Installazione
Apache - Tomcat
Indice

1 Introduzione al manuale

Il presente documento costituisce lo stato dell'arte per l'installazione degli applicativi weboriented
che utilizzano sistemi basati su archietttura **APACHE** e **TOMCAT**
La parte relativa all'installazione dell'ambiente di base è rivolta essenzialmente ad i sistemisti ed
agli specialisti di prodotto con adeguato skill che intendano cimentarsi nell' installazione iniziale.
Tale sezione descrive le modalità di installazione dei web-server ***Apache*** e ***Tomcat***, le strategie
di configurazione e le motivazioni di tali scelte.

2 Apache e Tomcat

2.1 Perché Utilizzare Apache e Tomcat

2.1.1 Vantaggi dell'utilizzo congiunto di Apache e Tomcat

Questa sezione del documento descrive come configurare Apache per effettuare il dispatching
di richieste JSP e servlets verso due o più server Tomcat in ascolto su porte diverse.

Apache è un web server ormai standard ed affidabile che eccelle soprattutto per la gestione di
contenuti statici, come pagine ed immagini statiche HTML.
Tomcat è per contro un servlet/JSP container aderente alle specifice Java Servlet e JavaServer
Pages (Tomcat4 Servlet 2.3 e JSP 1.2, Tomcat5 Servlet 2.4 e JSP 2.0) quindi più performante
nella gestione di Java Server Pages e servlets, mentre è decisamente inferiore ad Apache nella
gestione di contenuti statici.
La combinazione dei due, connessi tramite il connettore AJP, consente di costruire architetture
eterogenee e di ottenere vantaggi considerevoli:
1) Specializzazione: uso di un web server dedicato a questa attività invece di sfruttare Tomcat
anche come http server. Peraltro sembra che dopo un certo numero di sessioni attive contemporaneamente, le prestazioni di Tomcat regrediscono.
2) Bilanciamento: un web server come Apache è in grado di fornire un servizio di ***load balancer***. Secondo algoritmi di DNS o round robin (anche semplici ma abbastanza efficaci) è
quindi in grado di assegnare le chiamate applicative a turno su server applicativi che stanno a
valle. Un unico IP di accesso dall'esterno può quindi essere front end per più di un applicativo
che ne sta a valle.
3) Sicurezza: Apache Web server potrà anche essere utilizzato al di fuori della rete aziendale
e quindi posizionato in internet senza particolari problemi, lasciando al mondo solo la funzionalità di Web server mentre l'applicazione e naturalmente la base dati sono ben protetti
da firewall appositi dentro la rete aziendale. In questo modo le uniche vie di accesso all'applicazione dal web server sono una porta per ogni applicativo.
4) Un ambiente come Apache fornisce facilmente la possibilità di usare protocolli protetti utilizzando protocollo SSL e encryptioning.
Le configurazioni descritte nelle figure successive danno una idea tipica per un ambiente sicuro Intranet.

Figura 1 *- L'utilizzo di un connettore AJP permette di mettere un server HTTP di fronte a Tomcat: in questo modo lo si può isolare, rendendo più sicura l'architettura complessiva. Il server HTTP inoltre offre maggiori potenzialità di configurazione ed ampliamento*

Oltre a questo, avere un solo punto di entrata permette di realizzare architetture in cluster, dove a fronte di un solo indirizzo internet di entrata si possono associare più server Tomcat in
maniera del tutto trasparente e scalabile.

Figura 2 *- L'utilizzo di un connettore AJP consente di associare ad un solo server HTTP (quindi un solo indirizzo internet) più server Tomcat, dando vita in modo semplice ad una configurazione cluster facilmente scalabile e quindi capace di servire un numero maggiore di clienti.*

Per connettere Apache e Tomcat sono disponibili diverse alternative; quella prescelta nel nostro
caso é:

- **mod_jk**: questo connettore si basa sul protocollo AJP 1.2 (Apache Jakarta Protocol) o AJP 1.3.

mod_jk realizza il bilanciamento di carico mediante session affinity: quando un browser richiede

una pagina JSP per la prima volta, il load balancer ridirige la richiesta ricevuta da Apache ad
uno dei server Tomcat. Le ulteriori richieste originate dalla stessa sessione client saranno inviate
automaticamente alla stessa istanza di Tomcat.
Questa scelta di fondo non preclude la possibilità di utilizzare altri metodi di installazione: per esempio in ambito Microsoft, il servizio di load balancer di windows
2003 permette di ottenere un bilanciamento del carico direttamente da ambiente TOMCAT facendo riferimento ad un unico IP address. In questo caso non si ottengono
tutti i vantaggi descritti.

2.1.2 Topologia

E' possibile proporre casi diversi di topologia di installazione che possono dipendere anche da
valutazioni non strettamente tecnologiche, p.e. economiche, logistiche, etc…

2.1.2.1 Ambiente Tomcat

unico motore Tomcat ma Web Application diverse: la configurazione permette un risparmio in termini di amministrazione ma ha la controindicazione che nel caso di crash di Tomcat si bloccano tutte le applicazioni;

più motori Tomcat diversi su una stessa macchina: se si blocca uno dei Tomcat gli utenti continuano a lavorare grazie ai meccanismi di bilanciamento di carico, ma un problema generale della macchina blocca tutti gli utenti. Inoltre, come nel caso precedente, il carico grava su un unico hardware;

più motori Tomcat diversi su diverse macchine: gli utenti continuano a lavorare anche se una delle macchine si ferma.

2.1.2.2 Ambiente Apache

Il server apache che si occupa del bilanciamento è bene che sia installato su un sistema anch'esso replicato. In alternativa, può essere installato:

su una delle macchine su cui è installato il Tomcat: in questo caso se si ferma la macchina in questione comunque nessuno lavora più. Inoltre, questa stessa macchina ha un sovraccarico legato al lavoro del http server;

su una singola macchina dedicata: soluzione migliore ma più costosa. Il server Apache in DMZ dovrebbe comunque essere messo su una macchina dedicata altrimenti i criteri di sicurezza decadono.

2.1.3 Software necessario per l'installazione

Le componenti necessarie per effettuare l'installazione sono le seguenti:

- **Apache 2.x**
- **Modulo jk: connettore apache-Tomcat**
- **Java Virtual Machine 1.4.2_xx**
- **Tomcat 4.1.x comunque non inferiore alla 4.1.27**

Le installazioni verranno fatte mediante i files binari, sia su Windows che su Unix/Linux. Per
quest'ultimo è possibile eventualmente, scaricare i sorgenti e ricompilarli

2.2 Installazione di Apache

Per ottimizzare Apache, se il server dovrà servire esclusivamente come bilanciatore per Tomcat,
è possibile abilitare solo i moduli necessari all'installazione: questo può avere un aumento di
performance, soprattutto su sistemi ricompilati.

Link al sito da cui scaricare Apache 2.x
http://httpd.apache.org/download.cgi
A seconda del sistema operativo utilizzato, la versione di Apache viene scaricata in uno dei
seguenti formati:
- eseguibile per Windows
- tar (o zip) per Unix e Linux

2.2.1 Installazione di Apache su Windows

E' sufficiente lanciare l'eseguibile e seguire le indicazioni del setup, lasciando i default proposti
ed avendo cura di installare Apache come servizio Windows in ascolto sulla porta standard (80);

2.2.2 Installazione di Apache su Unix

Occorre effettuare i seguenti passi:
1. scaricare il file httpd-2.0.53.tar.gz da http://httpd.apache.org/download.cgi (questo e' un sorgente indipendente dalla piattaforma unix)
2. loggarsi con utente root
3. creare il gruppo apache
4. creare l'utente apache
5. eseguire questi comandi
cd /tmp
mkdir apache
cd apache
6. portare il file con ftp in binario sul server in **/tmp/apache** ed eseguire
gunzip httpd-2.0.53.tar.gz
tar tvf httpd-2.0.53.tar
```
# ./configure –prefix=/usr/local/apache (o altro path installazione)
# make
```
make install
7. impostare correttamente i seguenti parametri nel file **conf/httpd.conf**
Listen IP:porta
Servername nome:porta
User apache
Group apache
8. per far partire Apache lanciare **bin/apachectl start**
9. per stoppare Apache lanciare **bin/apachectl stop**
10. questi comandi vanno inseriti negli script di boot del server

2.2.3 Installazione di Apache su Linux Red-Hat 3.0 SE

Se il sistema operativo del server sul quale si vuole installare Apache è il Linux Red-Hat 3.0 SE,
è possibile che Apache sia già installato in quanto previsto dall'installazione completa del sistema operativo. In questo caso i precedenti punti da 1 a 6 possono essere sostituiti dai seguenti:
1. Verificare se esiste gia un Apache installato con il seguente comando:
find / -name httpd –print
dovrebbe visualizzarsi il seguente path:
/etc/httpd
nel caso fosse installato anche Oracle 9.2 verrà visualizzata una directory al di sotto della home di Oracle, in quanto anche Oracle installa una versione di Apache, tuttavia non la useremo. Per vedere su quali directory è effettivamente installato si può usare il comando
rpm –q httpd –all | more
che lista tutte le directory ricorsivamente anche se ci sono link virtuali

2. Se esiste verificarne la versione con il seguente comando:
httpd –v
una qualsiasi versione **2.x** va bene.
3. Verificare se e' attivo con il seguente comando:
ps –ef | grep httpd
se Apache è attivo verranno visualizzate più righe del tipo
/usr/sbin/httpd –k
in caso di prima installazione su server nuovo è più probabile che Apache sia installato ma non attivo, per avviarlo occorre posizionarsi nella directory
/usr/sbin
e lanciare il comando
./apachectl start
se prima non si è configurato l'indirizzo IP a cui deve rispondere, Apache visualizzerà il seguente messsaggio e risponderà all'indirizzo IP del server sulla porta di default 80.
httpd: Could not determine the server's fully qualified domain name, using 192.168.35.22 for ServerName
4. Verificare se funzionante colegandosi con un browser all'indirizzo http:*//sever:porta* (porta:
se omessa va sulla 80) se viene correttamente visualizzata la pagina iniziale di Apache ed è
una versione recente e correttamente funzionate possiamo utilizzarla.
5. A questo punto non resta che fare in modo che Apache si avvii automaticamente al boot, per
ottenere questo è consigliabile utilizzare il gestore grafico dei servizi ponendo il check avanti
al servizio httpd come in figura sotto

2.3 Installazione di mod_jk: connettore apache-Tomcat

2.3.1 Installazione di mod_jk su Windows-Unix-Linux

2.3.1.1 Configurazione Jk2 Module in httpd.conf

Si effettua un unico passo:
Sotto la direttiva
"# LoadModule foo_module modules/mod_foo.so"
inserire la linea
LoadModule jk2_module modules/mod_jk2.so
Non si fanno altri riferimenti al worker.properties o a contesti tomcat nel file **httpd.conf**.

2.3.1.2 Configurazione JK Module in httpd.conf

Occorre modificare il file di configurazione di Apache, **httpd.conf** nel seguente modo :
1. Sotto la direttiva **# LoadModule foo_module modules/mod_foo.so**
inserire le linee seguenti :

```
# Load mod_jk
#
LoadModule jk_module modules/mod_jk.so
#
# Configure mod_jk
#
JkWorkersFile conf/workers.properties
JkLogFile logs/mod_jk.log
JkLogLevel info
#
```

2. Sotto la linea **DocumentRoot** inserire le due linee seguenti :

JkMount /*.jsp loadbalancer
JkMount /servlet/* loadbalancer
Inserire anche il contesto applicativo che si utilizzerà, ad esempio:
JkMount /psglib/* loadbalancer
JkMount /jbf/* loadbalancer

Questo significa che tutti le pagine che iniziano con /psglib/… e tutte le pagine che iniziano con
/jbf/… non devono essere cercate su Apache ma devono essere richieste al loadbalancer che le
cercherà sui web-server tomcat che andremo a definire nel file seguente. Se su tale installazione vorremmo mettere anche XMPI e ADTWEB dovremmo aggiungere anche le 2
seguenti righe:

JkMount /Sianc/* loadbalancer
JkMount /ADTWEB/* loadbalancer

2.3.1.3 Creazione del file workers.properties per Jk

Occorre quindi creare il file worker.properties, riferito nel file di configurazione httpd.conf e metterlo sotto la directory conf, allo stesso livello di httpd.conf.
Il file worker.properties configura per Apache i vari server Tomcat ed indica le loro porte di ascolto.
Nell'esempio riportato, i due Tomcat server sono installati in una macchina diversa da quella in
cui si trova Apache.
Il connettore AJP13 del primo server ascolta sulla porta 11009 invece che su quella di default
(che è la 8009), il secondo sulla porta 12009.
Occorre poi DENOMINARE i due server Tomcat con un nome UNIVOCO che sarà riferito nel
relativo file di configurazione. Nell'esempio si usano i nomi tomcat1 e tomcat2
Il file va creato rispettando l'impostazione seguente:

```
#
# workers.properties
#
# In Unix, we use forward slashes:
ps=/
# list the workers by name
worker.list=tomcat1, tomcat2, loadbalancer
# ------------------------
# First tomcat server
# ------------------------
worker.tomcat1.port=11009
worker.tomcat1.host=192.168.35.22
worker.tomcat1.type=ajp13
# Specify the size of the open connection cache.
#worker.tomcat1.cachesize
#
# Specifies the load balance factor when used with a load balancing worker.
```

```
# Note:
# ----> lbfactor must be > 0
# ----> Low lbfactor means less work done by the worker.
worker.tomcat1.lbfactor=100
# ------------------------
# Second tomcat server
# ------------------------
worker.tomcat2.port=12009
worker.tomcat2.host=192.168.35.22
worker.tomcat2.type=ajp13
# Specify the size of the open connection cache.
#worker.tomcat2.cachesize
#
# Specifies the load balance factor when used with
# a load balancing worker.
# Note:
# ----> lbfactor must be > 0
# ----> Low lbfactor means less work done by the worker.
worker.tomcat2.lbfactor=100
# ------------------------
# Load Balancer worker
# ------------------------
# The loadbalancer (type lb) worker performs weighted round-robin load balancing with sticky
# sessions.
# Note:
# ----> If a worker dies, the load balancer will check its state once in a while. Until then all work is
# redirected to peer worker
worker.loadbalancer.type=lb
worker.loadbalancer.balanced_workers=tomcat1, tomcat2
#
# END workers.properties
```

2.3.2 Installazione di mod_jk su HP Tru64 Unix

2.3.2.1 Installazione di mod_jk2

Si fa riferimento, su questo ambiente, al connettore denominato **jk2.**
Su questa piattaforma il modulo **mod_jk.so** è diverso dal caso esaminato precedentemente.
Inoltre la configurazione di apache verso tomcat si differenzia anche per la sintassi del file **worker.properties**.
La libreria di connettore è stata ricavata, dopo lunghi tentativi e test, dalla compilazione dei sorgenti sulla piattaforma DEC_UNIX ed è reperibile chiedendola a Gigi Macchi che si è sbattuto
per ottenerla.
Per l'installazione è sufficiente copiare il file compilato **mod_jk2.so** sotto la directory **/usr/local/apache/modules**

2.3.2.2 Configurazione Jk2 Module in httpd.conf

Si effettua un unico passo:
Sotto la direttiva
"# LoadModule foo_module modules/mod_foo.so"
inserire la linea
LoadModule jk2_module modules/mod_jk2.so
Non si fanno altri riferimenti al worker.properties o a contesti tomcat nel file **httpd.conf**.

2.3.2.3 Creazione del file workers2.properties

In questo caso il file dovrà obbligatoriamente chiamarsi **workers2.properties** e la sua sintassi è
diversa dal caso precedente: un esempio riportato dimostra come si definiscono al suo interno
sia i puntamenti che i contesti.

```
[logger]
level=DEBUG
[config:]
file=/usr/local/apache/conf/workers2.properties ### PATH ASSOLUTO del file
debug=0
debugEnv=0
[uriMap:]
info=Maps the requests. Options: debug
debug=0
# Alternate file logger
#[logger.file:0]
#level=DEBUG
#file=${serverRoot}/logs/jk2.log
[shm:]
info=Scoreboard. Required for reconfiguration and status with multiprocess servers
file=/usr/local/apache/logs/jk2.shm ### PATH ASSOLUTO del file
size=1000000
debug=0
disabled=0
[workerEnv:]
info=Global server options
timing=1
debug=0
# Default Native Logger (apache2 or win32 )
# can be overriden to a file logger, useful
# when tracing win32 related issues
#logger=logger.file:0
[lb:lb]
info=Default load balancer.
debug=0
[lb:lb_1]
info=A second load balancer.
debug=0
[channel.socket:192.168.34.50:8009] ### PARAMETRI MODIFICATI PER PRIMO TOMCAT
info=Ajp13 forwarding over socket
debug=0
tomcatId=192.168.34.50:8009
[channel.socket:localhost:8019] ### PARAMETRI MODIFICATI PER EVENTUALE SECONDO
TOMCAT
info=A second tomcat instance.
debug=0
tomcatId=localhost:8019
lb_factor=1
group=lb
group=lb_1
```

disabled=0
[channel.un:/opt/33/work/jk2.socket]
info=A second channel connecting to localhost:8019 via unix socket
tomcatId=localhost:8019
lb_factor=1
debug=0
[channel.jni:jni]
info=The jni channel, used if tomcat is started inprocess
[status:]
info=Status worker, displays runtime informations
[vm:]
info=Parameters used to load a JVM in the server process
#JVM=C:\jdk\jre\bin\hotspot\jvm.dll
classpath=${TOMCAT_HOME}/bin/tomcat-jni.jar
classpath=${TOMCAT_HOME}/server/lib/commons-logging.jar
OPT=-Dtomcat.home=${TOMCAT_HOME}
OPT=-Dcatalina.home=${TOMCAT_HOME}
OPT=-Xmx128M
#OPT=-Djava.compiler=NONE
disabled=1
[worker.jni:onStartup]
info=Command to be executed by the VM on startup. This one will start tomcat.
class=org/apache/jk/apr/TomcatStarter
ARG=start
# For Tomcat 5 use the 'stard' for startup argument
# ARG=stard
disabled=1
stdout=${serverRoot}/logs/stdout.log
stderr=${serverRoot}/logs/stderr.log
[worker.jni:onShutdown]
info=Command to be executed by the VM on shutdown. This one will stop tomcat.
class=org/apache/jk/apr/TomcatStarter
ARG=stop
disabled=1
[uri:/jkstatus/*]
info=Display status information and checks the config file for changes.
group=status:
[uri:192.168.34.50:8009] ### **EVENTUALE RIFERIMENTO AL CONTESTO TOMCAT** REMOTO
info=Example virtual host. Make sure myVirtualHost is in /etc/hosts to test it
alias=192.168.34.50:8009
[uri:127.0.0.1:8003/ex]
info=Example webapp in the virtual host. It'll go to lb_1 (i.e. localhost:8019)
context=/ex
group=lb_1
[uri:/examples]
info=Example webapp in the default context.
context=/examples
debug=0
[uri:/examples1/*]
info=A second webapp, this time going to the second tomcat only.
group=lb_1

debug=0
[uri:/examples/servlet/*]
info=Prefix mapping
[uri:/examples/*.jsp]
info=Extension mapping
[uri:/examples/*]
info=Map the whole webapp

2.4 Installazione di Tomcat

NOTA BENE:
Come prerequisito per l'installazione di Tomcat, occorre che sia installata la versione della **Java**
Virtual Machine 1.4.2_xx. Link ai siti da cui scaricarla
tomcat (tutto)
http://jakarta.apache.org/tomcat
tomcat (binari)
http://jakarta.apache.org/site/binindex.cgi
java
http://java.sun.com/
NOTA BENE:
alcuni produttori forniscono una propria JVM da scaricare dal loro sito.

2.4.1 Installazione di JVM e Tomcat su Windows

Per quanto riguarda l'installazione della **Java Virtual Machine**, se il sistema operativo è Windows, è sufficiente lanciare l'eseguibile scaricato indicando la directory di default su cui
installare. Anche per il **Tomcat** è sufficiente lanciare l'eseguibile scaricato e seguire le indicazioni del setup, lasciando i default proposti ed avendo cura di installare Tomcat come servizio Windows in ascolto sulla porta standard (8080). Il secondo server Tomcat sarà installato
su una porta di ascolto diversa (es. 8081).
Alla fine si dovranno modificare i file di configurazione **catalina.bat** di entrambi i Tomcat come
descritto nel capitolo seguente.

2.4.2 Installazione di JVM e Tomcat su Unix

Nel caso di sistema operativo Unix, per installare la **JVM**, occorre scompattare il file ed eventualmente copiare la root-directory ottenuta sotto il path desiderato. Inoltre è obbligatorio
inserire questo path, comprendente la directory bin, al primo posto nella variabile di ambiente
PATH di Unix.
Per il **Tomcat** occorre semplicemente scompattare il file scaricato sulla directory di installazione
(ad esempio /usr/local) e poi duplicare gli ambienti installati. Ad esempio :

```
# cd /usr/local
# tar fvxz jakarta-tomcat-4.x.tar.gz
# mv jakarta-tomcat-4.x /usr/local/tomcat1
# cp -R /usr/local/tomcat1 /usr/local/tomcat2
```

Infine andranno modificati i file di configurazione **catalina.sh** di entrambi i Tomcat come descritto nel capitolo seguente.

2.4.3 Configurazione di Tomcat

2.4.3.1 Modifica del file catalina.sh o catalina.bat

Settaggi obbligatori di base

Le indicazioni del presente paragrafo valgono sia per Unix/Linux che per Windows, solo che in
quest'ultimo caso il file si chiama **catalina.bat**.
Occorre controllare (soprattutto per quanto riguarda Unix) che siano esplicitamente settate le
variabili d'ambiente che servono a Tomcat per puntare alla Java Virtual Machine. In particolare,
JAVA_HOME e CATALINA_HOME. Per quanto riguarda Unix, nel file catalina.sh, prima della
linea che contiene
"# ----- Verify and Set Required Environment Variables "
occorre inserire le linee seguenti :
JAVA_HOME=/usr/local/jdk1.4 ; export JAVA_HOME
CATALINA_HOME=/usr/local/tomcat1 ; export CATALINA_HOME
Per il secondo server, analogamente si setterà CATALINA_HOME al valore di /usr/local/tomcat2
nel file /usr/local/tomcat2/conf/catalina.sh .

Altri parametri per la Java Virtual Machine

Per ottimizzare l'utilizzo della memoria occorre inserire un altro parametro JAVA_OPTS nel file
catalina.sh (o .bat). Il parametro consente poi di specificare altre opzioni di configurazione e la
sintassi dipende dal sistema operativo e dalla versione della Jvm. In generale, si ha:

□

dove :

–Xms<numero>m specifica la dimensione in MB iniziale del pool di allocazione (default 2MB)
–Xmx<numero>m specifica la dimensione massima in MB del pool di allocazione (default 64MB)
– -Djava.awt.headless=true questo parametro permette alle classi JasperReport di poter esportare dati nei diversi formati supportati (xls, pdf, ecc..), anche se il server tomcat non viene lanciato da interfaccia grafica. In assenza del suddetto parametro e avviando il server da una sessione telnet, in fase di stampa di dati utilizzando jasperReport avremmo il seguente errore:

```
 !#"$&%('#")+*-,$!./.102 34-*&5-%(6#7988*&8
 :/(!#"$&%(';",$!./.-"3*&8)-<=21>-?) :@ 2BA2 :C *161D
 :/(!#"$&%(';",$!./.-"3*&8)-<=2-?,$!./.-" !#EFGHFD
 :/(!#"(I:"JK8&L!C1@M.!7+%/1@8*#%!<=2&%:'"-2 :N(*-M(-$JK8&L!C1@M.!7+%/1@8*#%
<=2&%/:?JK8&L!C1@M/.!7+%/!@8*#%!<=2&%/:" !&EO!P1D
 :Q% 2 :".3" .1L 2-882&L(*&8:.1"2&%('&@% 2#"5(:@$"R!SJK8&L!C-7+%/&T%!@:@-
$@U!2-8"@%!@:@-$@U!21JK8&L C-7+%/1?R1SJK8&L!C17+%/-T%!@:@1$@U!2-8" !
&EF>!P1D
```

Per **Linux e Windows**, aggiungendo :
-Xloggc:c\temp\gc.log -XX:+PrintGCDetails

si ottiene il log degli eventi di garbage collection.
Per **Tru64**, lo stesso parametro si setta con :
–verbose:gc ed i messaggi di log vengono scritti direttamente nel file catalina.out.

Attenzione:

L'attivazione dei file di log rallenta molto le prestazioni della JVM e di conseguenza di

Tomcat,
per cui è consigliabile attivare il log solo in fase di tuning del sstema o per ircercare eventuali
malfunzionamenti.
Occorre controllare sul manuale della macchina virtuale eventuali modifiche alla sintassi.

2.4.4 Modifica del file di configurazione conf/server.xml

Aggiunta di un unico jvmRoute al Catalina engine

Occorre sostituire (riferimento circa linea 100) :
<Engine name="Standalone" defaultHost="localhost" debug="0">
con :
<Engine jvmRoute="tomcat1" name="Standalone" defaultHost="localhost" debug="0">
(Per tomcat2, mettere jvmRoute="tomcat2").
Commentare altri jvmRoute eventualmente presenti

Modifica della porta di controllo

Occorre sostituire (riferimento circa linea 13):
<Server port="8005"
con :
<Server port="11005"
Per il server tomcat2, sostituire la porta 8005 con 12005. In questo modo si eviteranno conflitti
fra i due server se sono sulla stessa macchina.

Modifica della porta del connettore AJP13

Occorre sostituire nella definizione del connettore AJP 13 :
port="8009"
con :
port="11009"
Per il server tomcat2 server, sostituire la porta 8009 con 12009 se sono sulla stessa macchina.

Disabilitazione della porta http standalone

Se non vogliamo che i server Tomcat rispondano direttamente a richieste http, occorre commentare, la definizione della sezione HttpConnector del file server.xml, come nell'esempio :
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
<!—INIZIO COMMENTO
<Connector className="org.apache.catalina.connector.http.HttpConnector"
port="8080" minProcessors="5" maxProcessors="75"
enableLookups="true" redirectPort="8443"
acceptCount="10" debug="0" connectionTimeout="60000"/>
FINE COMMENTO -->

NOTA:
se si vuole lasciare la possibilità ai Tomcat di ascoltare su porte standalone, occorre scegliere
porte diverse per le due istanze Tomcat, in modo da evitare conlitti se sono sulla stessa macchina.

Disabilitazione del connettore WARP

Occorre commentare il tag <Connector...WarpConnector...> , ad esempio :
<Service name="Tomcat-Apache">
<!--
<Connector className="org.apache.catalina.connector.warp.WarpConnector"
port="8008" minProcessors="5" maxProcessors="75"
enableLookups="true" appBase="webapps"
acceptCount="10" debug="0"/>

-->
Ricordarsi di applicare le stesse modifiche anche per il file server.xml del secondo server tomcat2.

NOTA:

Nel file di configurazione per Linux Red Hat SE 3.0 non ho trovato questo connettore. Inoltre non ho commentato i seguenti connettori di tipo non-SSL Coyote sulla porta 8084 (per
tomcat1 e 8086 per tomcat2) per poter effettuare gli aggiornamenti con l'installatore come specificato nella seconda parte del manuale.

Tomcat1

```
<!-- Define a non-SSL Coyote HTTP/1.1 Connector on port 8080 -->
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
port="8084" minProcessors="5" maxProcessors="75"
enableLookups="true" redirectPort="8443"
acceptCount="100" debug="0" connectionTimeout="20000"
useURIValidationHack="false" disableUploadTimeout="true" />
```

Tomcat2

```
<!-- Define a non-SSL Coyote HTTP/1.1 Connector on port 8086 -->
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
port="8086" minProcessors="5" maxProcessors="75"
enableLookups="true" redirectPort="8443"
acceptCount="100" debug="0" connectionTimeout="20000"
useURIValidationHack="false" disableUploadTimeout="true" />
```

2.5 Test dell' installazione

Dopo aver avviato Apache posizionandosi sotto
/usr/sbin
e digitando
./apachectl.sh start
che i due servers Tomcat posizionandosi per il primo sotto
/usr/local/tomcat1/bin
e digitando
./startup.sh tomcat1
e fare analogamente per gli altri Tomcat presenti, è possibile effettuare le verifiche di funzionamento presenti nei 2 prossimi paragrafi.

2.5.1 Controllo che Apache gestisca i contenuti statici

Dal browser presente sullo stesso server cliccare sull'url http://localhost/, oppure da un'altra

macchina in rete cliccare sull'url -Apache>:80 in entrambi i casi
deve essere visualizza la pagina di default di Apache (index.html). Riprodotta sotto

2.5.2 Test che entrambi i tomcat gestiscano le JSP

Per verificare che entrambi i tomcat gestiscano le Java Server Pages dal browser, cliccare

sull'url oppure da un'altra macchina in rete cliccare sull'url

-Apache>/index.jsp in entrambi i casi deve essere visualizza la pagina di default di Tomcat seguente

2.6 Note di configurazione

2.6.1 Gestione del Failover

Quando una delle istanze Tomcat muore, il load balancer entra in azione e "ribilancia" le richieste al(i) server Tomcat rimasto(i) attivo(i). Le sessioni allocate all'istanza del Tomcat

morto
vegono comunque perse e gli utenti si devono loggare di nuovo.

2.6.2 Gestione dei pesi nel bilanciamento

Nel file conf/workers.properties, se si assegna il fattore di load balancing a 100 sia a "tomcat1"
che a "tomcat2", il carico sarà suddiviso equamente fra i vari Tomcat. Se si modifica il valore
lbfactor di "tomcat1" a 101, per esempio, l'effetto ottenuto è far ricevere più richieste a "tomcat1"
rispetto a "tomcat2".
Questa proprietà può essere utile nel caso in cui una delle due macchine risulti effettivamente
più performante dell'altra e possa sopportare un carico più elevato.

2.6.3 Versioni e Test

L'installazione suddetta è stata installata e provata su Sistema Operativo Windows NT, 2000 e
2003.
Le configurazioni testate sono :

- Apache 2.0.49 su Windows NT Server/2000
- Apache 2.0.40 su S.O. RedHat Linux 9a
- Apache 2.0.46 su S.O. Red Hat Linux 3.0 SE
- Tomcat 4.1.30 con java 1.4.2 su S.O. Tru 64 ver. 4.0F
- Tomcat 4.1.30 con java 1.4.2 su S.O. RedHat Linux 9a
- Tomcat 4.1.30 con java 1.4.2 su S.O. Sun Solaris 2.8
- Tomcat 4.1.31 con java 1.4.2 su Red Hat Linux 3.0 SE
- Tomcat 5.5.12 con java 1.5.0_05 su Red Hat Linux 3.0 AS 3.0