Filtre de Pàgines Web

José Manuel Barbero Moral
ETIG

Consultor: David Carrera Pérez

31/12/2004

2 Dedicatòria i agraïments

A la meva dona que ha aguantat el meu mal humor, les nits de soledat veient la televisió mentre jo em rebentava les poques neurones que em queden davant el tft.
Al meu germà i la meva mare que els he dit “no” moltes vegades quan m’han demanat un favor per que o bé estava en un estat de concentració que més valia a aprofitar o simplement no estava d’humor.
Al meu gat que moltes vegades em fa companyia silenciosament segurament preguntant-se que carai faig tantes hores assegut davant l’ordinador.

3 Resum

Aquest projecte es bassa en l'estudi del protocol http, un protocol que funciona amb l'estàndard client-servidor, la seva estructura, funcionament, explicació d'un exemple i el tipus de dades que maneguen les peticions i respostes. i la seva aplicació en el cas del servidors proxy.

Es presenten tres productes comercials dels quals s'explica una mica les seves característiques. A continuació es comenta el procés de creació d'una eina dissenyada per mi: el Web-Filter. Un proxy que permet fer de filtre en una xarxa local d'ordinadors i que a més guarda un registre de totes les seves connexions. Comença amb les especificacions del funcionament demanades , els esquemes i diagrames de funcionament i classes necessaris i comentaris sobre les diferents funcions i pantalles que incorpora el programa.

4 Índex de continguts

4.1 Índex de figures

5 Cos de la Memòria

5.1 Introducció

Un proxy es una eina que serveix per filtrar connexions entre un ordinador client i un altre servidor. Treballa a nivell de protocols. En el nostre cas es tracta del protocol HTTP.
Hi ha diferents tipus de proxys que treballen amb un sol protocol o més d'un a la vegada. També poden realitzar tasques diferents. Els proxys anònims serveixen per poder navegar per Internet sense que es pugui saber el nostre origen. Altres s'encarreguen de filtrar el contingut que es pot redirigir als clients, es a dir, impedir que es mostrin determinats continguts segons uns criteris establerts.

5.1.1 Justificació i context

La seva utilitat s'enfocaria en una xarxa local d'ordinadors als que es vol impedir el flux de tràfic indesitjat. En realitat seria quelcom semblant a un tallafocs però que es configura d'una manera diferent. En lloc de filtrar segons les IP's ho fa per continguts específics.

5.1.2 Objectius

Poder filtrar els continguts que es mostren als navegadors dels ordinadors clients segons els termes definits als filtres del programa.
Es tracta de desenvolupar un programa amb una interfície grafica que permeti fer un filtrat de totes les pagines web que passin pel navegador. Aquest filtre tindrà les següents funcions:

- Bloquejar l'acces a una pagina web que tingui un cert tipus de contingut basat en paraules clau definides per l'usuari en la interfície.
- Bloquejar l'acces a una pagina web concreta definida per l'usuari en la interfície.
- Crear un fitxer amb la informació relativa a la navegació. Aquest fitxer contindrà el nom de les webs visitades, la IP del client, l'hora i la data en que s'ha demanat la connexió, i si aquesta ha estat permesa.

5.1.3 Enfocament i mètode seguit

Amb l'avantatge de aconseguir una utilitat que sigui portable a diferents plataformes i a més per aprofitar la potència innata d'un llenguatge creat especialment per actuar en entorn de xarxa he optat per implementar el codi en JAVA, concretament la versió 1.4.2.

El sistema en que es crearà l'aplicació serà windows XP, encara essent en JAVA es podria implementar en Linux perfectament.
L'entorn de programació escollit ha sigut el JCreator, un editor de JAVA gratuït molt senzill d'utilitzar i que permet visualitzar el codi, la llista de classes creades al projecte, un índex dels mètodes de la classe que estem visualitzant i a mes permet d'incorporar un compilador, i també permet executar el codi.
Respecte al compilador he optat per incorporar un mes desenvolupat que el que porta el JCreator per defecte. Aquest es el JIKES en la seva versió 1.22 també gratuït.

1.4 Planificació del projecte

El pla de treball seguit ha sigut el següent:

Pla de Treball:

En aquest document es detallen el seguit de tasques a realitzar per tal d'implementar un programa en java que s'encarregara de filtrar les pagines web a les que puguin accedir amb el navegador, a mes de fer uns log que enregistri les webs i el temps de connexió.

Descripció del TFC

Es tracta de desenvolupar un programa amb una interfície gràfica que permeti fer un filtrat de totes les pagines web que passin pel navegador. Aquest filtre tindrà les següents funcions:

- Bloquejar l'accés a una pagina web que tingui un cert tipus de contingut basat en paraules clau definides per l'usuari en la interfície.
- Bloquejar l'acces a una pagina web concreta definida per l'usuari en la interfície.
- Crear un fitxer amb la informació relativa a la navegació. Aquest fitxer contindrà el nom de les webs visitades i el temps de connexió.

Tasca 1:

Temporització: 15 dies (23 de setembre al 8 d'octubre).
Descripció: Documentació de les necessitats i aspectes que ha de resoldre el programa.

- Objectius del programa
- Funcions generals (funcions que ofereix el programa)
- Usabilitat del programa (disseny de la interfície grafica del programa)
- Explicació bàsica de funcionament del programa a partir de la interfície grafica.

Tasca 2:

Temporització: 4 setmanes (9 d'octubre al 7 de novembre).
Descripció: Disseny de la interfície grafica i esquemes de funcionament.

- Explicació a nivell profund del funcionament del programa (relacions entre classes, diagrames de funcionament del programa, etc.)
- Diagrama i disseny basic de les classes

Tasca 3:

Temporització: 6 setmanes (8 de novembre al 19 de desembre).
Descripció: Implementació de les classes a partir del disseny obtingut en la tasca anterior.

Tasca 4:

Temporització: 1 setmana (del 19 al 26 de desembre).
Descripció: Documentació del programa:

Tasca 5:

Temporització: 2 setmanes (del 27 de desembre al 9 de gener).
Descripció: preparació la presentació del filtre de pagines web.

5.1.5 Productes obtinguts

Tots son pel sistema operatiu Windows.

Freeproxy: es un proxy que permet connectar diversos ordinadors a Internet a partir d'un únic punt de connexió, aquest es el proxy. Permet guardar un log de les connexions creades, te un filtre URL i marcació automàtica de connexió d'acord a un calendari determinat.

Abcproxy 2004: permet connectar diversos ordinadors a Internet a partir d'un únic punt de connexió. Incloeix un firewall. Esta preparat per connexions RTB, RDSI/ISDN, Frame Relay, Cable, ADSL/xDSL (àdhuc amb mudem USB), etc., i permet compartir serveis com la navegació per Internet, HTTP, HTTPS, FTP, e-mail, POP, SMTP, News, IRC, Chat, MSN Messenger, Media Player, SOCKS, etc.

MyProxy 6.51: MyProxy es un servidor proxy per a serveis HTTP, HTTPS, SMTP i POP3 que a més permet filtrar publicitat, finestres de pop-up altres funcions interessants. Gracias a la seva capacitat de filtrat de publicitat, pots navegar mes ràpid, una velocitat que també es veu incrementada amb la funció de grabació local de DNS, el que estalvia tràfic de xarxa. Com a servidor proxy, permet compartir una única connexió a Internet entre dos o més ordinadors. El programa controla l'existencia de possibles elementos de adware o spyware i impedeix l'us no autoritzat de la connexió mitjançant contrasenya.

5.2 Anàlisi i disseny de Web-Filter

5.2.1 Introducció al protocol HTTP

El Protocol de Transferència de HiperText (Hypertext Transfer Protocol) es un protocol client-servidor que manega els intercanvis d'informació entre els clients Web i els servidors HTTP. L'especificació completa del protocol HTTP versions 1.0, i 1.1 estan recollides en les RFC 1945 i 2616. Va ser proposada per Tim Berbers-Lee, per atendre les necessitats del sistema global de distribució d'informació que compren el Word Wide Web.

Des del punt de vista de les comunicacions, està suportat sobre els serveis de connexió TCP/IP, i funciona de la mateixa manera que la resta dels serveis comuns de l'entorn UNIX: un procés servidor escolta en un port de comunicacions TCP(per defecte el 80), i espera les sol·licituds de connexió dels clients Web. Una vegada que s'estableix la connexió, el protocol TCP s'encarrega de mantenir la comunicació i garantir un intercanvi de dades lliure d'errors.

HTTP es basa en senzilles operacions de sol·licitud/resposta. Un client estableix una connexió amb un servidor i envia un missatge amb les dades de la sol·licitud. El servidor respon amb un missatge similar, que conte l'estat de l'operació i el seu possible resultat. Totes les operacions poden adjuntar un objecte o recurs sobre el que actuen; cada objecte Web (document HTML, fitxer multimedia o aplicació CGI) es conegut per la seva URL.

Els recursos o objectes que actuen com a entrada o sortida d'un comando HTTP estan classificats per la seva descripció MIME. D'aquesta forma, el protocol pot intercanviar qualsevol tipus de dada, sense preocupar-se del seu contingut. La transferència es realitza en mode binari, byte a byte, i la identificació MIME permetrà que el receptor tracti adequadament les dades.

En primer lloc exposaré les característiques principals de la versió 1.0 del protocol HTTP, i desprès parlaré de les innovacions fetes a la versió 1.1.

Protocol HTTP 1.0:

- Tota la comunicació entre los clients i servidors es realitza a partir de caràcters de 8 bits. D'aquesta forma, es pot transmetre qualsevol tipus de document: text, binari, etc., respectant el seu format original.
- Permet la transferència d'objectes multimedia. El contingut de cada objecte intercanviat està identificat per la seva classificació MIME.
- Existeixen tres verbs bàsics (hi ha més, però generalment no s'utilitzen) que un client pot utilitzar per dialogar amb el servidor: GET, per a recollir un objecte, POST, per a enviar informació al servidor i HEAD, per a sol·licitar les característiques d'un objecte (per exemple, la data de modificació d'un document HTML).
- Cada operació HTTP implica una connexió amb el servidor, que es alliberada al terme de la mateixa. Es a dir, en una operació es pot recollir un únic objecte.

- No manté estat. Cada petició d'un client a un servidor no es influïda per les transaccions anteriors. El servidor tracta cada petició com una operació totalment independent de la resta.
- Cada objecte al que s'apliquen els verbs del protocol està identificat a través de la informació de situació del final de la URL.

HTTP es va dissenyar específicament pel World Wide Web: es un protocol ràpid ì senzill que permet la transferència de múltiples tipus d'informació de forma eficient i ràpida. Es pot comparar, per exemple, amb el FTP, que també es un protocol de transferència de fitxers, però te un conjunt molt ampli de comandos, i no s'integra massa be en les transferències multimedia.

Les característiques que diferencien l'HTTP 1.0 de l'HTTP 1.1 son:

- Respostes més ràpides, degut a que permet múltiples transaccions en una simple connexió persistent.
- Respostes més ràpides degut a que economitza amplada de banda utilitzant suport de cache.
- Respostes més ràpides per pagines generades dinàmicament, degut al suport de *chunked encoding*, que permet que una resposta sigui enviada sense conèixer la seva longitud total.
- Us eficient de les adreces IP, permetent que múltiples dominis siguin servits en la mateixa IP.

5.2.2 Etapes d'una transacció HTTP en la versió 1.0

Quan un client realitza una petició a un servidor, s'executen els següents passos:

1. L'usuari accedeix a una URL, seleccionant un enllaç d'un document HTML o introduint directament en el camp Location del client Web.
2. El client Web descodifica la URL, separant les diferents parts. Així identifica el protocol d'accés, la direcció DNS o IP del servidor, el possible port opcional (el valor per defecte es 80) i l'objecte requerit del servidor.
3. S'obre una connexió TCP/IP amb el servidor, cridant al port TCP corresponent.
4. Es realitza la petició. Per això, s'envia la comanda necessari (GET, POST, HEAD,…), la direcció de l'objecte requerit (el contingut de la URL que segueix a la direcció del servidor), la versió del protocol HTTP empleada (quasi sempre HTTP/1.0) i un conjunt variable d'informació, que incloeix dades sobre les capacitats del browser, dades opcionals pel servidor,…
5. El servidor torna la resposta al client. Consisteix en un codi d'estat i el tipus de dada MIME de la informació de retorn, seguit de la pròpia informació.
6. Es tanca la connexió TCP.

Aquest procés es repeteix en cada accés al servidor HTTP. Per exemple, si es recull un document HTML que conte inserides quatre imatges, el procés anterior es repeteix cinc vegades, una pel document HTML i quatre per les imatges.

Diferències amb la versió 1.1

En l'actualitat s'ha millorat aquest procediment, permeten que una mateixa connexió es mantingui activa durant un cert període de temps, de forma que sigui utilitzada en successives transaccions. Aquest mecanisme, denominat HTTP Keep Alive, es empleat per la majoria dels clients i servidors moderns. Aquesta millora es imprescindible en una Internet saturada, en la que l'establiment de cada nova connexió es un procés lent i costós.

Per veure millor com treballa el protocol HTTP, posaré un exemple d'un cas típic de una transacció HTTP i desprès analitzaré les parts del procés.

Des d'un client es demana la URL http://www.uoc.edu
Obrim una connexió TCP/IP amb el port 80 del sistema www.uoc.edu.
El client fa la sol·licitud, enviant una petició típica:

-- GET /index.html HTTP/1.0
GET es una comanda per fer una petició d'un objecte. En aquest cas index.html que seria la pagina principal del servidor. HTTP/1.0 indica la versió del protocol HTTP.

-- Accept: text/plain

-- Accept: text/html

-- Accept: audio/*

-- Accept: video/mpeg
A continuació ve la llista de tipus MIME que accepta o entén.

-- Accept: */*

Això vol dir que accepta altres possibles tipus MIME.

-- User-Agent: Mozilla/4.0 (compatible; MSIE 6.0, windows NT 5.1)

Aquesta línia conte informació sobre el tipus de client

Línia en blanc, indica el final de la petició

A continuació posem un exemple de la resposta del servidor:

-- HTTP/1.0 200 OK
Status de la operació; en aquest cas, correcte

-- Date: Friday, 3-December-2004 10:18:23
Data i hora de l'operació

-- Server: NCSA 1.4
Indica el tipus i versió del servidor

-- MIME-version: 1.0

Indica la versió de MIME que manega

-- Content-type: text/html
Definició MIME del tipus de dades de la resposta

-- Content-length: 254
Indica la longitud de les dades que venen a continuació

-- Last-modified: 03-December-2004 6:00:00 GMT
Data i hora de modificació de les dades

Línia en blanc

<HTML>

<HEAD><TITLE>.. .. .. </TITLE></HEAD>

<BODY>
.. .. ..
.. .. ..
</HTML>

Això ja son les dades de la resposta.

Desprès de cada línia tant de petició com de respostes va un CRLF que es representa amb els caràcters \r\n. Quan hi ha una línia en blanc es posen dos CRLF.
El cos de les dades es tracta d'una manera especial i no porta cap CRLF.

Desprès d'això es tanca la connexió.

5.2.3 Mètodes del servei HTTP

L'estàndard HTTP/1.0 defineix únicament tres mètodes que representen les operacions de recepció i enviament d'informació i control d'estat:

GET - S'utilitza per obtenir la URL especificada en la línia de petició. EL servidor normalment tradueix el camí de l'URL a un nom de fitxer o de programa:

- En el primer cas el cos de l'entitat serà el contingut del fitxer.
- En el segon cas el servidor executarà el programa i l'entitat serà el resultat que generi.

Els components *paràmetre* i/o *consulta* de l'URL es poden utilitzar com a arguments del programa.

HEAD - Sol·licita informació sobre un objecte (fitxer): tamany, tipus, data de modificació… Es bàsicament igual que el GET però en la resposta el cos es buit, i per tant, només hi haurà la capçalera. Normalment es utilitzat pels gestors de caus de pagines o els servidors proxy, per conèixer quan es necessari actualitzar la copia que es manté d'un fitxer.

POST - Serveix per enviar informació al servidor que aquest incorporarà al recurs identificat per l'URL de la línia de petició. L'operació que es realitza amb la informació proporcionada depèn de la URL utilitzada. S'utilitza, sobre tot, en els formularis.

A la versió 1.1 del HTTP hi ha a més d'aquestes una seria de mètodes que s'han afegit Per dotar de major potencia i flexibilitat als servidors:

PUT – serveix per crear un recurs amb l'URL especificat en la petició.
DELETE - Elimina el document especificat del servidor.
OPTIONS – serveix per obtenir informació sobre les opcions de transferència.
TRACE – serveix per a obtenir una copia del missatge com ha arribat a la seva destinació final.

5.2.4 Les capçaleres

Els missatges HTTP porten un conjunt de variables amb la missió de modificar el seu comportament o incloure informació d'interès. En funció del seu nom, pot aparèixer en els requeriments d'un client, en les respostes del servidor o en ambdós tipus de missatges. El format general d'una capçalera es:

Nom de la variable
:
Cadena ASCII amb el seu valor

Els noms de variables es poden escriure amb qualsevol combinació de majúscules i minúscules. A més, s'ha de incloure un espai en blanc entre el signe : i el seu valor. En cas de que el valor d'una variable ocupi varies línies, aquestes hauran de començar, al menys, amb un espai en blanc o un tabulador.

Els camps que hi pot haver en la capçalera d'un missatge HTTP es poden classificar en quatre grups:

Camps generals que poden ser presents tant en els missatges de petició com en els de resposta.

Date: data local de l'operació. Les dates ha d'incloure la zona horària en que resideix el sistema que genera l'operació. Per exemple: Sunday, 12-Dec-96 12:21:22 GMT+01. No existeix un format únic en les dates; àdhuc es possible trobar casos en els que no es disposa de la zona horària corresponent, amb els problemes de sincronització que això produeix. Els formats de data que s'emplean estan recollits en els RFC 1036 i 1123.
Pragma: permet incloure informació variada relacionada amb el protocol HTTP en el requeriment o resposta que s'està realitzant. Per exemple, un client envia un Pragma: no-cache per informar de que desitja una copia nova del recurs especificat.

Camps referents a l'entitat continguda en el cos del missatge i que també poden ser presents en peticions i respostes:

Content-Type: descripció MIME de la informació continguda en aquest missatge. Es la referència que utilitzen les aplicacions Web per donar el correcte tractament a les dades que reben.
Content-Length: longitud en bytes de les dades enviades, expressat en base decimal.
Content-Encoding: format de codificació de les dades enviades en aquest missatge. Serveix, per exemple, per enviar dades comprimides (x-gzip o x-compress) o encriptades.
Last-Modified: data. Indica la data i l'hora en que el recurs contingut en el cos de la resposta es va modificar per última vegada.
Expires: data. Indica la data i l'hora a partir de la qual el contingut del cos de la resposta es pot considerar obsolet o caducat a l'efecte del seu emmagatzematge en la memòria cau. La presència d'aquest camp significa que , possiblement, el recurs es modificarà en la data indicada o deixarà d'existir, però no implica que els canvis, si es produeixen, s'hagin de fer necessàriament en aquesta data. Ara bé, si la data de caducitat és igual o anterior a la que hi ha especificada en al camp *Date*, l'entitat no s'ha d'emmagatzemar en la memòria cau.
Allow: informa de los comandos HTTP opcionals que es poden aplicar sobre l'objecte al que es refereix aquesta resposta. Per exemple, Allow: GET, POST.

Camps propis de les peticions HTTP /1.0

Accept: camp opcional que conte una llista de tipus MIME acceptats pel client. Es pot utilitzar * per indicar ran gos de tipus de dades; tipus/* indica tots els subtipus d'un determinat medi, mentre que */* representa a qualsevol tipus de dada disponible.
Authorization: clau d'accés que envia un client per accedir a un recurs d'us protegit o limitat. La informació incloeix el format d'autorització empleada, seguit de la clau d'accés pròpiament dita. L'explicació s'incloeix més endavant.
From: camp opcional que conte la direcció de correu electrònic de l'usuari del client Web que realitza l'accés.
If-Modified-Since: permet realitzar operacions GET condicionals, en funció de si la data de modificació de l'objecte requerit es anterior o posterior a la data proporcionada. Pot ser utilitzada pels sistemes d'emmagatzematge temporal de pàgines. Es equivalent a realitzar un HEAD seguit d'un GET normal.
Referer: conté URL del document des d'on s'ha activat aquest enllaç. D'aquesta forma, un servidor pot informar al creador d'aquell document de canvis o actualitzacions en los enllaços que conte. No tots els clients l'envien.
User-agent: cadena que identifica el tipus i versió del client que realitza la petició.

Camps propis de les respostes HTTP 1.0

Server: Aquest camp és semblant a *User-Agent*, però referit al servidor.
Location: informa sobre la direcció exacta del recurs al que s'ha accedit. Quan el servidor proporciona un codi de resposta de la sèrie 3xx, aquest paràmetre conte la URL necessària per accessos posteriors a aquest recurs.
Server: cadena que identifica el tipus i versió del servidor HTTP. Per exemple, Server: NCSA 1.4.
WWW-Autenticate: quan s'accedeix a un recurs protegit o d'accés restringit, el servidor retorna un codi d'estat 401, i utilitza aquest camp per informar dels models d'autentificació vàlids per accedir a aquest recurs.

Camps afegits a la versió 1.1 del protocol HTTP

En el cas del grup 1º s'afegeixen cinc camps generals:

Cache-Control: directrius sobre la política de memòria cau.
Connnection: opcions de connexió. L'opció *close* indica que l'emissor tancarà la connexió després que s'hagi tramès la resposta.
Transfer-Encoding: codificació aplicada al cos del missatge.
Upgrade: versions del protocol suportades.
Via: intermediaris pels quals ha passat el missatge.

En el cas del 2º grup s'afegeixen sis nous camps:

Content-Base: adreça base per interpretar URL relatius.
Content-Languaje: llenguatge que s'ha d'emprar.
Content-Location: URI de l'entitat, en el cas que el recurs corresponent en tingui més d'una.
Content-MDS: seqüència de bits per comprovar la integritat del contingut.
Content-Range: per si una entitat s'envia en diferents fragments.
Etag: Etiqueta associada a l'entitat, per si el recurs en té més d'una.

En el cas del grup 3º s'afegeixen dotze camps:

Accept: tipus de contingut que el client pot acceptar.
Accept-Charset.
Accept-Encoding.
Accpet-Languaje.
Host: nom del servidor a qui va adreçada la petició, per si en té més d'un; aquest camp, obligatori en les peticions HTTP/1.1, permet que un ordinador amb diferents noms actuï com si fos diferents servidors alhora.
If-Match: permet comparar entitats per les seves etiquetes.
If-None-Match.
If-Range.
If-Unmodified-Since.
Max-Forwards.
Proxy-Authorization.
Range: serveix per a sol·licitar un fragment d'una entitat.

En el cas de grup 4º s'afegeixen sis camps més:

Age.
Proxy-Authenticate.
Public: llista de mètodes suportats pel servidor.
Retry-After.
Vary: llista de camps de la petició que s'han d'utilitzar per a seleccionar una entitat, quan el recurs en té més d'una.
Warning: informació addicional sobre l'estatus de la resposta.

Codis d'estat del servidor

Per a cada transacció amb un servidor HTTP, aquest retorna un codi numèric que informa sobre el resultat de l'operació, com a primera línia del missatge de resposta. Aquests codis apareixen en alguns casos en la pantalla del client, quan es produeix un error. El format de la línia d'estat es:

- Versió de protocol HTTP utilitzada
- Codi numèric d'estat (tres dígits)
- Descripció del codi numèric

Depenent del servidor, es possible que es proporcioni un missatge d'error més elaborat, en forma de document HTML, en el que s'expliquen les causes de l'error i la seva possible solució.

Els més comuns son els següents:

Codi	**Comentari**	**Descripció**
200	OK	Operació realitzada satisfactòriament.
201	Created	L'operació ha sigut realitzada correctament, i com a resultat s'ha creat un nou objecte, amb l'URL d'accés continguda al cos de la resposta. Aquest nou objecte ja està disponible. Pot ser utilitzat en sistemes d'edició de documents.
202	Accepted	L'operació ha sigut realitzada correctament, i com a resultat s'ha creat un nou objecte, amb l' URL d'accés continguda al cos de la resposta. El nou objecte no està disponible de moment. En el cos de la resposta s'ha d'informar sobre la disponibilitat de la informació.
204	No Content	L'operació ha sigut acceptada, però no s'ha produït cap resultat d'interès. El client no haurà de modificar el document que s'està mostrant en aquest moment.
301	Moved Permanently	L'objecte al que s'accedeix ha sigut mogut a un altre lloc de forma permanent. El servidor proporciona, a més, la nova URL en la variable Location de la resposta. Alguns browsers accedeixen automàticament a la nova URL. En cas de tenir capacitat, el client pot actualitzar l'URL incorrecta, per exemple, en l'agenda de bookmarks.
302	Moved Temporarily	L'objecte al que s'accedeix ha sigut mogut a un altre lloc de forma temporal. El servidor proporciona, a més, la nova URL en la variable Location de la resposta. Alguns browsers accedeixen automàticament a la nova URL. El client no ha de modificar cap de las referències a l'URL errònia.

304	Not Modified	Quan es fa un GET condicional, i el document no ha sigut modificat, es retorna aquest codi d'estat.
400	Bad Request	La petició te un error de sintaxi i no es entesa pel servidor.
401	Unauthorized	La petició requereix una autorització especial, que normalment consisteix en un nom i clau que el servidor verificarà. El camp WWW-Autenticate informa dels protocols d'autentificació acceptats per aquest recurs.
403	Forbidden	Està prohibit l'accés a aquest recurs. No es possible utilitzar una clau per modificar la protecció.
404	Not Found	L'URL sol·licitada no existeix.
500	Internal Server Error	El servidor ha sofert un error intern, i no pot continuar amb el processament.
501	Not Implemented	El servidor no te capacitat, pel seu disseny intern, per portar a terme el requeriment del client.
502	Bad Gateway	El servidor, que està actuant com a proxy o passarel·la, ha trobat un error al accedir al recurs que havia sol·licitat el client.
503	Service Unavailable	El servidor esta actualment deshabillitat, i no es capaç d'atendre el requeriment.

Caus de pàgines, passarel·les i servidors proxy

Molts clients Web utilitzen un sistema per a reduir el nombre d'accessos i transferències d'informació a través d'Internet, i així agilitar la presentació de documents prèviament visitats. Per això, emmagatzemen en el disc del client una copia de les darreres pàgines a les que s'ha accedit. Aquest mecanisme, denominat "cau de pàgines", manté la data d'accés a un document i comprova, a través d'una comanda HEAD, la data actual de modificació del mateix. En cas que es detecti un canvi o actualització, el client accedirà, ara a través d'un GET, a recollir la nova versió del fitxer. En cas contrari, es procedirà a utilitzar la copia local.

Un sistema semblant, però amb més funcions es el denominat "servidor proxy".
En la configuració amb proxy, el client estableix la connexió amb el servidor proxy que, al seu torn, estableix una altra connexió amb el servidor final. Quan el proxy rep el missatge del client, pot generar una resposta pròpia o retransmetre la petició al servidor final. En el segon cas, el proxy pot introduir modificacions en la petició, segons

l’aplicació per la qual estigui dissenyat, i quan rebi la resposta del servidor final, la envia al client, també amb la possibilitat de fer-hi canvis.

Es possible connectar més d’un proxy en cadena.

La principal diferència entre enviar una petició a un proxy o fer-ho a un servidor es que quan es fa a un proxy la línia de petició no ha de ser un RRL relatiu , sinó que ha de ser un URL absolut. Altrament, el proxy no pot saber qui és el servidor final a qui va destinat la petició.

Les avantatges que poden tenir aquests dos sistemes son:

La principal avantatge de ambdós sistemes es la dràstica reducció de connexions a Internet necessàries, en cas de que els clients accedeixin a un conjunt similar de pàgines, com passa amb molta freqüència. A més, determinades organitzacions limiten, per motius de seguretat, els accessos des de la seva organització l’exterior i viceversa. Per això, es disposa de sistemes denominats "tallafocs" (firewalls), que son els únics habilitats per connectar-se amb l’exterior. En aquest cas, l’ús d’un servidor proxy es torna indispensable.

En determinades situacions, l’emmagatzematge de pàgines en una cau o en un proxy pot fer que es mantinguin copies no actualitzades de la informació, com per exemple en el cas de treballar amb documents generals dinàmicament. Per aquests situacions, els servidors HTTP poden informar als clients de l’expiració del document, o de la impossibilitat de ser emmagatzemat en una cau, utilitzant la variable Expires en la resposta del servidor.

5.2.5 Objectius del programa

Es tracta de desenvolupar un programa amb una interfície gràfica que permeti fer un filtrat de totes les pagines web que passin pel navegador segons unes regles determinades i a més permeti examinar les connexions que s'han produït.

Base teórica

La teoría básica en que es fundamenta aquest programa es que quan naveguem per Internet a traves de les pagines web usant el protocol www estem rebent informació al nostre ordinador a traves del port 80. Els paquets de informació que donaran lloc a les pagines que presenta el navegador en pantalla es poden recollir amb un programa que escolti el port 80 i enviï els aquests mateixos paquets al navegador. El que farà el programa es tractar aquests paquets abans d'enviar-los al navegador. Els haurà d'analitzar en busca de les coincidències necessàries perquè es compleixin els criteris de filtrat que hem definit al engegar el programa. Els paquets que es considerin que compleixen les regles de filtrat no seran enviats al navegador. Apart d'aixo el programa enregistrarà en un log totes les connexions pel port 80 guardant les dades corresponents a nom de l'adreça i IP, data i hora de la connexió i un altre camp que farà referencia a si aquesta connexió ha sigut filtrada o no.

5.2.6 Instal·lació i configuració del programa Web-Filter

El programa es bàsicament un proxy, per tant es pot connectar en un ordinador i navegar a través del proxy des del mateix ordinador o conectarlo en un ordinador en una xarxa local per a que doni suport a més ordinadors (a alguns o a tots els de la xarxa local). Per això hem de configurar els navegadors dels ordinador que vulguin navegar a traves del proxy. En el cas del Windows XP hem de clissar l'opció del menú superior "Herramientas / Opciones de Internet / Conexiones / Configuración de LAN", i una vegada aquí marcar la casella "Servidor Proxy". Desprès obrir en "Opciones Avanzadas", i a la casella del protocol HTTP posar l'adreça del ordinador on està instal·lat el proxy (en el cas que sigui el mateix es posa 127.0.0.1) i el nombre de port 8080, i fer "Acceptar".

El programa consta de quatre classes en JAVA. Una vegada compilades s'ha d'executar la classe "Gestor". A partir d'aquí surten les pantalles i el funcionament es l'explicat a l'apartat dedicat e les pantalles.

5.2.7 Funcions generals

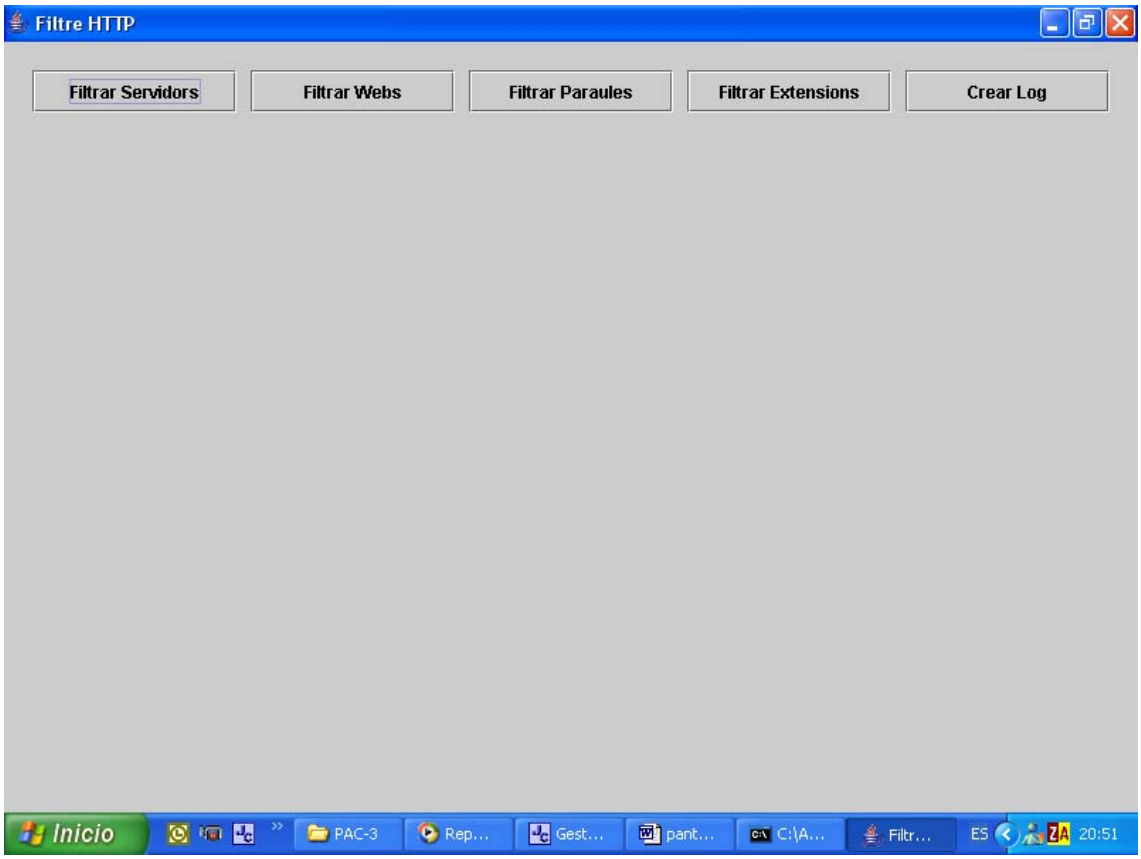
. Aquest filtre tindrà les següents funcions:

- Bloquejar l'acces a una pagina web que tingui un cert tipus de contingut basat en paraules clau definides per l'usuari en la interfície.
- Bloquejar l'acces a una pagina web concreta definida per l'usuari en la interfície.
- Bloquejar l'acces a un servidor concret definit per l'usuari en la interfície.
- Bloquejar l'entrada en l'ordinador de arxius via web amb una extensió definida per l'usuari.

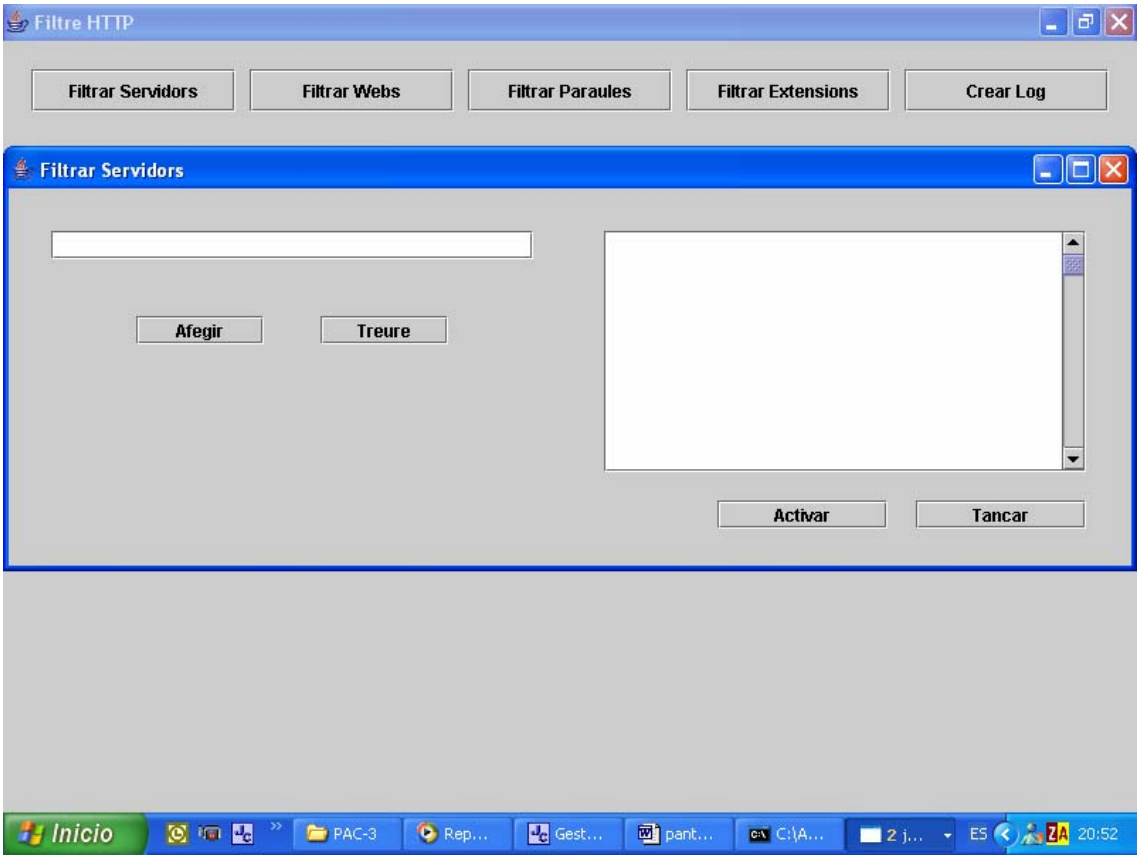
- Crear un fitxer amb la informació relativa a la navegació. Aquest fitxer contindrà el nom de les webs visitades i la data i hora de la connexió.

5.2.8 Disseny de la interfície grafica i funcionament del programa

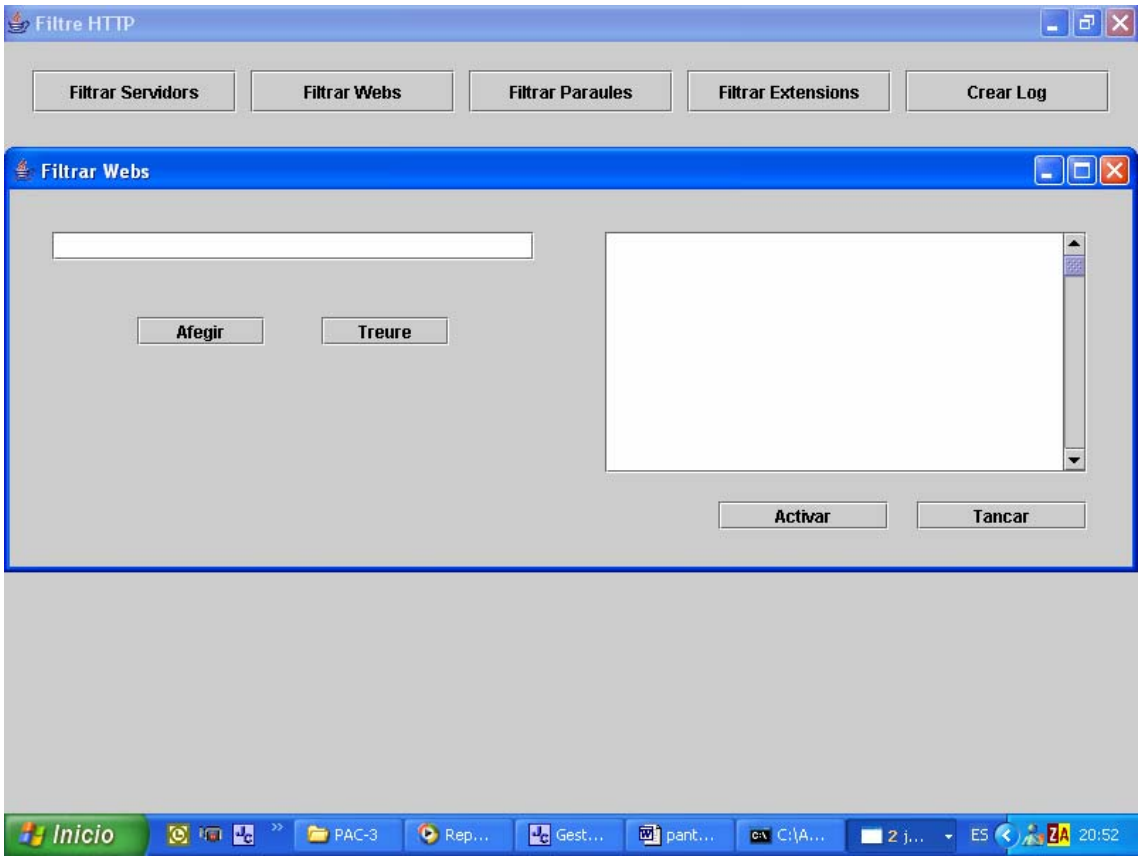
Per començar he de dir que el programa te un nom, es dirá Web-Filter.
Al iniciarse el programa surt una primera pantalla que es un menú amb cinc opcions. Cada opció dona accés a una pantalla. Les primeres quatre pantalles es on es configuren la totalitat de les opcions de filtrat del programa. Cada opció se pot activar o desactivar independentment de les altres.
Amb la cinquena s'accedeix a la llista de connexions.
Les primeres cinc pantalles (els filtres) tenen un model de disseny i una mecànica de funcionament molt semblant per facilitar la seva interacció amb l'usuari.



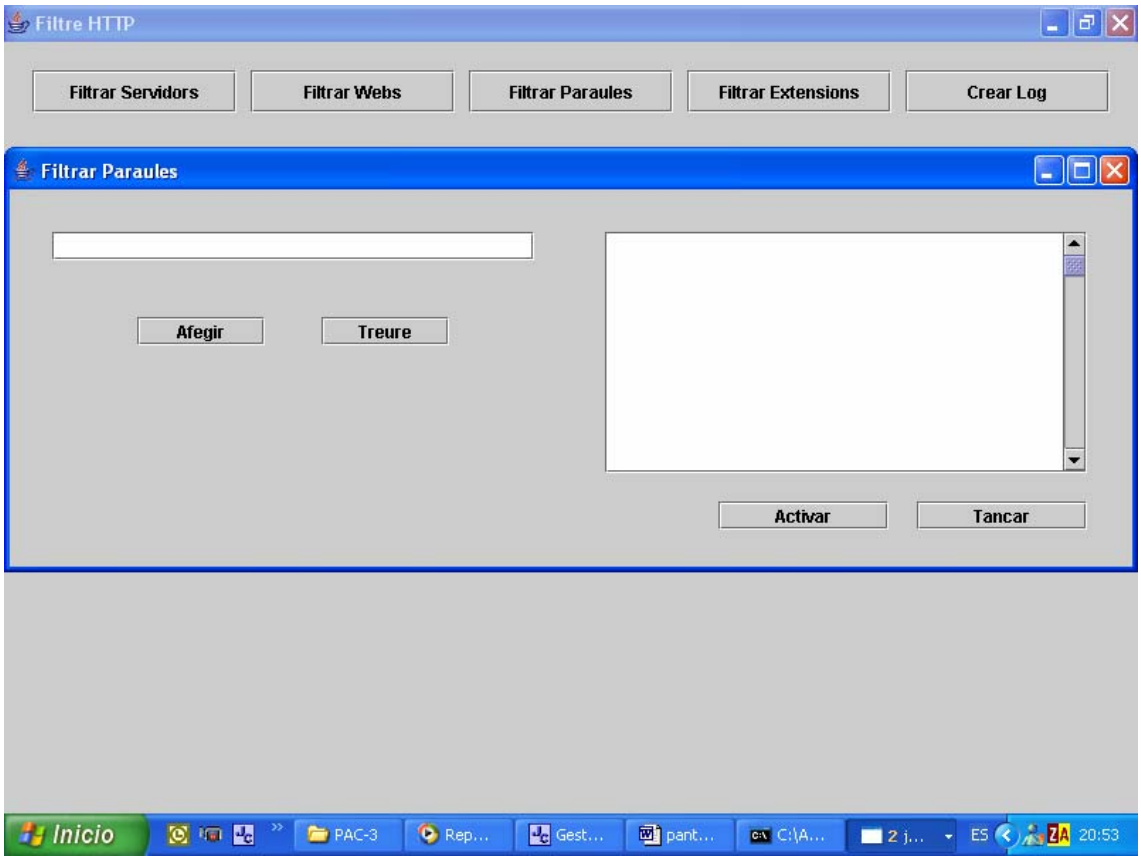
A la primera opció del menú posa la paraula “Servidors”. Aquesta pantalla dona la possibilitat de bloquejar l’accés als servidors.
Al camp on s’ha d’introduir el nom del servidor es pot posar el nom o la IP del servidor per afegir a la llista de la dreta. També està el botó per treure de la llista les que no volem que es bloquessin.



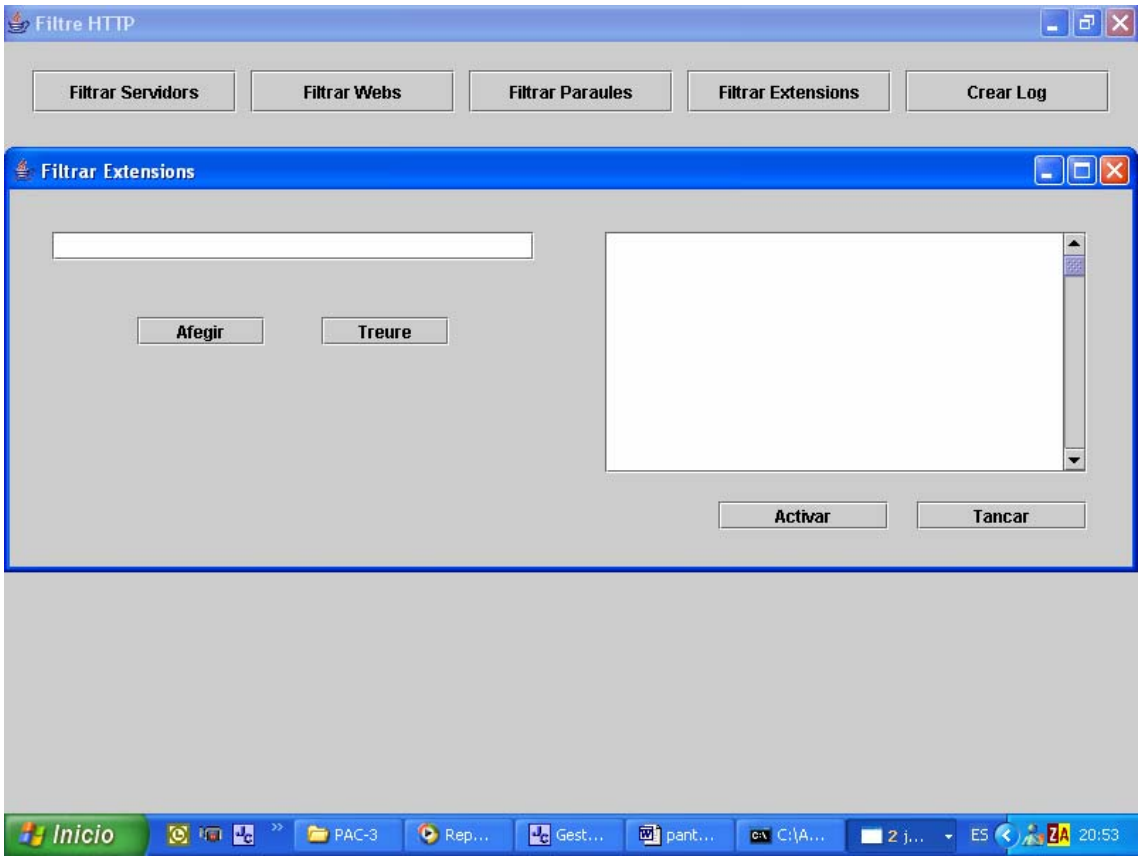
A la segona opció del menú posa la paraula “Webs”. En aquesta pantalla s’activa el filtre de webs.
Hi ha un camp on s’ha d’introduir el nom de la web per afegir a la llista de webs que seran bloquejades.
També hi ha un botó per treure de la llista les que no volem que es bloquessin.



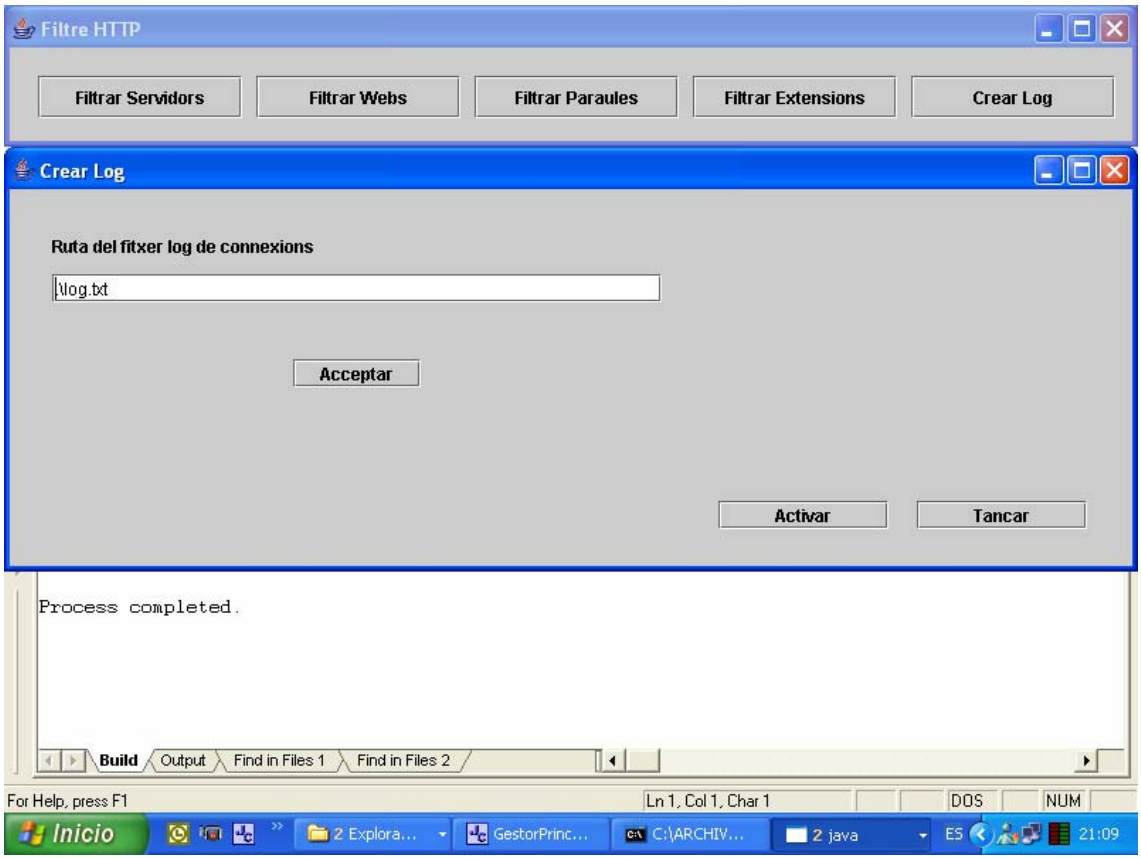
La tercera opció es diu “Paraules”. Hi ha un camp on s’introdueix una paraula i al clissar el botó afegir la paraula s’afegeix a la llista de paraules que hi ha a la dreta. Aquesta llista representa totes les paraules que seran filtrades. A mes es pot decidir si la paraula afegida es filtrara solament al títol de la web o també al contingut d’aquesta.
El botó de treure fa justament el contrari. Escrivim una paraula al camp i al prémer “Treure” aquesta desapareix de la llista de la dreta.
Al prémer el botó “Activar” es posa en funcionament el filtre corresponent a la pantalla. Llavors canvia la llegenda del botó i posarà “Desactivar”.



La quarta opció del menú correspon al filtre de arxius. Es fa mitjançant el tipus d'arxiu, es a dir, tenint en compte la seva extensió. Hi ha un camp on s'ha d'introduir la extensió per afegir a la llista de la dreta.
També hi ha un botó per treure de la llista les que no volem que es bloquessin.

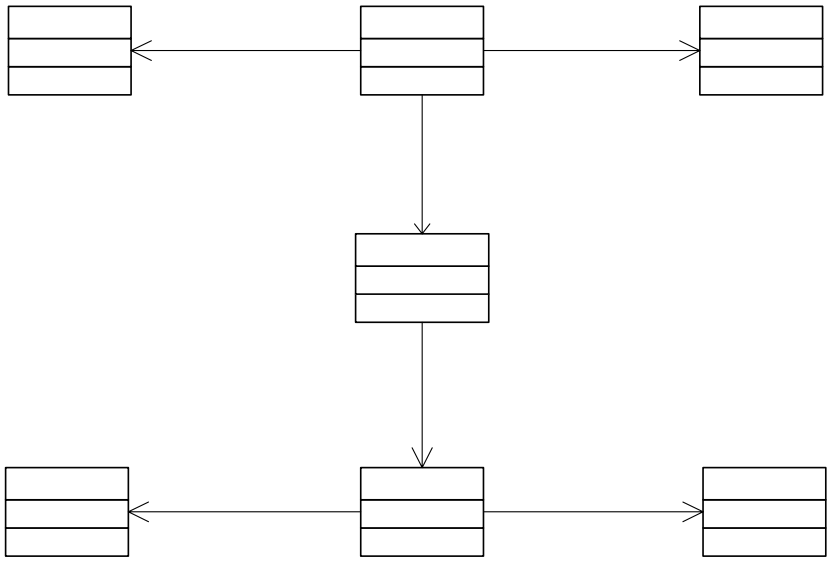


La cinquena i ultima opció del menú no es cap filtre sino que dona accés a les connexions via http que es fan en l'ordinador. Hi ha un camp per introduir la ruta del fitxer log on es guardaran les dades corresponents.



Si fem una ullada al log corresponent podem veure les dades de connexions, de totes les connexions, tant filtrades com no filtrades. A l'esquerra del nom de la web s'indica si s'ha efectuat el bloqueig. A continuació surten la IP a que corresponen, la data i l'hora

5.2.9 Diagrama UML de classes



La classe principal que inicia el programa es el Gestor. Aquesta classe te les següents responsabilitats:

- Crear el GestorPantalla.
- Crear els diferents filtres.
- Escoltar la xarxa i crear les connexions "Conex" quan detecti una que pertanyi al protocol http.
- Crear el log, encara que no escriu res a dins

Crear el GestorPantalla

A l'executar-se el Gestor aquest crea el GestorPantalla. Aquesta classe s'encarrega de crear la PantallaMenu. El motiu que es creï una classe com aquesta sols per crear un altre es que ho faci amb un procés paral·lel, per a que pugui haver més d'un fil d'execució. Mentre s'activen les pantalles segueix funcionant la resta del programa.

Crear els diferents filtres

Al principi havia pensat crear una classe per a cada filtre que heretessin d'una superclasse anomenada filtre. Desprès em vaig adonar que no calia fer diferents classes

ja que bàsicament tos els filtres tenen els mateixos mètodes i les mateixes variables, per tant he definit quatre instàncies de la classe Filtre per crear cada u dels quatre.

Escoltar la xarxa i crear les connexions

La classe Gestor es la que conté a més el bucle principal d'execució que ha d'escoltar la xarxa i quan es detecta una connexió corresponent al protocol http capta els paquets i els fa passar pels diferents filtres a més de crear el log.
De fet no fa tot això, sino que solament escolta la xarxa, el fet de captar el tràfic http es automàtic ja que al estar definit el programa com un proxy http solament li arribarà aquest tipus de paquets.

Una vegada detecta una connexió http crea una instancia de la classe Conex que la que en realitat gestiona tota la connexió.

Crear el log.

El programa crea un log amb la informació de totes les connexió http que entren al sistema, tant si passen els filtres com si no (això s'indica en el log). La classe Gestor no escriu res a dins , solament el crea. L'estructura del log es un fitxer d'extensió "txt" amb els camps:
Origen (host del client), Destí (host del servidor), Data (data i hora de la connexió) i un camp que diu si ha sigut filtrada o no la connexió:

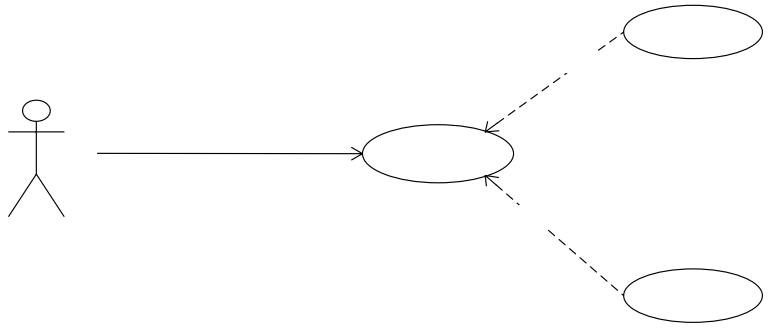
Funcionament de la classe Conex

La classe Conex es la que manega una connexió concreta una vegada creada. Bàsicament s'encarrega de rebre la petició del client, enviar-la al servidor, rebre la resposta i pasar-la pels filtres. Depenent de si passa els filtres o no envia una resposta de "Pagina filtrada" o la resposta del servidor al client. A més d'això s'encarrega de registrar les dades de la connexió al log.

El funcionament de les classes Pantalla

La classe PantallaMenu que ha sigut creada pel GestorPantalla presenta el menú de les cinc opcions del programa.
Al apartat que parla de les pantalles ja s'ha explicat com funcionen. Apart d'això afegir que a nivell de programari cada PantallaFiltre el que fa es activar o desactivar el filtre corresponent, però aquest ja ha sigut creat pel Gestor.

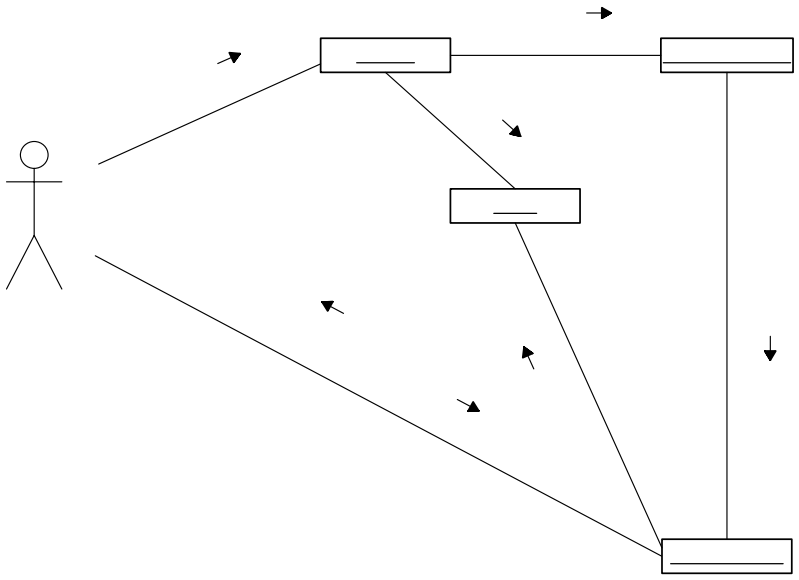
5.2.10 DIAGRAMA DE CASOS D'US



Aquest esquema representa tots els tipus d'ús que pot donar un usuari al programari. El cas "Filtrar connexió" es l'objectiu principal del programari encara que l'usuari no ho fa expressament ja es el programa que efectua la seva tasca en detectar que l'usuari inici una connexió amb el protocol http.
Els altres dos casos que son extendits del principal representen la configuració de les opcions del programa segons les quals ha de funcionar.

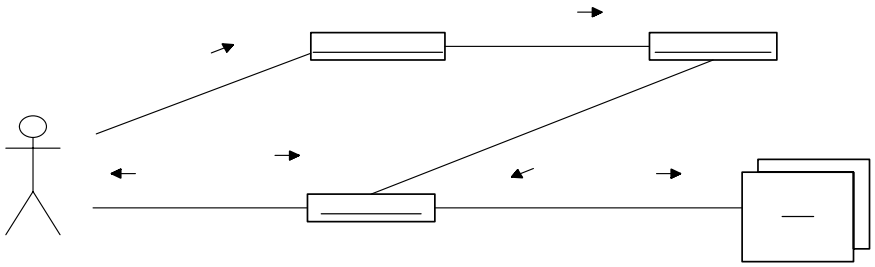
5.2.11 DIAGRAMES DE SEQUENCIA I COLLABORACIO

Activar filtre



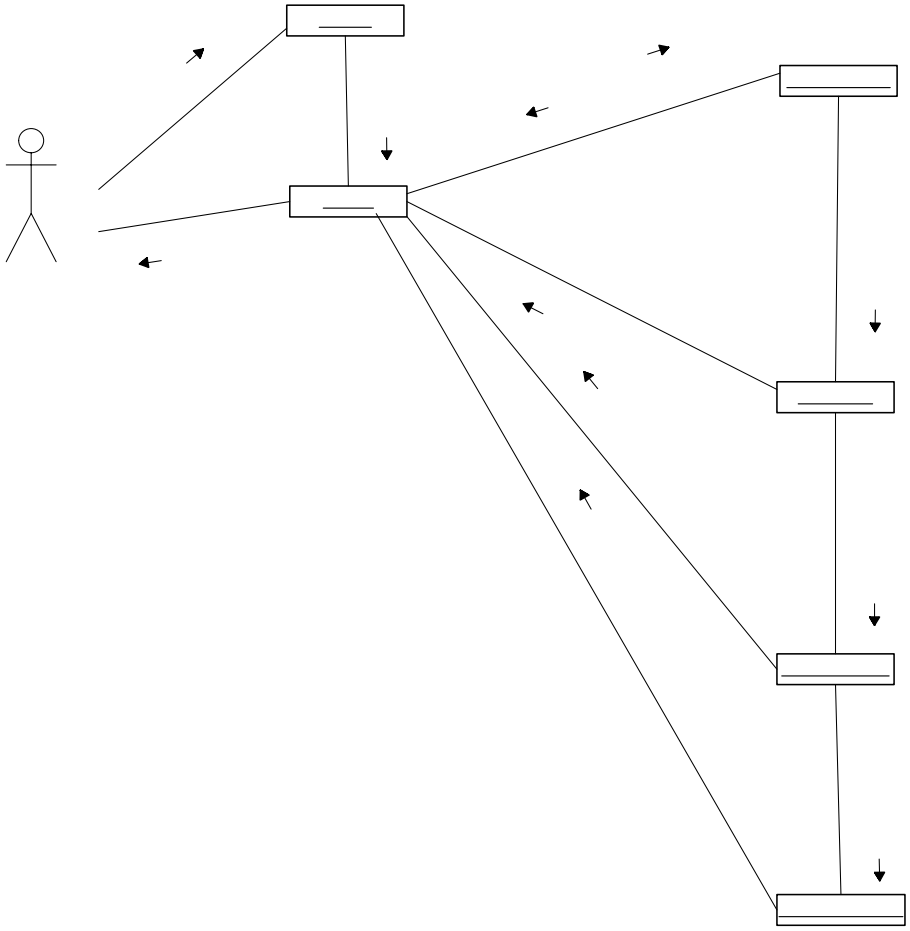
Aquest diagrama representa al cas d'ús "Activar filtre" que s'encarrega de rebre les dades de l'usuari per omplir les opcions dels diferents filtres, registrades en la classe *Filtre* i engegar-lo mitjançant el botó d'activació d ela pantalla del filtre.

Activar log



En aquest diagrama on es representa al cas d'ús "Activar log", l'usuari ha d'introduir el directori on vol que aparegui el fitxer log i posteriorment activar-lo o no amb el botó corresponent.

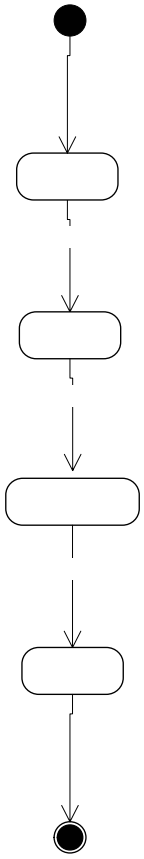
Filtrar connexió



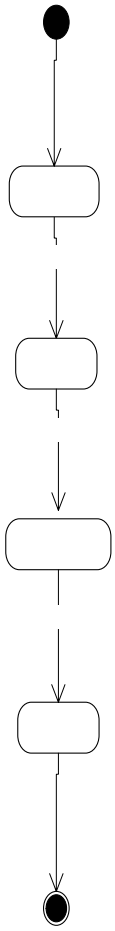
Aquest es el procediment principal del programa una vegada configurades les dues opcions anteriors. Com ja he dit anterior ment el programa no inicia el filtrat al arrencar el programa sinó al detectar una connexió entrant amb el protocol http.

5.2.12 DIAGRAMES D'ESTATS

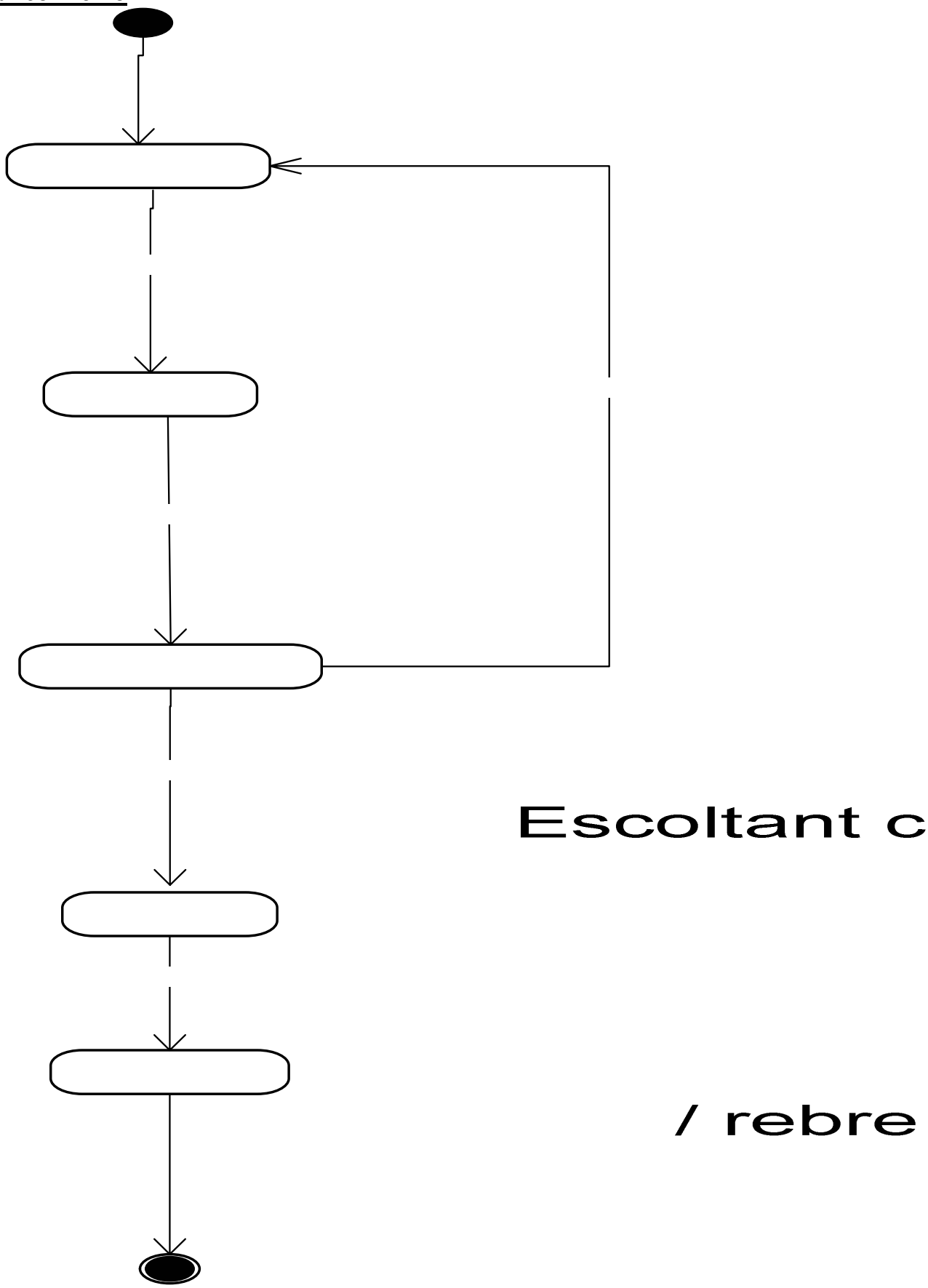
Activar filtre



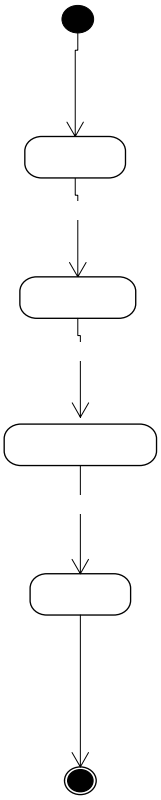
Activar log



Filtrar connexió



Crear log



5.3 Conclusions

La realització del programa Web-Filter en principi havia de basar-se en el funcionament d'un proxy en protocol http. Al principi vaig perdre massa temps ja que la meva idea original era basar-lo en un resolver DNS i desprès de aclarir conceptes amb el consultor vaig tenir que canviar els meus plantejaments inicials. Llavors vaig començar des de zero l'estudi del protocol http i la seva aplicació en els proxys.

M'he adonat que el programa podria incorporar un major control sobre les respostes a les peticions dels clients respecte a la permissió o denegació de connexions i a la seva notificació mitjançant missatges als clients. Però crec que aquest no era l'objectiu prioritari del programa ja que de fet havia de fer un filtre per pàgines web, així que m'he limitat a llegir les respostes dels servidors, passar-les pels filtres i enviar-les als clients tal qual.

D'altre banda he après molt sobre el protocol http del que no sabia gairebé res. Bàsicament que hi ha dos versions: la 1.0 i la 1.1. Que la segona versió estalvia un gran nombre de connexions i ajuda a fer més ràpides la càrrega de pàgines web.

6 Glossari

CGI: Veure: "Common Gateway Interface".

Common Gateway Interface : CGI (Interfaz Comú de Pasarella): Interfaç d' intercanvi de dades estàndar en WWW a través del qual s'organitza l'envament i la recepció de dades entre navegador i programes residents en servidors WWW

Client: Un sistema o procés que sol·licita a un altre sistema o procés que li presti un servei. Una estació de treball que sol·licita el contingut d'un fitxer a un servidor de fitxers es un client d'aquest servidor. Veure també: "client-server model", "server".

CRLF:

Cookie: Conjunt de caràcters que s'emmagatzema en el disc dur o en la memòria temporal de l'ordinador d'un usuari quan accedeix a les pàgines de determinats llocs web. S'utilitzen per a que el servidor accedit pugui conèixer les preferències de l'usuari. Donat que poden ser un perill per la intimitat dels usuaris, aquests han de saber que els navegadors permeten desactivar els cuquis.

Firewall: Encaminador amb capacitat de filtrat de paquets d'acord amb unes regles establertes. Aquestes regles es defineixen en cada instal·lació d'acord amb els problemes de seguretat de la xarxa en que s'instal·la.

HTML: Veure "HyperText Markup Language".

HTTP: Veure "HyperText Transmission Protocol".

HyperText Markup Language: (Llenguatje de Marcatje d'Hipertext):
Llenguatge en el que s'escriuen les pàgines a les que s'accedeix a través de navegadors WWW. Admet components hipertextuals i multimedia.

HyperText Transfer Protocol : (Protocol de Transferència d'Hipertext):
Protocol utilitzat per la transferència de documents WWW.

IP address (direcció IP): Direcció de 32 bits definida pel Protocol Internet en STD 5, RFC 791. Es representa usualment mitjançant notació decimal separada per punts.

Internet Protocol (Protocol Internet): Conjunt de regles que regulen la transmissió de paquets de dades a través d'Internet. La versió actual es IPv4 mentre que en el projecte Internet2 s'intenta implementar la versió 6 (IPv6), que permetria millors prestacions dins del concepte QoS (*Quality of Service*).

JAVA: Llenguatge de programació desenvolupat per Sun per l'elaboració de petites aplicacions exportables a la xarxa (*applets*) i capaços d'operar sobre qualsevol plataforma a través, normalment, de navegadors WWW. Permet donar dinamisme a les pàgines web.

Log: Fitxer normalment de text on s'enregistren dades generades per les operacions efectuades per un programa concret. Usualment correspon a programes relacionats amb connexions de xarxa.

MIME: Veure: "Multipurpose Internet Mail Extensions".

Multipurpose Internet Mail Extensions : (Extensions Multipropósit del Correu Internet): Conjunt d'especificacions Internet de lliure distribució que permet tant l'intercanvi de text escrit en llenguatges amb diferents jocs de caràcters com el correu multimedia entre ordinadors i aplicacions que segueixen els estàndards de correu Internet. Les especificacions MIME estan recollides en nombroses RFCs, entre les que es troben els RFC1521 i 1848.

Protocol client-servidor: Model per a representar aplicacions locals en què es defineix una part servidora (que és qui proporciona els serveis) a la qual accedeix una part client seguint les indicacions d'un usuari humà a través d'una interfície d'usuari, o una altra aplicació.

Proxy: Servidor que rep peticions del client, les retransmet a un servidor remot, en rep les respostes i les torna al client, sovint després d'aplicar-li algun procés o transformació.

RFC (Request For Documents): conjunt de documents numerats que cobreixen diverses àrees tècniques, incloses les pròpies bases de funcionament d'Internet.

TCP/IP: (Transmission Control Protocol/Internet Protocol) Protocol de Control de Transmissió/Protocol Internet. Sistema de protocols, definits en RFC 793, en els que es basa bona part d'Internet. El primer s'encarrega de dividir la informació en paquets en origen, perquè després es recomponguin en el destí, mentre que el segon es responsabilitza de dirigir-la adequadament a través de la xarxa

Thread: fil d'execució d'un programa que funciona paral·lelament al fil principal d'execució o a altres threads.

UNIX: Sistema operatiu interactiu i de temps compartit creat en 1969 per Ken Thompson. Rescrit a meitat de la dècada dels '70 per ATT va aconseguir una enorme popularitat en els ambients acadèmics, i més tard en els empresarials, com un sistema obert, robust, flexible i portable, molt utilitzat en els entorns Internet.

URI: Veure: "Uniform Resource Locator/Uniform Resource Identifier".

URL: Veure: "Uniform Resource Locator/Uniform Resource Identifier".

Uniform Resource Locator/Universal Resource Identifier : (Localitzador Uniforme de Recursos/Identificador Universal de Recursos): Sistema unificat d'identificació de recursos en la xarxa. Les direccions es componen de protocol, FQDN i direcció local del document dins del servidor. Aquest tipus de direccions permet identificar objectes WWW, Gopher, FTP, News, ...

7 Bibliografia

Referències a pàgines web

Apuntes de Lenguage JAVA:
http://www.arrakis.es/~abelp/ApuntesJava/indice.htm

HTMLWEB: Sockets en JAVA:
http://www.htmlweb.net/articulos/sock_java.html

El Protocolo HTTP:
http://cdec.unican.es/libro/HTTP.htm

Tutorial de JAVA – Servidor HTTP:
http://www.itapizaco.edu.mx/paginas/JavaTut/froufe/parte20/cap20-14.html

JAVA en Castellano – Firewall en JAVA
http://www.programacion.com/java/codigo/31/

JAVA en castellano – Programación en red
http://www.programacion.com/java/tutorial/joa_red/7/

PC paso a paso – Los cuadernos de hackxcrack:
http://www.hackxcrack.com/

Linux 24x7 – Funcionamiento de l protocolo HTTP:
http://www.24x7linux.com/documentation/internet/how_http_works.shtml.es

Tutorial de JAVA – tabla de contenidos:
http://www.cica.es/formacion/JavaTut/Intro/tabla.html

Tutorial de JAVA: streams de entrada:
http://www.cica.es/formacion/JavaTut/Cap8/str_ent.html

Tutorial de JAVA - cliente HTTP:
http://www.itapizaco.edu.mx/paginas/JavaTut/froufe/parte20/cap20-10.html

The JAVA tm tutorial – Trail: custom networking:
http://java.sun.com/docs/books/tutorial/networking/index.html

Planet Source Code:
http://www.planet-source-code.com/vb/scripts/ShowCode.asp?txtCodeId=4561&lngWId=2

La web del programador:
http://www.lawebdelprogramador.com/news/new.php?id=44&texto=Java

JAVA en castellano – Foros:
http://www.programacion.com/java/foros/6/

HTTP Made Really Easy
A Practical Guide to Writing Clients and Servers
http://www.jmarshall.com/easy/http/

Distributed Systems Lab 2004 - Hypertext Transfer Protocol (HTTP) Tutorial:
http://www.dslab.tuwien.ac.at/Task_Description/http.html

Glosario básico inglés-español para usuarios de Internet:
http://www.ati.es/novatica/glosario/glosario_internet.txt

Documents en PDF

Aprenda JAVA como si estuviera en primero

Thinking in JAVA 3ªEdition

RFC 1945

RFC 2616