Installation Scenarios

This chapter covers the following topics:

- Installation within Your Own Tomcat Environment
- Usage with WebSphere
- Using Application Designer on Tomcat behind Apache

Installation within Your Own Tomcat Environment

Application Designer provides an environment which includes a Tomcat servlet engine and a Java Runtime Environment. Inside this environment, Application Designer is installed as a web application.

Of course, you can also use Application Designer inside your own Tomcat and Java environment. You can use any Tomcat servlet engine from release 4.0 on.

Proceed in the following way:

- Install Application Designer (both environment and Application Designer installation).
- Copy the directory *<installdir>/tomcat/webapps/cis* into the *webapps* directory of your Tomcat installation.
- In case you have updated the *web.xml* file, adjust the *web.xml* file so that
  - the `cis.home` parameter points to the file directory of the web application, and
  - the `cis.log` parameter points to the log directory.

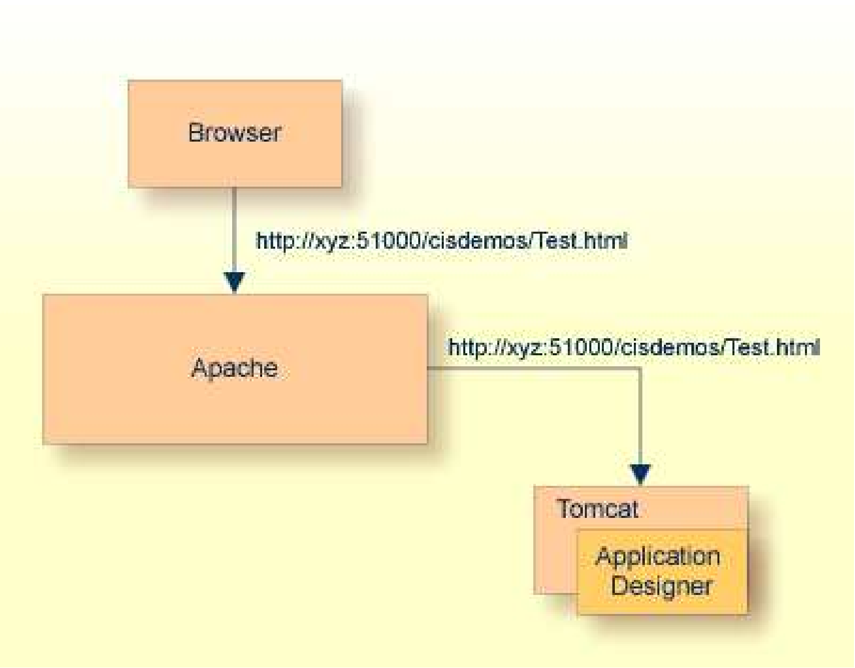
Usage with WebSphere

Application Designer is deployed as a web application or part of a web application to Websphere. Proceed in the following way:

- Install Application Designer (both environment and Application Designer installation).
- Build a web archive file (*.war*) by zipping the directory contents of *<installdir>/tomcat/webapps/cis/* into the file *citv<nn>.war*. Or use the WAR Packager to create the *.war* file.
- Deploy this *.war* file to Websphere. Adjust the *web.xml* file during deployment as described above if you have changed the file.

Using Application Designer on Tomcat "behind Apache"

This section describes the configuration of the following scenario:


Browser
http://xyz:51000/cisdemos/Test.html
Apache
http://xyz:51000/cisdemos/Test.html
Tomcat
Application Designer

Apache is used as the web server to the outside world. All requests from outside are going through Apache. This means that only one server and one port is opened to the outside (typically the Apache server with port 80). Requests to Application Designer which runs, for example, on Tomcat under port 51000 are going as a "port 80" request to Apache and are routed from Apache as a "port 51000" request to the Tomcat server.

The consequence is that you only have one point for letting outside users into your network. This simplifies the configuration of firewalls.

Steps

The configuration steps for this scenario are:

1. Install Application Designer just as normal and check whether it is running normally inside Tomcat.
2. Configure Apache to route certain requests to Tomcat/Application Designer.
3. Restart Apache.

Application Designer Installation

Just perform a normal installation of Application Designer (both environment and Application Designer installation).

Apache Configuration

Configure the Apache *httpd.conf* file in the following way:

1. Find the following line and activate it by removing the comment letter (#) at the beginning. If it is already active, there is nothing to do for you.

```
LoadModule rewrite_module modules/mod_rewrite.so
```

2. Do the same for the following line:

   ```
   LoadModule proxy_module modules/mod_proxy.so
   ```

3. Do the same for the following line:

   ```
   AddModule mod_rewrite.c
   ```

4. Do the same for the following line:

   ```
   AddModule mod_proxy.c
   ```

5. Search for the configuration lines below and activate them by removing the comment letters:

   ```
   #
   # Proxy Server directives. Uncomment the following lines to
   # enable the proxy server:
   #
   <IfModule mod_proxy.c>
       ProxyRequests On

       <Directory proxy:*>
   #        Order deny,allow
   #        Deny from all
   #        Allow from local
            Allow from all
       </Directory>

       #
       # Enable/disable the handling of HTTP/1.1 "Via:" headers.
       # ("Full" adds the server version; "Block" removes all outgoing Via: headers)
       # Set to one of: Off | On | Full | Block
       #
   #    ProxyVia On

       #
       # To enable the cache as well, edit and uncomment the following lines:
       # (no cacheing without CacheRoot)
       #
   #    CacheRoot "C:/Programme/Apache Group/Apache/proxy"
   #    CacheSize 5
   #    CacheGcInterval 4
   #    CacheMaxExpire 24
   #    CacheLastModifiedFactor 0.1
   #    CacheDefaultExpire 1
   #    NoCache a-domain.com another-domain.edu joes.garage-sale.com

   </IfModule>
   ```

6. Configure the rewrite module by appending the following lines at the end of the *httpd.conf* file:

   ```
   RewriteEngine On
   RewriteRule ^/cis/(.*) http://localhost:51000/cis/$1 [P]
   ```

This definition routes all incoming requests to Apache, which have certain context roots to the 51000 port. If there is a request to Apache with the URL *http://apache:80/cis/cisdemos/Test.html*, Apache forwards this request to Tomcat with the URL *http://localhost:51000/cis/cisdemos/Test.html*.

Advantages of this Configuration

The advantage of this type of configuration is that Apache and Tomcat are totally independent from one another. Actually, it is a coupling of two standalone systems. Tomcat is running in a separate process and as a consequence can do everything without disturbing Apache. Intranet applications can still access Tomcat directly via the Tomcat port.

The disadvantage of this scenario is that every request which is going to Apache is routed to Tomcat via a TCP/IP routing - which from the performance point of view is consuming more resources than directly processing the request inside the Apache process.

Of course, more complex configurations are also possible with Apache. This section just described one scenario.