

OBPM 11g Demo

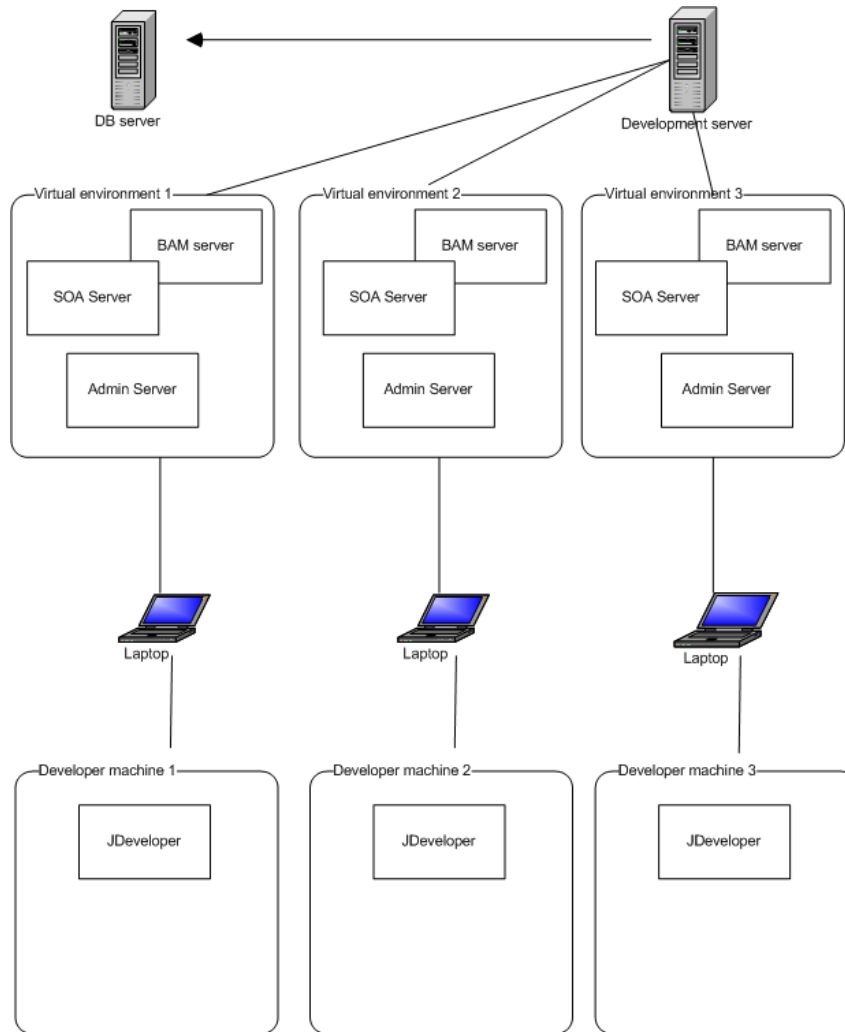
Preparing a Development
Environment

02.11.2010

This document covers some best practices about preparing a capable development environment for Oracle BPM 11g.

1. Development Architecture

The proposed development architecture is as follows:



It is made up of two servers:

- The DB server is hosting an Oracle DB where each development environment will put its metadata store.
- The development server must have a 64-bit operating system, at least a capable quad-core CPU, and enough memory for the virtual environments it is hosting (one for each developer). In each virtual environment the default SOA domain setup will be used containing 3 servers. The BAM server does not need to run all the time, the other two are necessary for the whole development time, therefore each virtual environment should contain at least 5 GB of RAM.

The developer machines will host JDeveloper, and deploy each artifact to the respective virtual environment, so 2 GB memory in the developer machines should be sufficient.

2.

Development Best Practices

- (1) Do not use the built-in subprocess feature; it is merely a graphical grouping. Should you need real subprocesses, create them as processes, and include them in the composite.xml (in the graphical design they will show up as an automatic activity).
- (2) If the process size reaches about 1 MB, do not use the built-in deploy features in JDeveloper. It is faster to create the SAR archive and deploy it with a script.
- (3) Component declarations generated by JDeveloper often contain absolute paths to Oracle SOA or BPM artifacts. If a component (for example a human task) needs to be regenerated, JDeveloper reads paths from these declarations and doesn't throw an error even if the path does not exist on the machine. If the developers have their Oracle SOA Homes or workspaces in different directories, these paths need to be extracted and a utility must be created to replace these paths after updating from the version control system.
- (4) Because of this restriction it is best to add external jars the project depends on to the project structure in the version control system as well, and add it to the project from there. This way relative paths can be used in the configuration XMLs hence the dependency will be added correctly on other developer machines as well.
- (5) When designing business rules it must be thought about whether a business rule must be reusable in some other process. If so, the business rule must be deployed from a separate application and included in the two processes as a web service. Another solution if the same subprocess containing the same business rule is repeated in two different processes is to extract the subprocess itself as a separate application and use the process as an automatic activity (like in point 1).
- (6) Automatically generated human task WAR files contain about 10MB of jars. This means the more human tasks you have, the bigger EAR file (containing all the human task WAR files) you will get. A good approach is to extract the jars from the human task WAR files and deploy them as a shared library to weblogic. After that, the generated WAR profile in JDeveloper must be modified not to include any libraries, but the weblogic.xml must be modified by hand to include the library reference to the deployed shared library. This method largely decreases human task EAR file size, thus reduces deploy times.