha realitzat la següent taula on es mostren les pàgines servides per un dia tipus a un dels servidors internacionals.

Pàgines	Hits	Diferents	Mida Pàg (KB)	Memòria (KB)
Home	800	1	7	7
Cursos				
Home	500	1	9	9
Buscador	10.000	5.175	15	77.625
Fitxa	8.500	5.665	9	50.985
Centres				
Home	220	1	9	9
Buscador	2.200	1.000	15	15.000
Fitxa	2.500	916	9	8.244
Totals	**24.720**	**12.759**	**10**	**151.879**

Com es pot observar a la taula anterior es possible aconseguir una reducció de sol·licituds de pàgines al servidor dinàmic del 50% amb una utilització aproximadament de 150Mbytes de memòria. De tota manera, com veurem més endavant hi ha altres consideracions a tenir en compte que poden fer créixer aquestes necessitats de memòria fins a duplicar-les. A més preveient un augment del tràfic degut a només funcionalitats futures caldrà dimensionar el servidor per tal que disposi de almenys 2GBytes de memòria RAM o més.

5.4.7. Configuració de les polítiques de seguretat

En aquesta fase configurarem IPTABLES per tal de que només permeti l'accés als serveis estrictament necessaris des de les IPs de confiança. Així doncs per defecte tots els serveis al node extern estaran denegats excepte:

- Connexions d'entrada http al port 80 que escoltarà varnish.
- Connexions ssh al port 22 des de la IP del repositori de contingut estàtic que escoltarà el servidor de ssh.
- Connexions de sortida al port 80 dels servidors de contingut dinàmic.

6. Entorn de producció

Durant el desenvolupament del projecte s'ha explorat la possibilitat de contractar un servidor remot als Estats Units i fins i tot s'han fet certes proves però final s'ha decidit que la instal·lació pilot es farà al node de Barcelona per minimitzar els costos del pilot. Si aquests són satisfactoris és contractarà un hosting remot i replicarà la solució.

Per tal de realitzar una instal·lació el més propera a la realitat possible és realitzaran totes les tasques com si el servidor estigues efectivament allotjat al hosting remot. L'objectiu és servir els webs Educawb.mx i Educaweb.br.com des d'aquest node, tant el contingut dinàmic com l'estàtic.

6.1. Instal·lació d'una màquina virtual

S'ha muntat un servidor virtual al mateix node que tenim actualment a Barcelona amb les següents característiques.

Node Extern (NE)	
Hostname	Linux02
IP	77.73.87.232
Sistema Operatiu	Ubuntu Server 12.04 LTS (GNU GPL i altres llicències lliures)
Frontal Web	Varnish Cache 3.0.2 (BSD)
Servidor Web	NGINX 1.1.19 (BSD 2-Clause-like)
Firewall	IPTABLES 1.4.12 (GNU GPL)
SSL	OpenSSL 1.0.1 (Apache License 1.0 i 4-clause BSD License)
Descripció:	Servidor que actuarà com a Node Extern o sigui com a Frontal Web de les peticions als serveis que tingui assignats a través de la configuració dels registres DNS dels diferents dominis. Servirà tant contingut dinàmic a través del Varnish Cache com el contingut estàtic que li proveirà el servidor NGINX. També disposarà d'una replica en temps real del repositori de contingut estàtic.

Contractació de dominis per contingut estàtic

Per tal de servir el contingut estàtic des d'un domini diferent al contingut dinàmic i evitar que les Cookies s'arrosseguin entre subdominis hem comprat el domini **edcwb.com** que servirà per servir tot el contingut estàtic. En aquesta primera instal·lació servirem el contingut estàtic com es mostra a continuació:

- Node de Barcelona: eu.edcwb.com
- Node de América: us.edcwb.com

6.2. Configuració de DNS

Un cop contractat el domini hem afegit les regles de DNS necessàries per aconseguir que cada subdomini apunti a la seva IP correcte. Des de fa temps per aconseguir que els navegadors serialitzin les peticions al contingut estàtic s'utilitzen un conjunt de subdominis a la crida dels continguts estàtics, caldrà tenir en compte aquesta particularitat.

Per el node Barcelona configurarem les entrades DNS:

eu.edcwb.com.	A	77.73.87.231
eu00.edcwb.com.	CNAME	eu.edcwb.com.
eu01.edcwb.com.	CNAME	eu.edcwb.com.
eu02.edcwb.com.	CNAME	eu.edcwb.com.
eu03.edcwb.com.	CNAME	eu.edcwb.com.
eu04.edcwb.com.	CNAME	eu.edcwb.com.

Per el node "americà" configurarem les entrades DNS:

us.edcwb.com.	A	77.73.87.232
us00.edcwb.com.	CNAME	us.edcwb.com.
us01.edcwb.com.	CNAME	us.edcwb.com.
us02.edcwb.com.	CNAME	us.edcwb.com.
us03.edcwb.com.	CNAME	us.edcwb.com.
us04.edcwb.com.	CNAME	us.edcwb.com.

Per tal de realitzar proves en l'entorn real sense afectar l'entorn de producció s'ha creat l'entrada de DNS següent:

ww2.educaweb.br.com.	A	77.73.87.232
ww2.educaweb.mx	a	77.73.87.232

Si aquestes proves són satisfactòries la modificació que caldrà fer serà:

www.educaweb.br.com.	A	77.73.87.232
www.educaweb.mx.	A	77.73.87.232

6.3. Instal·lació de la replicació del contingut dinàmic

El servidor Linux01 que actualment serveix el contingut estàtic del node de Barcelona farà de node mestre de contingut estàtic i sincronitzarà el seu contingut amb la resta de nodes, per tant caldrà configurar-lo per tal de que comenci a sincronitzar-se amb el servidor Linux02. A l'annex 1 trobarem instruccions detallades d'aquest procés.

6.4. Instal·lació del Node Extern

- Actualització a la versió Ubuntu 12.04
- Creació d'usuaris i grups d'administració i web
- Instal·lació i configuració de NGINX
- Instal·lació i configuració d'OpenSSL
- Instal·lació i configuració de RSYNC i LSYNCD
- Instal·lació i configuració de Varnish
- Instal·lació i configuració d'IPTABLES
- Instal·lació i configuració de Munin.

Podeu veure les instruccions concretes per a la instal·lació de tots els elements a l'Annex 1.

6.5. Confinguració de Varnish

- Configuració dels backends de contingut dinàmic i estàtic.
- Redirecció del contingut estàtic al servidor NGINX.
- Detecció de pàgines cachejables.
- Redirecció de pàgines no cachejables.
- Eliminació de Cookies.
- Temps d'expiració del cache.
- Gestió de les versions d'experimentació.

A l'annex 4 es pot veure una relació complerta de totes les accions que cal dur a terme a cada servidor per obtenir aquesta configuració.

7. Resultats

Degut a les diferents etapes a desenvolupar i la seva dependència entre elles és aviat per poder parlar de resultats definitius però intentarem explicar els resultats parcials obtinguts en aquest moment.

En aquests moment tenim en funcionament:

- El servidor virtual que actuarà de node extern instal·lat i replicable amb mínims retocs d'ips.
- La sincronització d'arxius estàtics en funcionament.
- El servidor NGINX per contingut estàtic al Linux02 escoltant el port 8080 en funcionament amb *failover* al servidor de Linux01.
- El servidor Varnish servint peticions per l'adreça ww2.educaweb.br.com i ww2.educaweb.mx servint el contingut dinàmic equivalent a www.educaweb.br.com i www.educaweb.mx i redirigint les peticions del contingut estàtic al NGINX del Linux02.
- El servidor Munin per monitoritzar l'estat del servidor i dels seus serveis, Varnish entre d'ells.

A la finalització d'aquesta memòria el servidor Linux02 està en ple funcionament servint de forma correcta els continguts dels dominis de proves i aprofitant totes les possibilitats del Varnish Cache. Per la posada en producció final només cal que:

- L'equip de desenvolupament acabi les modificacions de l'aplicació de contingut dinàmic.
- Modificació del registres DNS dels serveis que vulguem publicar sota Varnish.

Tal i com es veurà al capítol de conclusions totes les funcionalitats demandades al projecte han estat assolides i es disposa d'un servidor web totalment funcional replicable a qualsevol lloc del món amb mínims retocs de configuració.

Simulacions de rendiment

Com ja s'ha comentat els resultats funcionals del projecte han estat un èxit ja que la instal·lació funciona sense problemes però caldrà esperar al desplegament en producció per treure les conclusions finals sobre el nivell d'assoliment d'objectius en quant a millora del rendiment. Per això hem realitzat alguns test d'estrès al servidor per poder mesurar-ne el seu rendiment quan aquest es posi en producció.

Utilitzant els logs d'activitat del servidor s'ha extret un porció de l'activitat d'un dia i s'ha creat un fitxer amb 5640 urls que representen peticions reals durant unes hores al servidors. Aquestes urls s'han seleccionat per a que siguin *cachejables* i poder mesurar el comportament i rendiment del cache.

El resultats són els següents:

	Temps de descàrrega	Request	Total
Directe	1,04s	5640	5865,60s
Varnish Miss	1,17s	3165	3703,05s
Varnish Hit	0,41s	2505	1027,05s

S'observa que el Varnish Cache suposa una sobrecàrrega de temps en cas de que no tingui la pàgina a la seva memòria de 130ms en canvi quan dispossem de la pàgina a la memòria cau Varnish millora en 630ms la velocitat de les crides directes. Amb aquesta mostra d'URLs Varnish millora en un 20% el temps d'entrega de les pàgines.

Si ampliem la taula calculada a l'apartat 2.2 de la memòria extrapolant els resultats obtinguts:

Pàgines	Pàgines Úniques	%	Temps Directe	Temps Varnish	Millora
17.992	8.219	45,68%	18.712	13.623	27,19%
33.733	14.159	41,97%	35.082	24.591	29,90%
40.000	15.684	39,21%	41.600	28.320	31,92%
54.843	17.962	32,75%	57.037	36.137	36,64%
61.522	17.746	28,84%	63.983	38.711	39,50%
204.027	49.657	24,34%	212.188	121.390	42,79%
663.002	103.208	15,57%	689.522	350.269	49,20%
1.857.409	187.685	10,10%	1.931.705	904.178	53,19%
2.664.334	238.860	8,97%	2.770.907	1.273.911	54,03%
4.619.921	340.207	7,36%	4.804.718	2.152.725	55,20%

Un dia normal amb 40.000 pàgines cachejables Varnish millora en un 32% el temps de resposta de les pàgines.

8. Valoració econòmica

8.1. Costos d'implantació

A continuació mostrem una taula dels costos d'implementació del projecte:

	Hores	Cost Unitari	Cost
Informe de requeriments	40	25,00 €	1.000,00 €
Redacció Pla de Treball	8	25,00 €	200,00 €
Instal·lació entorn de proves	20	25,00 €	500,00 €
Instal·lació Base			0,00 €
Selecció del programari	40	25,00 €	1.000,00 €
Instal·lació del programari de sincronització	20	25,00 €	500,00 €
Instal·lació del programari de Cache	30	25,00 €	750,00 €
Proves de funcionament i rendiment	40	25,00 €	1.000,00 €
Configuració			0,00 €
Especificacions tècniques	30	25,00 €	750,00 €
Modificacions de l'aplicació	80	25,00 €	2.000,00 €
Configuració acurada de la sincronització	20	25,00 €	500,00 €
Configuració acurada del Cache	60	25,00 €	1.500,00 €
Proves de funcionament i rendiment	20	25,00 €	500,00 €
Entorn de producció			0,00 €
Instal·lació en entorn real	30	25,00 €	750,00 €
Proves de funcionament	20	25,00 €	500,00 €
Total			11.450,00 €

A banda dels costos d'implementació cal considerar els costos de recurrents que tindrem en la posada en producció del mateix.

	Hores	Cost Unitari	Cost
Cost Hosting VPS (anual)			614,40 €
Manteniment Plataforma (anual)	50	25,00 €	1.250,00 €

8.2. Increment d'ingressos

La millor de la velocitat de càrrega d'un lloc web té influència en els següents elements:

- **Rati de conversió:** El nombre d'usuaris que acaben realitzant l'objectiu que busquem al nostre lloc web, una compra, una sol·licitud d'informació, un registre d'usuari, etc.

- **Rati d'abandonament:** El nombre d'usuaris que abandonen el nostre lloc web al visitar la primera pàgina, també se l'anomena rati de rebot o *bounce rate* en anglés.

- **Rati de retorn:** El nombre de visitants que tornaran a visitar-nos ja que la seva experiència d'usuari ha estat positiva.

- **Confiança en la marca:** Diversos estudis han conclòs que existeix una relació entre el rendiment d'un lloc web i la impressió que aquest deixa sobre la marca als visitants.

- **Posicionament orgànic:** Fa força temps que Google va anunciar que utilitzaria la velocitat de càrrega dels llocs web com un dels múltiples factors que utilitza per mesurar la rellevància dels llocs web a la seva sortida de resultats orgànics.

Per tant s'espera que amb l'execució d'aquest projecte millori el nombre de visitants, visites i sol·licituds. Com ja s'ha mencionat anteriorment els ingressos de l'activitat provenen de les sol·licituds d'informació que rep el lloc web, així doncs l'increment un increment en el nombre de visites i/o el rati de conversió augmenta de forma directa el nombre d'ingressos.

A tall d'exemple si el nombre de visites i el rati de conversió augmenten un 10%, el nombre de sol·licituds rebudes augmentarà un 10%, per tant els ingressos i així progressivament.

	Visites	CRT	Conversions
0%	100.000	4,00%	4.000
10%	110.000	4,40%	4.840
20%	120.000	4,80%	5.760

8.3. Reducció de costos

Amb l'ús de llicències de programari lliure s'ha aconseguit dissenyar una solució que permet una reducció de costos per dos motius.

1. El cost de les llicències de programari: Tot el programari utilitzat està basat en llicències de programari lliure sense cost, l'alternativa hagués estat muntar sistemes basats en Windows 2008 Server Web Edition que té un cost d'uns 400€ per llicència.

2. Els requeriments de maquinari dels nodes externs: Windows té uns requeriments de maquinari força superiors als de Linux, això hagués obligat a contractar maquinari amb prestacions més elevades i per tant més costos.

9. Conclusions

9.1. Acompliment d'objectius

El principal objectiu del projecte era la millora del rendiment en la càrrega de les pàgines web que serveix el portal. Tal i com s'ha indicat el procés de càrrega depén del temps de processament i del temps de transferència. Per això el projecte ha atacat aquests dos aspectes, **acostant geogràficament els continguts** dels serveis als usuaris internacionals per millorar la distancia entre usuari i servidor i **minimitzar la càrrega del servidor dinàmic**.

El primer objectiu l'acostament geogràfic dels continguts no s'ha pogut aconseguir durant l'execució del projecte degut tot i que els resultats són molt esperançadors ja que s'ha aconseguit una màquina virtual desplegable a qualsevol lloc en molt poc temps i amb un cost molt baix i que de ben segur, si es tria el hosting adient, oferirà una molt millor connectivitat als usuaris llunyans.

Pel que fa al segon objectiu la minimització de la càrrega del servidor s'ha aconseguit clarament ja que Varnish és capaç de servir el 60% del contingut en un dia normal, tot aquest processament queda alliberat al servidor dinàmic cosa que redundarà en un millor temps de resposta per a les planes no cachejades i una escalabilitat major de la plataforma.

9.2. Acompliment de requeriments

Analitzarem els requeriments inicials del projecte i el seu grau d'acompliment amb la realització del projecte.

Reduir el temps de càrrega de les pàgines.

Fins que el projecte no estigui en ple funcionament serà impossible avaluar completament el grau d'acompliment d'aquest requeriment ja que cal monitoritzar el rendiment del servidor des de localitzacions remotes i per això cal contractar el servei de monitorització quan el web que es desplegui.

Tal i com s'ha vist al capítol de resultats el sistema millora els temps de càrrega substancialment. Amb la càrrega de pàgines d'un dia normal **Varnish aconsegueix millorar el temps de càrrega en un 32%** i en dies amb forta audiència aquesta millora pot arribar al 40%.

Minimitzar les necessitats de processament de l'aplicació per permetre l'increment d'usuaris.

Amb les estadístiques proporcionades de repetició de pàgines dinàmiques servides es pot comprovar que efectivament **s'ha reduït les necessitats de processament de pàgines en un 50%** cosa que permetrà alliberar de procés al servidor de contingut dinàmic.

Que sigui una solució fàcilment escalable.

Un cop establerta la configuració base de la màquina remota, afegir-hi altres màquines que donin servei a altres webs o fins i tot al mateix web balancejant la càrrega és un procés trivial que es resol amb el desplegament d'un nou servidor amb una configuració pràcticament idèntica a la del primer node.

Que sigui transparent a l'aplicació de contingut dinàmic.

Tot i que s'ha intentat aconseguir que l'aplicació de contingut dinàmic no hagués de patir cap modificació això ha estat impossible degut a certes peculiaritats de la mateixa com el registre d'estadístiques a la banda del servidor o les URLs de contingut estàtic basades en un subdomini. D'altra banda aquest inconvenient ha estat clau per l'endarreriment de la posada en producció del projecte ja que no és trivial la construcció del *script* de recollida de dades i del servei web de recepció de les dades.

La solució ha de posar-se en producció sense aturades de servei pels usuaris actuals.

En aquests moments el sistema no està en producció tot i així es preveu acomplir plenament aquest requeriment ja que només es farà el canvi de DNS per a que apuntin al node extern quan el sistema estigui completament validat, d'altra banda com que el sistema actual no es destrueix ja que és totalment necessari per servir les peticions de contingut dinàmic seguirem tenint un servidor totalment operatiu en cas de problemes per tornar a la situació inicial.

Que el seu manteniment sigui escalable

Al tractar-se de nodes pràcticament idèntics els sistemes seran fàcilment gestionables i monitoritzables. L'increment de nodes no implicarà un augment de gestió directament proporcional al nombre de nodes ja que les configuracions són pràcticament idèntiques. El fet de disposar d'eines de monitorització automatitzada ens permetrà tenir un mapa clar de la disponibilitat dels serveis en tot moment.

Mantenir els costos el més acotats possibles, utilitzant eines de programari lliure sempre que sigui possible.

Els costos d'implantació no han estat baixos en termes de recursos humans com es pot veure a la taula de costos, tot i així un cop implementada la solució els costos de manteniment i d'infraestructura són realment baixos i assumibles . D'altra banda gràcies a l'ús d'aplicacions de programari lliure no s'han incrementat els costos en llicències i el que és més important, la replicació d'aquest model tindrà un cost molt més baix ja que com s'ha indicat anteriorment el manteniment de diversos no incrementa els costos de manteniment de forma lineal.

Que compleixi amb la legislació vigent sense gaires dificultats.

La base de dades d'usuaris i clients es manté al node dinàmic per tant no hi ha transferència de dades internacionals més enllà de les que l'usuari voluntàriament introdueix al sistema. En el cas de que haguéssim optat per la implementació de nodes actius hagués estat necessari la replicació de la base de dades i per tant de tota la informació d'usuaris amb el que hauríem de tenir molta cura de complir correctament la Llei Orgànica de Protecció de Dades (LOPD).

9.3. Els continguts del Màster i el projecte.

Tal i com veurem a continuació gran part dels continguts del Màster han estat de gran utilitat per a la realització d'aquest projecte. Per tal d'il·lustrar-ho mostraré una relació de les assignatures cursades que han estat claus en un bon coneixement i gestió del projecte.

Administració de sistemes GNU/Linux
Administració avançada del sistema operatiu GNU/Linux

En general el coneixement de les eines d'instal·lació d'aplicacions, configuració de permisos i estructura de directoris de les distribucions de Linux ha estat molt útil per moure's amb facilitat pels sistemes. Més en concret la instal·lació d'OpenSSH i la creació d'usuaris *passwordless* van ser pràctiques desenvolupades a aquestes assignatures. També la realització de scripts bash ha facilitat la lectura i modificació dels scripts d'inicialització dels *daemons*.

Aspectes avançats de seguretat en xarxes

A aquesta assignatura es va veure en profunditat el programari de tallafocs *iptables* que ha estat de gran utilitat per la configuració d'una bona estructura de seguretat especialment per protegir l'accés via SHH només a les màquines de confiança.

També el coneixement d'eines com *Wireshark* utilitzades en aquesta assignatura han permès la monitorització de la transferència de dades entre els diferents equips.

Desenvolupament d'aplicacions web

Gràcies als coneixements del protocol HTTP, les capçaleres, les Cookies, els mètodes, la compressió de les dades, les diferències entre el contingut estàtic i dinàmic, entre d'altres, ha estat més fàcil entendre les implicacions que s'han de tenir en compte en quant al cache de pàgines web.

També es va treballar la configuració del servidor web HTTP Apache i tot que el projecte utilitzarà un servidor diferent tots els conceptes tenen validesa per entendre'n el funcionament i la seva configuració.

Introducció al programari lliure

El coneixement de les diverses llicències existents en l'àmbit del programari lliure ha estat clau per entendre de forma correcta cadascuna de les llicències en que es basen el programes utilitzats.

Implantació de sistemes de programari lliure

La gestió de projectes era un punt clau en aquesta assignatura el que ha donat un caire més formal al disseny de la solució en general. L'estudi de diversos projectes de programari lliure durant l'assignatura m'ha permès moure'm amb facilitat per les pàgines oficials dels projectes i trobar la documentació adequada en tot moment.

9.4. Principals problemes

Com tot projecte aquest no ha estat exempt de problemes dels que en faré un breu recull.

Pla de treball

En primer lloc la realització del Pla de Treball en un punt tan inicial de les pràctiques no permet tenir una visibilitat clara de com serà la millor manera de resoldre el problema, fet que ha provocat dos inconvenients: un pla inicial molt poc detallat, i un canvi substancial del pla passades 3 setmanes de l'inici de les pràctiques.

Tasques dependents fora de projecte

En segon lloc, la càrrega de treball prevista per a la realització del projecte és viable assolir-la durant el temps previst però al tractar-se d'un tema tant relacionat amb els productes principals de l'empresa i que a l'hora requereix de tasques parcials fora de l'àmbit del projecte que han de ser realitzades per l'equip de desenvolupament però necessàries fan perillar la possibilitat d'assolir tots els objectius en la franja de temps de la realització d'aquest projecte.

Posada en producció

Per això en el moment de tancament d'aquesta memòria disposarem d'una versió totalment funcional del projecte i a punt per ser posada en producció però no serà possible fer el pas final ja que faltarà tancar alguns aspectes de desenvolupament per part de l'equip que quedàvem fora de l'àmbit de treball d'aquest projecte.

Annex 1. Instal·lació base

A continuació mostrem una descripció pas a pas de la instal·lació base, per fer-ho s'ha seguit la següent numenclatura: si la línia comença per # es tracta d'un comentari descriptiu respecte les línies que venen a continuació, si la línia comença per *static* o *node01* fa referència al servidor en qüestió.

Instal·lació del programari

Actualització del repositori de d'actualitzacions
static@administrator:~/$ sudo apt-get update
node01@administrator:~/$ sudo apt-get update

Instal·lació de rsync
static@administrator:~/$ sudo apt-get install rsync

Instal·lació d'open ssl
static@administrator:~/$ sudo apt-get install openssl
node01@administrator:~/$ sudo apt-get install openssl

Instal·lació d'iptables
static@administrator:~/$ sudo apt-get install iptables
node01@administrator:~/$ sudo apt-get install iptables

Instal·lació de lsyncd
static@administrator:~/$ sudo apt-get install lsyncd

Instal·lació de varnish cache
node01@administrator:~/$ sudo apt-get install varnish

Creació del login ssh *passwordless*

Creació de l'usuari de sincronització, actualment ja tenim al repositori estàtic l'usuari ftp-data que pertany al grup www-data que és el propietari del repositori, pel que només el crearem

node01@administrator:~/$ sudo adduser –home /home/ftp-data –shell /bin/bash –ingroup www-data ftp-data

Farem login com a ftp-data i crearem el directori .ssh dins el directori /home/ftp-data per poder-hi guardar les claus.

node01@ftp-data:~/$ mkdir .ssh

Farem login com el usuari ftp-data al servidor *static* per tal de generar les claus rsa i poder configurar l'accés via ssh sense password que utilitzarà rsync per connectar-se. Quan generem la clau ens demanarà si volem passphrase però la deixarem buida. No oblidem canviar els permisos com en el pas anterior.

```
static@ftp-data:~/$ mkdir .ssh
static@ftp-data:~/$ cd .ssh
static@ftp-data:~/.ssh$ ssh-keygen -t rsa
```

Transferim la clau al node01 dins del directori .ssh que previament hem creat

```
static@ftp-data:~/$ cat ~/.ssh/id_rsa.pub | ssh node01 'cat - >> ~/.ssh/authorized_keys2'
```

Ara cal canviar els permisos a tots dos directoris .ssh creats i al seu contingut o no funcionarà el login passwordless. Els permisos han de ser d'accés total només per l'usuari propietari ftp-data i cap altre permís.

```
static@ftp-data:~/.ssh$ chmod 700 .
static@ftp-data:~/.ssh$ chmod 600 *

node01@ftp-data:~/.ssh$ chmod 700 .
node01@ftp-data:~/.ssh$ chmod 600 *
```

En aquest punt hauriem de poder connectar-nos via ssh des del servidor static al servidor node01 sense password, per provar-ho podem utilitzar la següent instrucció, el primer cop ens demanarà si confiem en el servidor que anem a connectar-nos, li direm que si.

```
static@ftp-data:~$ ssh node01
```

Configuració de la sincronització

Crearem la mateixa estructura de dades que volem sincronitzar i li assignarem els permisos de propietat a ftp-data i de grup a www-data, això ho farem com a usuari administrator

```
node01@administrator:~$ sudo mkdir /var/www/201-static
node01@administrator:~$ chown ftp-data:www-data /var/www/201-static
```

Configurarem la sincronització del repositori de *static* amb el *node01*. Encara com a usuari ftp-data crearem una carpeta al seu directori home per guardar-hi tots els arxius relatius a lsyncd

```
static@ftp-data:~$ mkdir .lsyncd
static@ftp-data:~$ cd .lsyncd
static@ftp-data:~/.lsyncd$ chmod 700 .
static@ftp-data:~/.lsyncd$  mkdir log
static@ftp-data:~/.lsyncd$  mkdir conf
```

A continuació crearem l'arxiu lsyncd.conf.lua a la carpeta conf amb el següent contingut

```
static@ftp-data:~/.lsyncd$  nano ./conf/lsyncd.conf.lua
```

```
settings = {
        delay = 1,
        maxProcesses = 5,
        logfile = "/home/ftp-data/.lsyncd/log/lsyncd.log",
        statusFile = "/home/ftp-data/.lsyncd/log/lsyncd-status.log",
        statusInterval = 20 }


        sync { default.rsyncssh,
              source="/var/www/201-static",
              host="node01",
              targetdir="/var/www/201-static"
        }
```

A continuació hem d'aconseguir que lsyncd s'executi com a *daemon* a l'inici i com a usuari ftp-data i no com a root per això editarem el script que executi el dimoni a modificarem les rutes necessàries per tal de que apunti als directoris que hem creat. Per això cal que entrem com a administrator ja que ftp-data no és del grup sudoers.

```
static@adminsitrator:~$ sudo nano /etc/init.d/lsyncd
```

Modificarem les línies següents:

```
CONFIG=/home/ftp-data/.lsyncd/conf/lsyncd.conf.lua
PIDFILE=/home/ftp-data/.lsyncd/$NAME.pid
```

Afegirem la variable USER

```
USER="ftp-data"
```

A les crides que trobem de start-stop-daemon afegirem -c $USER per aconseguir que s'executi com a usuari ftp-data, a continuació un exemple

d'una d'aquestes línies.

```
start-stop-daemon -c $USER --start --quiet --pidfile $PIDFILE --exec $DAEMON \ --test > /dev/null \
```

Per últim reiniciem el servei

```
static@adminsitrator:~$ sudo /etc/init.d/lsyncd restart
```

Si tot ha funcionat correctament en aquests moments tot el directori /var/www/201-static s'hauria d'estar replicant des de *static* al *node01*, podem fer-ne el seguiment a través dels logs que es troben a la carpeta /home/ftp-data/.lsyncd/log que hem creat anteriorment.

Configuració de Varnish Cache

Per configurar Varnish de forma sencilla sense tenir en compte les particularitats de l'arquitectura, el tema de les cookies, etc, només cal que retoquem l'arxiu /etc/varnish/default.vlc al node01

```
node01@adminsitrator:~$ sudo nano /etc/varnish/default.vlc
```

A aquest arxiu modificarem les següents línies

```
backend default {
    .host = "www.educaweb.com";
    .port = "80";
}

backend static {
    .host = "127.0.0.1";
    .port = "8080";
}
```

També descomentarem la funció sub vcl_recv completament i afegirem el següent codi al principi per utilitzar el servidor de NGINX per a totes les peticions de contingut estàtic.

```
 sub vcl_recv {

    if (req.http.host == "static.educaweb.com" || req.http.host == "static01.educaweb.com" || req.http.host == "static02.educaweb.com") {
        set req.backend = static;
     }

....
```

Per acabar comentarem el següent fragment de la mateixa funció per evitar que no faci cache de les peticions amb Cookies, evidenment aquest comportament s'haurà de cambiar per aquest punt volem provar el seu funcionament.

```
#     if (req.http.Authorization || req.http.Cookie) {
#         /* Not cacheable by default */
#         return (pass);
#     }
```

Ara ens tocarà configurar Varnish per que escolti les peticions pel port 80 ja que per defecte no ho fa, per aixo editarem l'arixu /etc/default/varnish

```
node01@adminsitrator:~$ sudo nano /etc/default/varnish
```

Buscarem l'arrecanda etiquetada com a Alternativa 2 i la descomentarem si és que no ho està comentant les altres. Un cop descomentada caldrà modificar la línia següent

```
DAEMON OPT="-a :6081 \
```

Aquí canviarem el 6081 per 80
Ens quedarà reiniciar el varnish

```
node01@adminsitrator:~$ sudo /etc/init.d/varnish restart
```

Modificació del port d'escolta de NGINX

#Al configurar varnish li hem dit que envii les peticions al localhost al port 8080 que és on trobarà el contingut estàtic. Per defecte NGINX el tenim configurat al port 80 per tant haurem de cambiar aquesta configuració editant l'arxiu /etc/nginx/sites-enabled/201-static

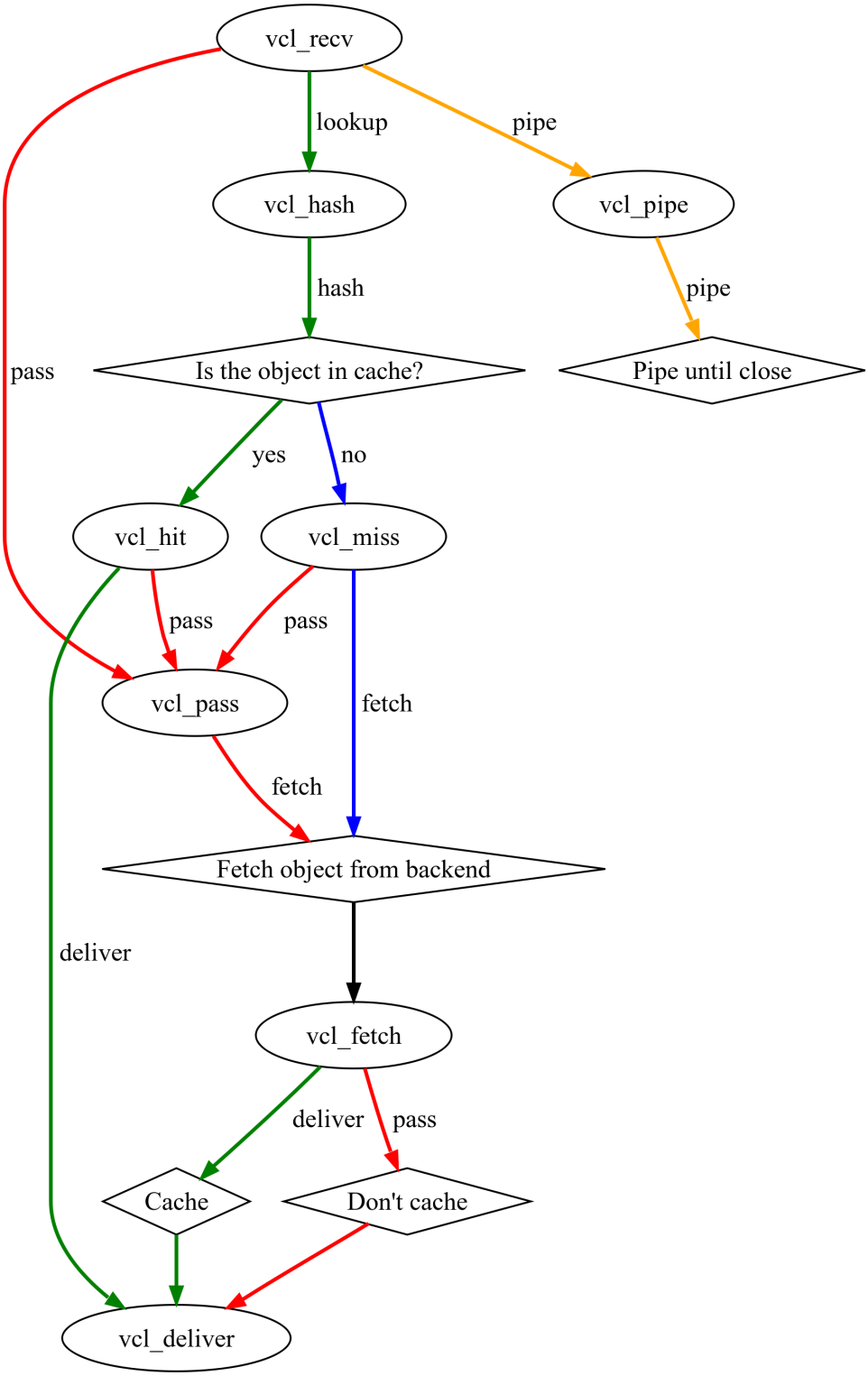
```
static@adminsitrator:~$ sudo nano /etc/nginx/sites-enabled/201-static
```

Modificarem la primera línia que diu listen 80 per listen 8080 i reiniciarem

```
static@adminsitrator:~$ sudo /etc/init.d/nginx restart
```

En aquest punt ja hem aconseguit una instal·lació completa i funcional que fa cache tant del contingut estàtic com del dinàmic, els següents pasos estaran dedicats a la configuració correcta de cada element per assolir els nostres requeriments.

Annex 2. Varnish request flow simplified



Imatge: https://www.varnish-software.com/static/book/VCL_Basics.html

Annex 3. Configuració Varnish

Donat el flux d'una crida que veiem a l'Annex 2, la configuració de Varnish es realitza a través d'un fitxer de configuració que conté un *script* en llenguatge VCL (*Varnish Configuration Language*)[9]. El VCL és un petit llenguatge dissenyat per gestionar les peticions i les polítiques de cache de les mateixes. Per fer-ho crearem l'arxiu /etc/varnish/default.vcl.

VCL és un llenguatge escrit en C i el seu estil de codificació està basat en aquest llenguatge. Ofereix un conjunt d'objectes accessibles des del cos del script de configuració i un conjunt de funcions predefinides que codifiquen el comportament de Varnish en cadascun dels estats pels que passarà o pot passar l'execució.

Les principals funcions que hem de conèixer de Varnish són:

vcl_recv: s'executa quan es rep una petició del client, possibles finals *lookup*, *pass* o *pipe*.

vcl_hash: es defineix que fa única una crida i ens permetrà saber com buscar-la a la cache, possibles finals *hash*.

vcl_hit: un cop definit quins elements fan únic un element s'executa si hem trobat aquest element a la cache, possibles finals *pass*.

vcl_miss: si no hem trobat l'element a la cache s'executa, possibles finals *pass*, *fetch*.

vcl_pass: s'executa després de vcl_recv o quan s'ha decidit que no es guardarà al cache un element, possibles finals *fetch*.

vcl_fetch: s'executa quan s'ha sol·licitat al backend que no es trobava a la cache, possibles finals *deliver*, *pass*.

vcl_deliver: s'executa quan s'ha trobat l'element a la cache abans d'enviar la resposta al client, el seu final acaba l'execució.

vcl_pipe: s'executa quan decidim que varnish no tractarà aquesta petició i la desviarà al backend, el seu final acaba l'execució.

Cada funció d'aquestes té un final que modifica l'estat de Varnish i provoca l'execució de la següent funció, com s'indica als possibles finals. Varnish té configurat un comportament per defecte que s'executarà sempre i quan no s'indiqui el contrari definint el nostre final de l'estat a cada funció.

9 The Varnish Reference Manual – VCL (https://www.varnish-cache.org/docs/3.0/reference/vcl.html)

Annex 4. Varnish default.vcl

Per defecte Varnish proporciona una configuració que és molt important tenir-la present ja que si no sobreescrivim aquesta configuració amb les nostres funcions i oferim un final alternatiu. A continuació mostrem aquesta configuració per defecte.

```
/*-
 * Copyright (c) 2006 Verdens Gang AS
 * Copyright (c) 2006-2011 Varnish Software AS
 * All rights reserved.
 *
 * Author: Poul-Henning Kamp <phk@phk.freebsd.dk>
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL AUTHOR OR CONTRIBUTORS BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
 * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
 * BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
 * WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE
 * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
 * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * The default VCL code.
 *
 * NB! You do NOT need to copy & paste all of these functions into your
 * own vcl code, if you do not provide a definition of one of these
 * functions, the compiler will automatically fall back to the default
 * code from this file.
 *
 * This code will be prefixed with a backend declaration built from the
 * -b argument.
 */

sub vcl_recv {
    if (req.restarts == 0) {
        if (req.http.x-forwarded-for) {
            set req.http.X-Forwarded-For =
                req.http.X-Forwarded-For + ", " + client.ip;
        } else {
            set req.http.X-Forwarded-For = client.ip;
```

```
        }
    }
    if (req.request != "GET" &&
      req.request != "HEAD" &&
      req.request != "PUT" &&
      req.request != "POST" &&
      req.request != "TRACE" &&
      req.request != "OPTIONS" &&
      req.request != "DELETE") {
        /* Non-RFC2616 or CONNECT which is weird. */
        return (pipe);
    }
    if (req.request != "GET" && req.request != "HEAD") {
        /* We only deal with GET and HEAD by default */
        return (pass);
    }
    if (req.http.Authorization || req.http.Cookie) {
        /* Not cacheable by default */
        return (pass);
    }
    return (lookup);
}

sub vcl_pipe {
    # Note that only the first request to the backend will have
    # X-Forwarded-For set.  If you use X-Forwarded-For and want to
    # have it set for all requests, make sure to have:
    # set bereq.http.connection = "close";
    # here.  It is not set by default as it might break some broken web
    # applications, like IIS with NTLM authentication.
    return (pipe);
}

sub vcl_pass {
    return (pass);
}

sub vcl_hash {
    hash_data(req.url);
    if (req.http.host) {
        hash_data(req.http.host);
    } else {
        hash_data(server.ip);
    }
    return (hash);
}

sub vcl_hit {
    return (deliver);
}

sub vcl_miss {
    return (fetch);
}
```

```
sub vcl_fetch {
    if (beresp.ttl <= 0s ||
        beresp.http.Set-Cookie ||
        beresp.http.Vary == "*") {
                /*
                 * Mark as "Hit-For-Pass" for the next 2 minutes
                 */
                set beresp.ttl = 120 s;
                return (hit_for_pass);
    }
    return (deliver);
}

sub vcl_deliver {
    return (deliver);
}

sub vcl_error {
    set obj.http.Content-Type = "text/html; charset=utf-8";
    set obj.http.Retry-After = "5";
    synthetic {"
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
  <head>
    <title>"} + obj.status + " " + obj.response + {"</title>
  </head>
  <body>
    <h1>Error "} + obj.status + " " + obj.response + {"</h1>
    <p>"} + obj.response + {"</p>
    <h3>Guru Meditation:</h3>
    <p>XID: "} + req.xid + {"</p>
    <hr>
    <p>Varnish cache server</p>
  </body>
</html>
"};
    return (deliver);
}

sub vcl_init {
        return (ok);
}

sub vcl_fini {
        return (ok);
}
```

Annex 5. Configuració Varnish

Configuració dels Backends dinàmics i estàtics

Abans d'entrar al flux hem de crear els backends web1 i web2 i configurarem la prova de salut amb crides a una URL creada especificiament per aquest propòsit.

```
backend web1 {
   .host = "77.73.87.225";
   .port = "80";
   .probe = {
      .url = "http://77.73.87.225/200.html";
      .interval = 5s;
      .timeout = 1s;
      .window = 5;
      .threshold = 3;
   }
}

backend web2 {
   .host = "77.73.87.226";
   .port = "80";
   .probe = {
      .url = "http://77.73.87.226/200.html";
      .interval = 5s;
      .timeout = 1s;
      .window = 5;
      .threshold = 3;
   }
}

/* Un cop definits els backeds de contingut dinàmic crearem un director per tal de que faci el balanceig utilitzant un algoritme de planificació Round Robin. */

director default_director round-robin {
   { .backend = web1; }
   { .backend = web2; }
}
```

Farem el mateix pels backends de contingut estàtic tot i que aquests cop sense director ja que decidiriem si utilitzem el static2 només en cas de fallada del static1.

```
backend static1 {
   .host = "127.0.0.1";
   .port = "8080";
   .probe = {
      .url = "http://127.0.0.1:8080/200.html";
      .interval = 5s;
      .timeout = 1s;
      .window = 5;
      .threshold = 3;
   }
}

backend static2 {
   .host = "eu.edcwb.com";
   .port = "80";
   .probe = {
      .url = "http://eu.edcwb.com/200.html";
      .interval = 5s;
      .timeout = 1s;
      .window = 5;
      .threshold = 3;
   }
}
```

Funció vcl_recv

```
sub vcl_recv {

        /*El primer que farem és la detecció d'elements de contingut estàtic per derivar-los al
        servidor de contingut estàtic. Utilitzarem l'objecte req que ens dona accés a tota la
        informació de la petició i avaluarem amb expressions regulars si la url té la forma
        us.edwb.com.*/

         if (req.http.host ~ "^(([^\s.]+[.])*us[0-9]*[.]edcwb)")
            {
              set req.backend = static1;
              # Si el servidor static no està operatiu intentarem demanar al contingut al static2
              if (!req.backend.healthy) {
                        set req.backend = static2;
              }
              # No cachejarem el contingut estàtic
              return(pass);
            } else {
              # En cas contrari establim el backend de contingut dinàmic
              set req.backend = default_director;
            }

        /* Les peticions que no siguin GET o HEAD no les cachejarem, així com tot el contingut
        sota les carpetes /client/ o /user/ i pàgines amb extensions .aspx o .ashx no ho són */
```

```
        cachejables. */

        if ((req.request != "GET" && req.request != "HEAD") ||
            req.url ~ "^/client/" ||
            req.url ~ "^/user/" ||
            req.url ~ "^.aspx" ||
            req.url ~"^.ashx")
        {
                return(pass);
        }

        /* Per últim eliminarem les Cookies de la petició ja que gràcies al script de client que
        hem desenvolupat aquestes no són necessàries */

        unset req.http.cookie;

        /* És important veure que no acavem la funció amb un return per tant en aquest cas
        s'executarà a continuació tota la lògica per defecte */

}
```

Funció vcl_fetch

```
sub vcl_fetch {

        /* Per evitar guardar les pàgines d'error a la memòria cau haurem de fer-ho des de la
        funció vcl_fetch ja que no sabem que es produirà un error fins que no reben la resposta
        del servidor. A on fixarem el temps de vida (TTL) de la pàgina a 0 per evitar que es
        guardi a la memòria cau. */

        if (beresp.status>=400 && beresp.status<600) {
                set beresp.ttl = 0s;
        }

        /* Un cop recuperada la pàgina del backend hem de decidir si la guardem al cache o no,
        i el temps de validessa de la pàgina */

        if ((req.request != "GET" && req.request != "HEAD") ||
            req.url ~ "^/client/" ||
            req.url ~ "^/user/" ||
            req.url ~ "^.aspx" ||
            req.url ~"^.ashx")
        {
                set beresp.ttl = 0s;
        } else {
                unset beresp.http.set-cookie;
                set beresp.ttl = 86400;
        }
}
```

Programari i llicències

Ubuntu 12.04.1 LTS
http://www.ubuntu.com/
http://www.ubuntu.com/project/about-ubuntu/licensing

LSYNCD 2.04
http://code.google.com/p/lsyncd/
http://directory.fsf.org/wiki/License:GPLv2

RSYNC 3.0.9
http://rsync.samba.org/
http://rsync.samba.org/GPL.html

OpenSSL 1.0.1
http://www.openssl.org/
http://www.openssl.org/source/license.html

IPTABLES 1.4.12
http://netfilter.org/
http://netfilter.org/licensing.html

NGINX 1.1.19
http://nginx.org/
http://nginx.org/LICENSE

Varnish Cache 3.0.2
https://www.varnish-cache.org/
https://www.varnish-cache.org/about

Mozilla Firefox 16.02
http://www.mozilla.org
http://www.mozilla.org/MPL/

Oracle VirtualBox 4.2.1
https://www.virtualbox.org/
https://www.virtualbox.org/wiki/GPL

LibreOffice 3.5.6.2
http://www.libreoffice.org/
http://www.libreoffice.org/download/license/

Bibliografia

Sincronització

Content Delivery Network: a poor man's guide
https://maze.io/2010/10/12/content-delivery-network-a-poor-mans-guide/index.html

HOWTO build your own open source Dropbox clone
http://fak3r.com/2009/09/14/howto-build-your-own-open-source-dropbox-clone/

HOWTO: Passwordless ssh logins
http://fak3r.com/2006/08/10/howto-passwordless-ssh-logins/

Passwordless SSH
http://usefulubuntu.blogspot.com.es/2009/03/passwordless-ssh.html

Lsyncd 2.0.x ‖ Layer 4 Config ‖ Default Behavior
https://github.com/axkibe/lsyncd/wiki/Lsyncd%202.0.x%20%E2%80%96%20Layer%204%20Config%20%E2%80%96%20Default%20Behavior

Utilize Lsyncd to sync multiple web servers
http://www.thisisnotsupported.com/lsyncd/

Varnish Cache

The Varnish Book
Tollef Fog Heen (Varnish Software), Kristian Lyngstøl (Varnish Software), Jérôme Renard (39Web) Documentation version-4.4 / Tested for Varnish 3.0.2 Varnish Software AS 2010-2012, Redpill Linpro AS 2008-2009
https://www.varnish-software.com/static/book/

Varnish configuration step by step
http://www.slideshare.net/kimlindholm/varnish-configuration-step-by-step

Varnish caching proxy configuration resources
http://nwlinux.com/varnish-caching-proxy-configuration-resources/

Varnish HTTP Accelerator + ASP.NET + IIS
http://www.keebler.net/blog/2009/02/01/varnish-http-accelerator-aspnet-iis/

Varnish HTTP Accelerator configuració i arrencada
http://www.gnutoolbox.com/varnish-http-accelerator-cpanel/

IPTABLES

25 Most Frequently Used Linux IPTables Rules Examples
http://www.thegeekstuff.com/2011/06/iptables-rules-examples/

Munin

Easy Monitoring of Varnish with Munin
http://blog.gomiso.com/2011/01/04/easy-monitoring-of-varnish-with-munin/

UML

Teach Yourself UML in 24 hours Second Edition Autor: Joseph Schmuller Ed. Sams Plubishing. Any 2002

Optimització Web

Web Performance Optimization – Javier Casares
http://javiercasares.com/wpo/

Man Pages

curl(1) – Linux man page
http://linux.die.net/man/1/curl