

Oracle Servlet Engine OSE

Karl Seematter, Akadia AG

Introduction

For internet applications with dynamic contents, more and more J2EE components (Servlets, JSP's, EJB) come into operation.

With servlet based applications, the servlets are stored in the filesystem and accessed and executed by a web server with its servlet engine e.g. Apache/Jserv. For database access, methods like JDBC or SQLJ are used.

Oracle8i release 3 (8.1.7) brings some extensions in the area of serverside Java programming. In this article we consider a configuration of a Web application where:

- Java servlets **with** database access are loaded into the database and executed by the Oracle Servlet Engine.
- Java servlets **without** database access are stored in the filesystem and executed by the Oracle HTTP Server (Apache/Jserv).

References

[#]	Name of document	source
[1]	Oracle 8i Oracle Servlet Engine User's Guide	http://technet.oracle.com/docs/products/oracle8i/doc_index.htm
[2]	Oracle 8i Java Developers Guide	
[3]	Oracle Java Tools Reference	
[4]	Java Naming and Directory Interface JNDI	http://java.sun.com/products/jndi/
[5]	Oracle 8i Database Access with Apache / Jserv	http://www.akadia.com

Components

All required components (Oracle HTTP Server , JRE, JDK, JSDK) are available with the Oracle 8i Release 3 installations CD-ROM.

Terms

Oracle Servlet Engine OSE

See [1]

OSE is a built-in Web server with an integrated servlet engine running inside Oracle 8i. OSE executes Java Server Pages, Servlets and Java Stored Procedures.

Java Naming and Directory Interface JNDI

See [3] [4]

JNDI defines a hierarchical directory model for the administration of Web application objects (Servlets, initial parameters, JSP's etc.).

Access to the JNDI structure and its objects is permitted by a session shell tool called „sess_sh“. The tool is driven by a command language similar to Unix. JNDI supports three type of access rights: READ, WRITE and EXECUTE.

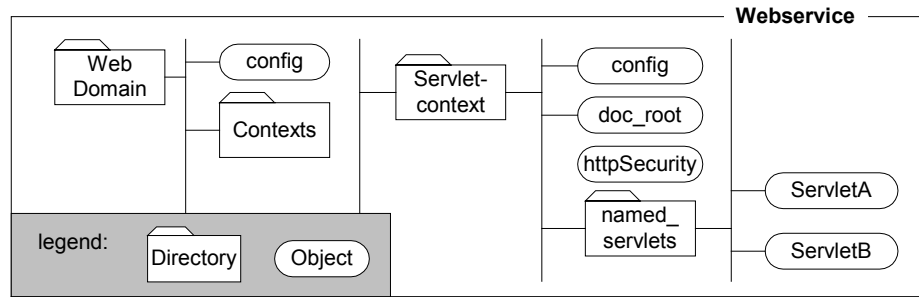


Fig 1 Example: JNDI structure for a single domain Web service

A **Web service** defines a root directory within the JNDI, the hierarchical directory structure and the stored objects for one ore more **Webdomains**.
 OSE supports two different Web service configurations: single-domain and multi-domain.
 Example (command within the session shell tool):

```
$ createwebservice -root /SougDemoRoot SougDemo
                        |                               |
                        |                               |→Name of the Web service
                        |→Name of the Rootdirectory
```

In a single-domain Web service the **Web domain** corresponds to the rootdirectory of the Web service.

The **Servletcontext** -(part of) an application - contains servlets, configuration parameters and eventually pointers to static Web contents in the OS filesystem.

The hierarchical model represents a presentation layer and complies to the JNDI standard. In effect, directories and objects are stored in database tables.

Endpoint

An endpoint corresponds to a dynamic port in the listener for a Web service. The listener receives requests from a HTTP-Client or an external HTTP-Server and transfer them to the Web service or to the OSE respectively.

Example:

```
$ addendpoint -port 7778 -register SougDemo SougEndpt01
                        |                               |
                        |                               |→Name of the Endpoint
                        |→Webservice
```

MOD_OSE

The modul MOD_OSE is attached to the Oracle HTTP Server (use directive „LoadModule“). MOD_OSE serves as a conduit form the Oracle HTTP Server to OSE using Net8.

Access to OSE Servlets

Oracle8i OSE provides two different access methods:

1.) Direct access from a HTTP-Client to the OSE

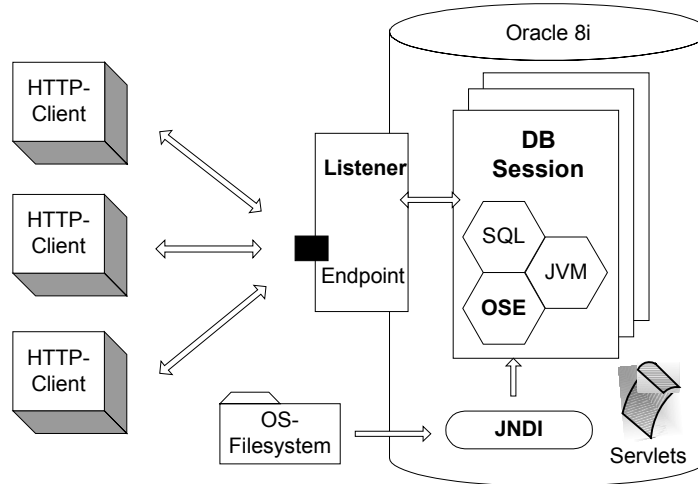


Fig2 Direct client access

Application Flow

The HTTP-Client sends an URL in the form:

`http://<host>:<port>/<webservice>/<servlet>`

to the listener port (endpoint). The listener forwards the request to the OSE, which reads the associated servletcontext from the JNDI and executes the servlet.

An URL with a the pattern:

`http://<host>:<port>/<webservice>/<html_page>`

causes the OSE to read the HTML-page from the predefined OS-Directory.

Database Session

Each HTTP-Client request initiates the OSE to start a new DB session. The session runs in the context of the Web service owner. A DB session activates its own Java Virtual Machine (JVM). The JVM runs in the same address space as the SQL- and PLSQL-engine and provides therefor a faster access to database objects without additional login. DB-Session termination is controlled by the use of a timeout parameter.

2.) Indirect access over the Oracle HTTP-Server (Apache)

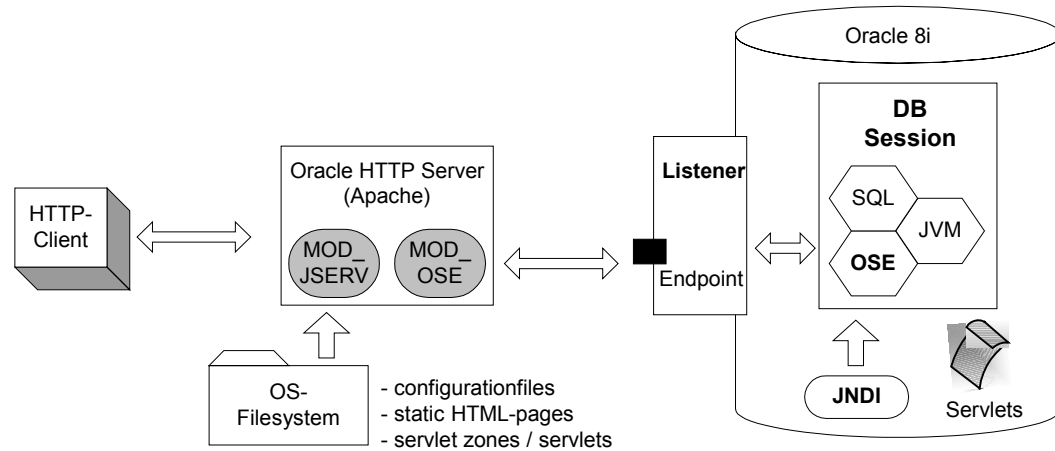


Fig3 Indirect access to the OSE over the Oracle HTTP Server

In this configuration the OSE serves as an additional servlet engine for the Oracle HTTP server. Servlets **without database access** are stored in the repository or in a servlet zone of the Apache Web server respectively (see [5]). Servlets **with database access** are loaded into the database. This architecture allows a flexible scalability for Web applications.

Application Flow

Der Oracle HTTP-Server analyses a client-URL:

- if it contains a static page from the path indicated by the DOC_ROOT directive, this page is sent back to the caller.
- if it contains a servlet zone, the request is forwarded to the MOD_JSERV module.
- if it contains a servletcontext of an OSE-handler, the first part „http://<host>:<port>“ of the URL is removed and the remaining part is transferred to MOD_OSE, which then connects to the OSE.

Remark: A servletcontext „SougDemo“ is defined with the following directive:
(configurationfile mod_ose.conf)
<Location /SougDemo/* >
SetHandler aurora-server
</Location>

Stateful and Stateless Servlets in the OSE

Servlets loaded and published in a JNDI servlet context can be configured to run either in stateful or in a stateless context.. A stateful application tracks status information for all request-response sequences of a HTTP-client session. Cookies or URL-rewrites are the mechanism in charge. For details see [1].

Summary

Using the OSE for 3-tier applications offers additional scalability. Depending of its task, Java servlets can run either in the middle-tier or in the database server.

The administration of a Web service within the JNDI namespace is rather annoying! A GUI tool or at least a browser would make the configuration work easier. Beside that, the session shell tool „sess_sh“ has errors (version 1.0)!. E.g. the command to remove a whole Web service („rm -r“) does not its complete job. Therefore you have to delete the directory trees step by step.

Example Configuration

On our homepage www.akadia.com under the item „Current Publications“ you will find running examples for the different access methods and step by step instructions for the configuration of the Oracle Servlet Engine, the Oracle HTTP Server (Apache/Jserv) and the particular examples.

www.akadia.com:7777 leads you directly to the example page.

Karl Seematter, Akadia AG

Mail: karl.seematter@akadia.com

Internet: www.akadia.com