Setting up a WebLogic Server Environment

At the end of this module you will be able to:

- ✓ Explain the motivation behind distributed systems
- ✓ List the major components of the J2EE specification
- ✓ Know the terminology used throughout the course

Road Map

1. **Distributed Architecture**
   - J2EE Technologies
   - Web & WLS Terms
2. Setting Up a WebLogic Server Environment

Distributed Systems

- *Distributed systems* divide the work amongst several independent modules.
- Failure of a single module has less impact on the overall system which makes them more:
  - available
  - scalable
  - maintainable

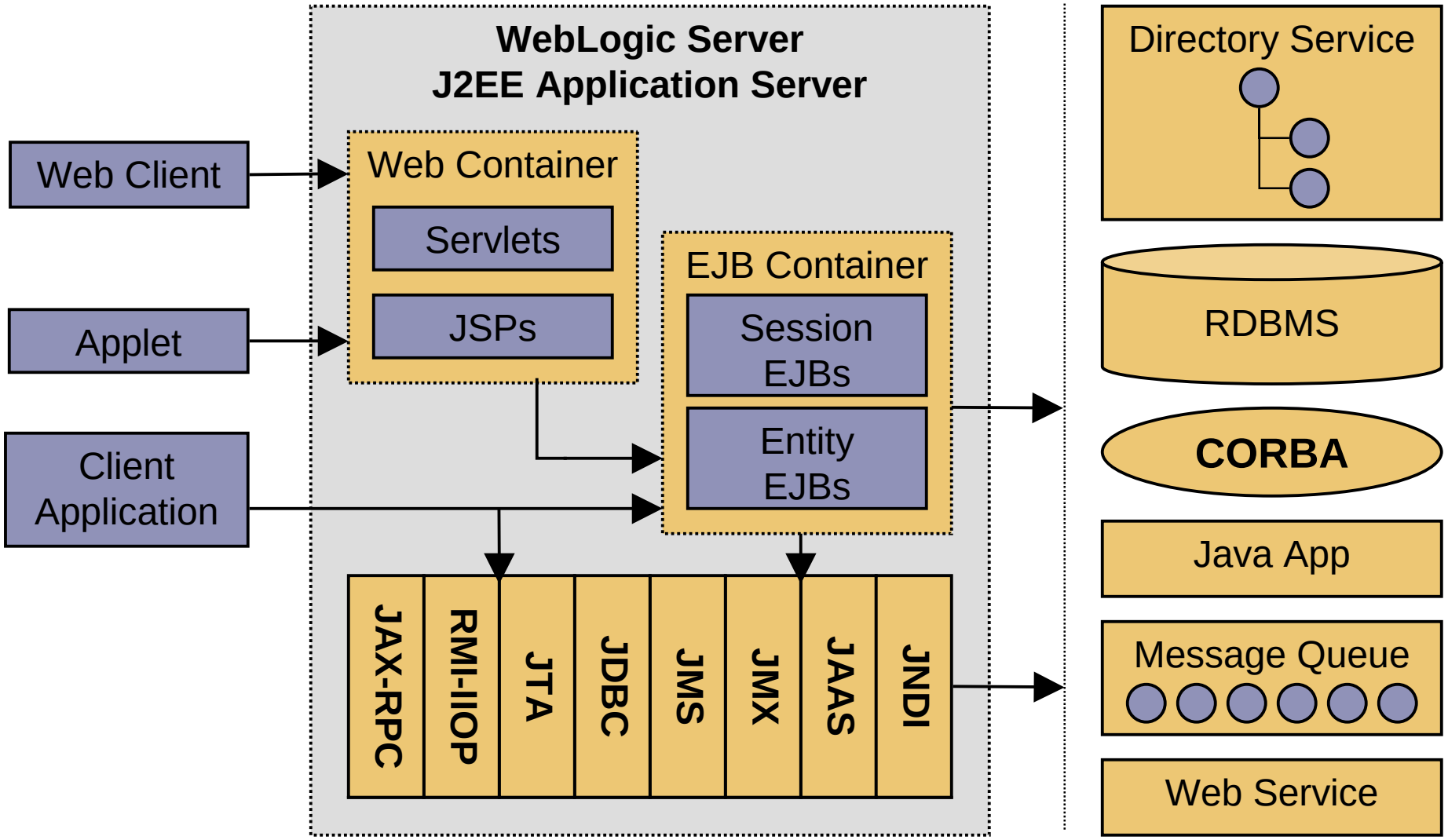
How Standards Help

- Many of the advantages of distributed systems come from standards.
- Standards:
  - provide separation of difficult problems to separate platforms
  - allow modularization of complex hardware and software
  - allow a larger portion of project costs to go toward solving business software needs

The J2EE Standard

- Java Platform 2 Enterprise Edition (J2EE) helps to overcome distribution liabilities.
- Applications deployed with J2EE technologies are:
  - standardized
  - adherent to specification guidelines
  - written in Java
  - deployable in any compliant application server

The J2EE Architecture



Java Servlets

- A Servlet is a Java "program" that executes on the server, accepting client requests and generating dynamic responses.
- The most prevalent type of Servlet is an `HttpServlet` that accepts HTTP requests and generates HTTP responses.
- Servlets:
  - do not just generate HTML
  - can also be used to generate other MIME types, such as images

JavaServer Pages (JSPs)

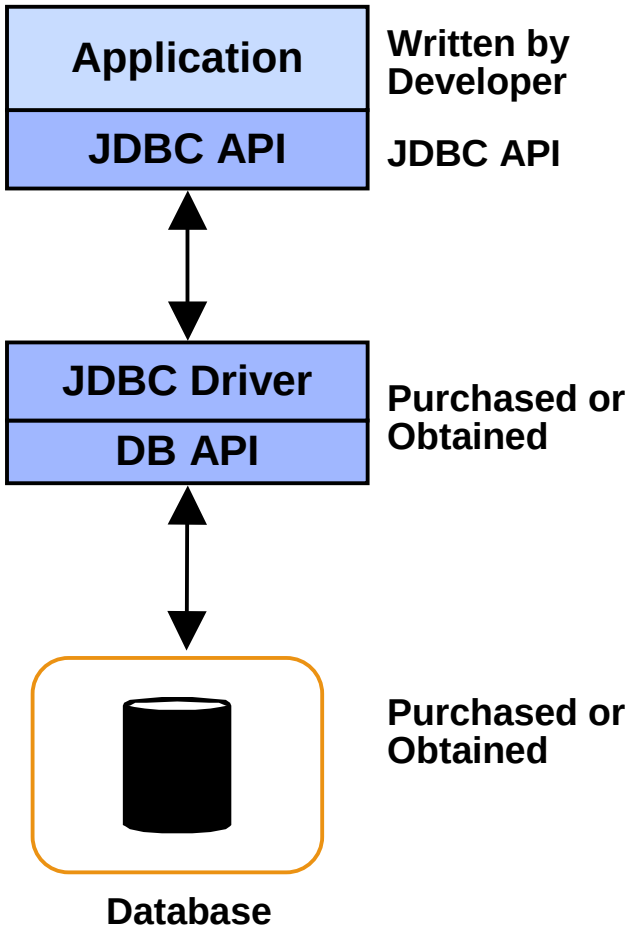
- JavaServer Pages are HTML documents interweaved with Java.
- JSPs:
  - provide a dynamic response that is based on the client's request
  - provide for the separation of responsibilities between Web presentation and dynamic content
  - are portable (write once, run anywhere)
  - compile and run as servlets

Enterprise JavaBeans (EJBs)

- EJBs are distributed components written in the Java programming language.
- EJBs:
  - provide distributable and deployable business services (logic) to clients
  - have well-defined interfaces
  - are reusable across application servers
  - execute within a container that provides management and control services
- WebLogic Server 9.X supports the EJB 2.1 specification.

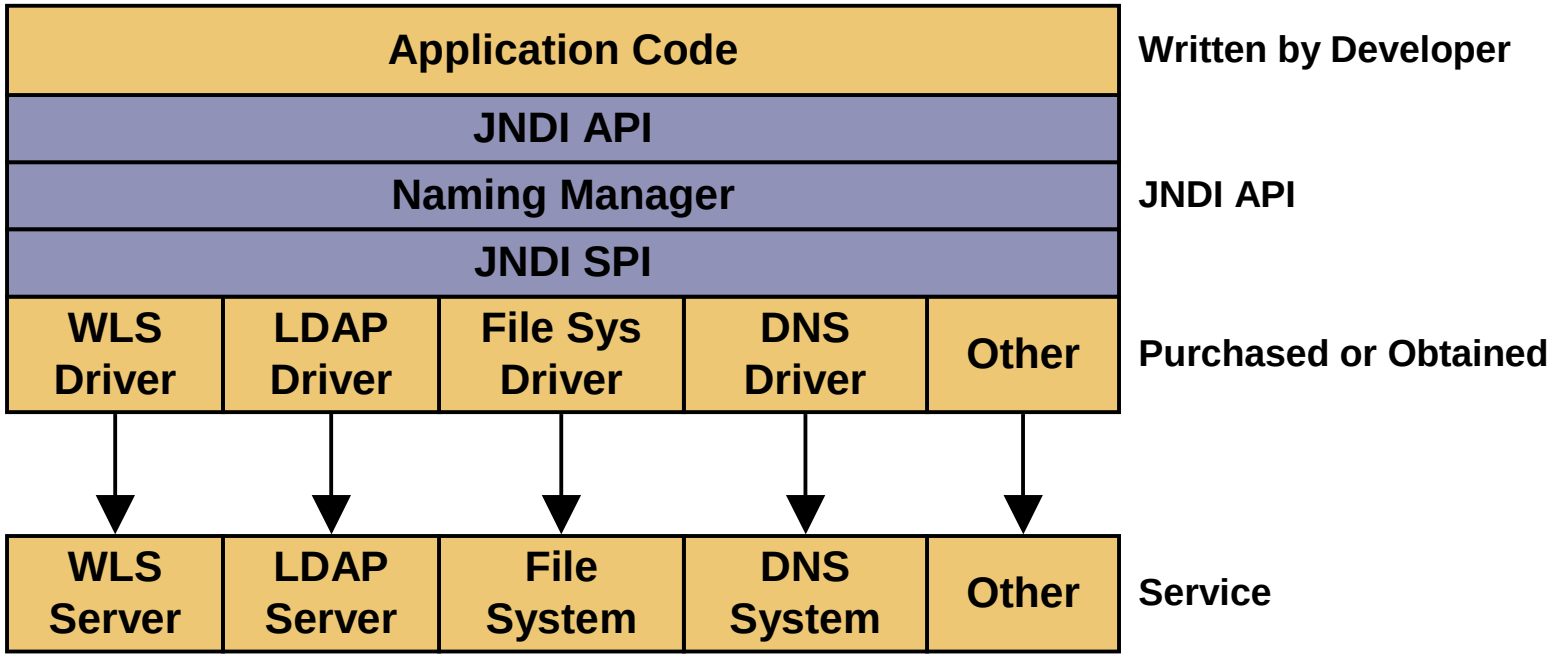
JDBC (Java Database Connectivity)

- JDBC is:
  - a standard Java interface for accessing heterogeneous databases
  - a specification that defines four different driver types for connecting to databases



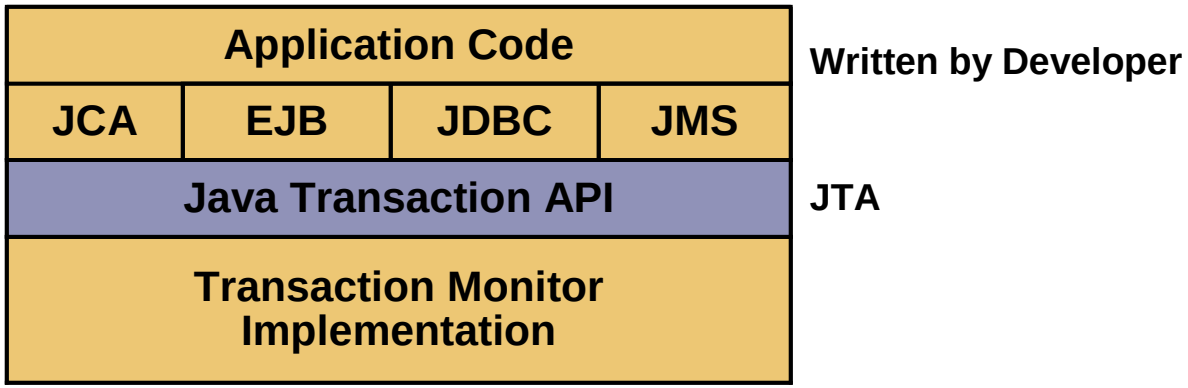
Java Naming & Directory Interface (JNDI)

- JNDI is:
  - a Java API for accessing naming and directory servers
  - built as a layer over DNS, LDAP, etc.



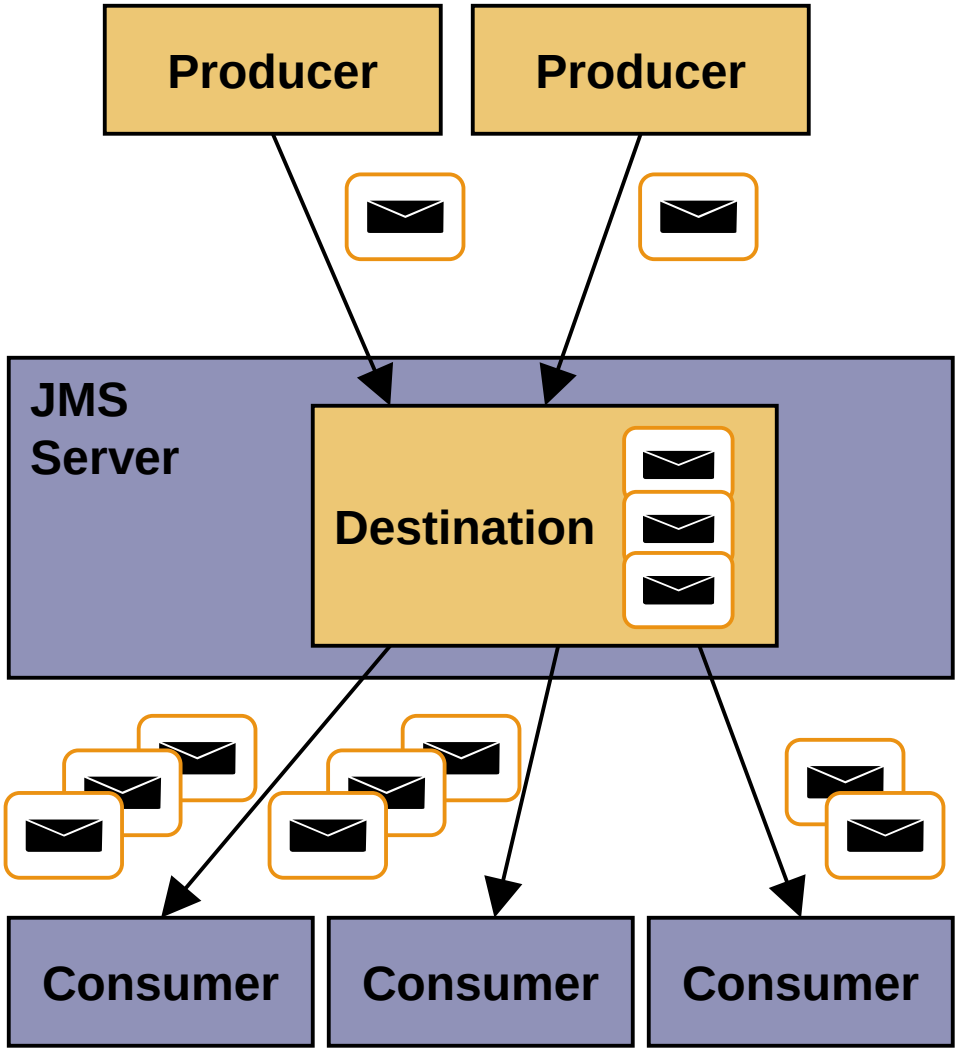
Java Transaction API (JTA)

- JTA is a standard Java API for demarcating transactions within a program.
- WebLogic Server supports local and distributed transactions.



Java Message Service (JMS)

- JMS is a Java API for accessing message-oriented middleware.
- The interface supports:
  - the Point-to-Point domain
  - the Publish/Subscribe domain
  - guaranteed message delivery
  - transactional participation
  - dynamically configurable services
  - application- or system-scoped resources
  - interoperability with other messaging systems

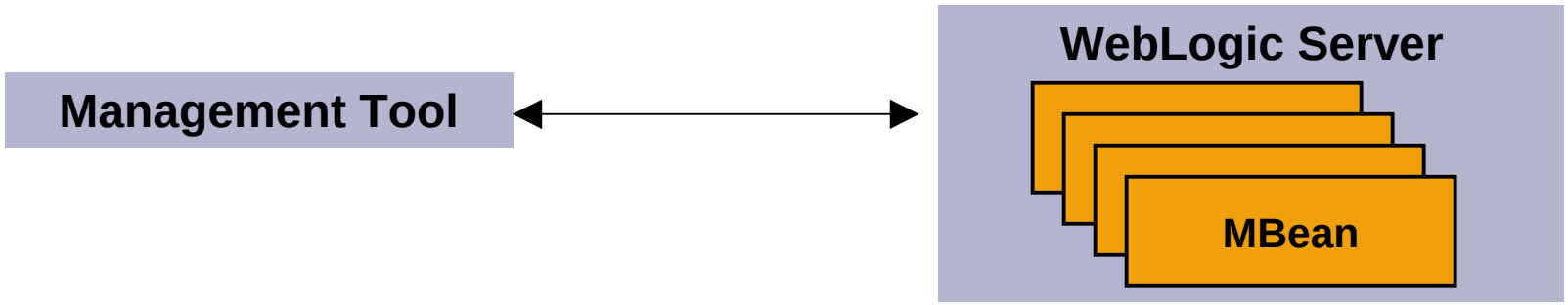


Java Authentication and Authorization

- Java Authentication and Authorization Service (JAAS) is a Java-based security management framework.
- JAAS supports:
  - single sign-on
  - a Pluggable Authentication Module (PAM)
- JAAS enables flexible control over authorization whether it is based on:
  - users
  - groups
  - roles

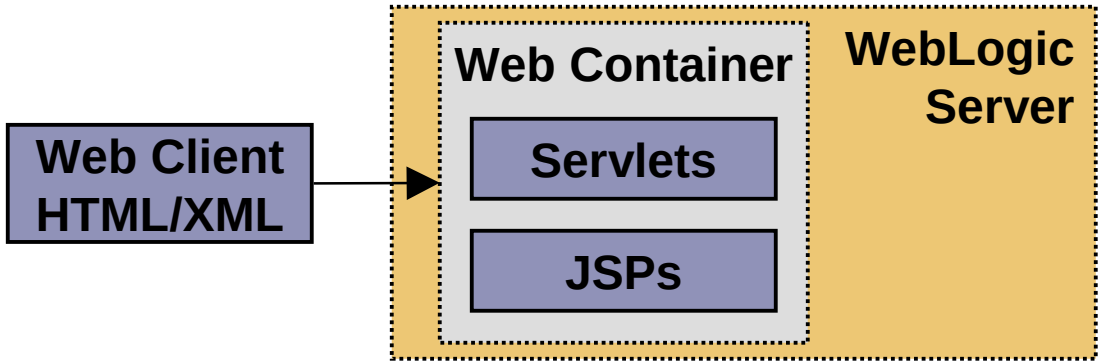
JMX

- The Java Management Extensions (JMX):
  - defines a standard infrastructure to manage a device from Java programs
  - decouples the managed device from the management tools
- The specification describes MBeans, which are the building blocks of JMX.



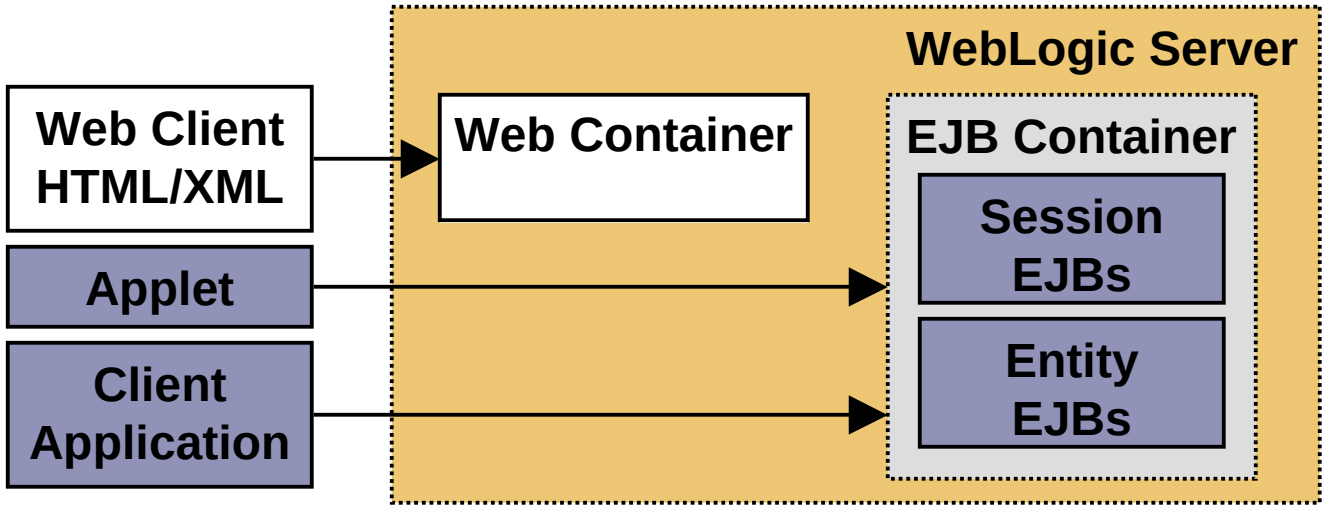
Web Client

- A *Web client* interacts with WLS via HTTP using Servlets/JSPs.
- Types of Web clients include:
  - Browser
  - Web Services (SOAP over HTTP)



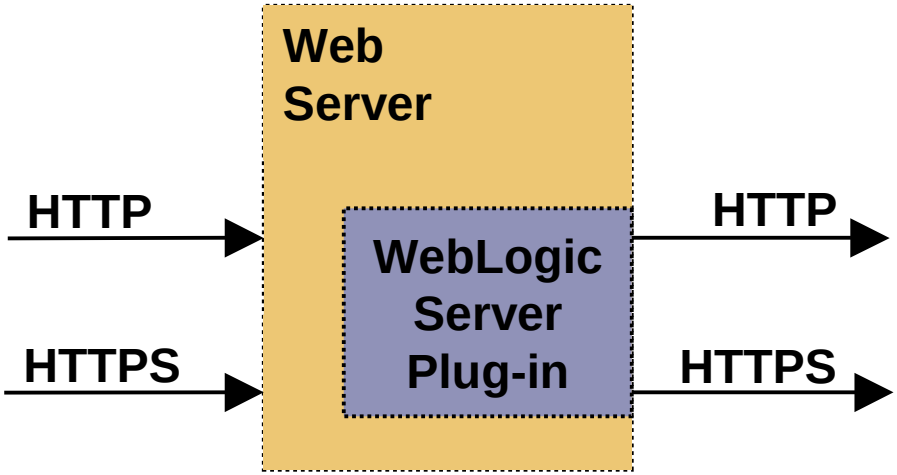
Client Application

- A *client application* interacts with WLS through JRMP/T3, IIOP, COM.
- Types of Clients include:
  - Standalone Java applications
  - Applets within a browser



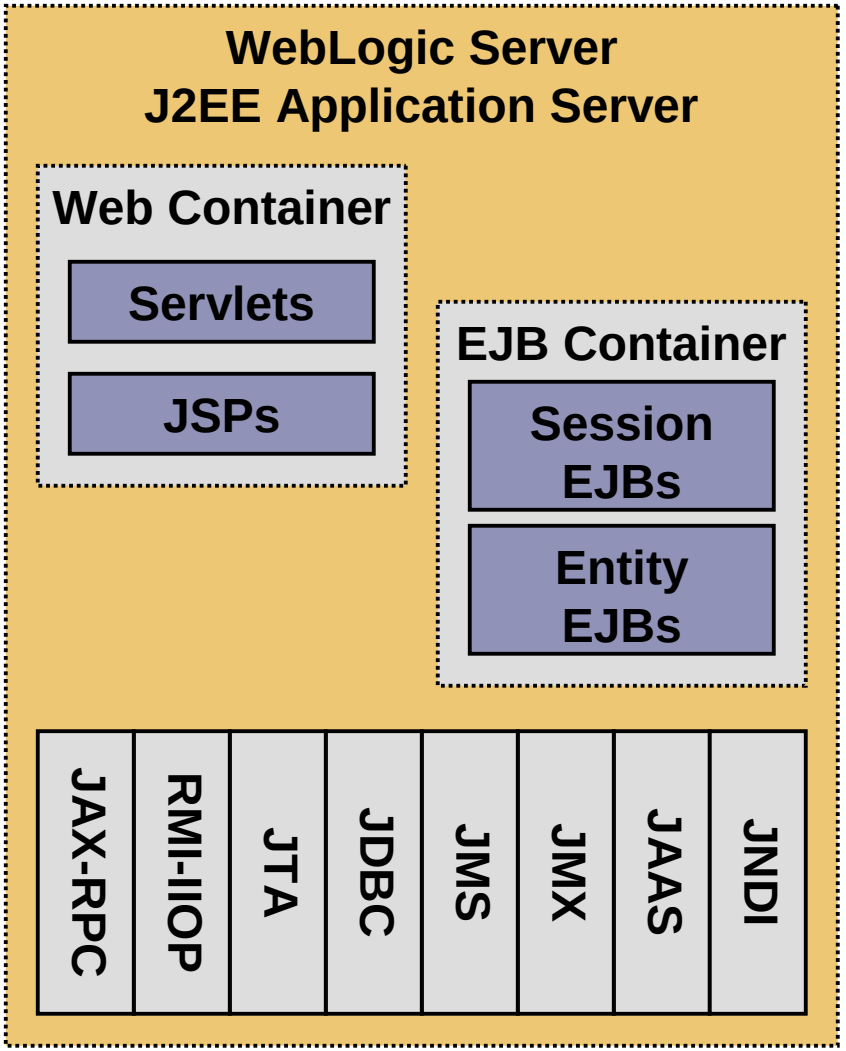
Web Server

- Web servers:
  - provide Web content
  - communicate via HTTP, FTP, etc
  - can handle CGI requests
  - proxy some requests to Application Servers



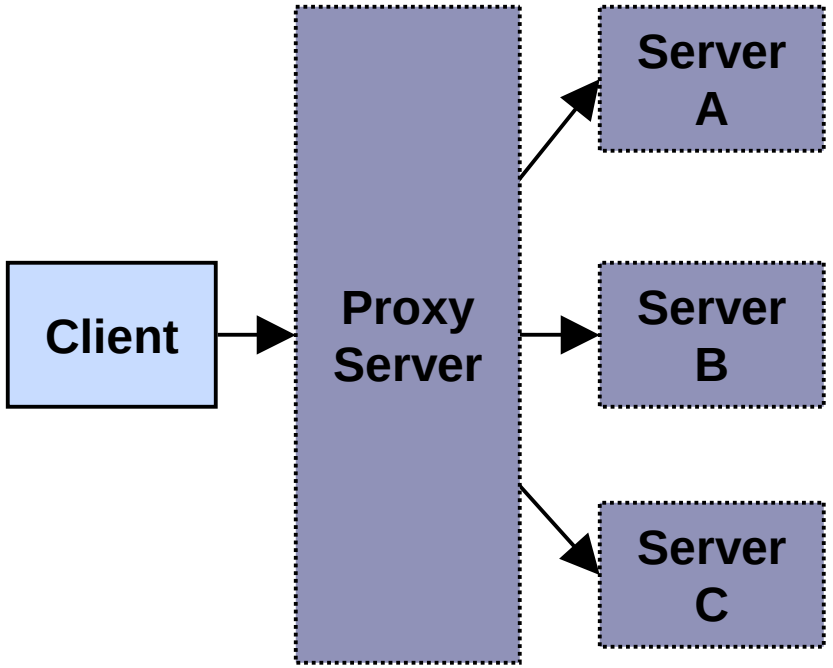
Application Server

- Application servers:
  - provide services that support the execution and availability of deployed applications
  - handle heavier processing chores than Web servers



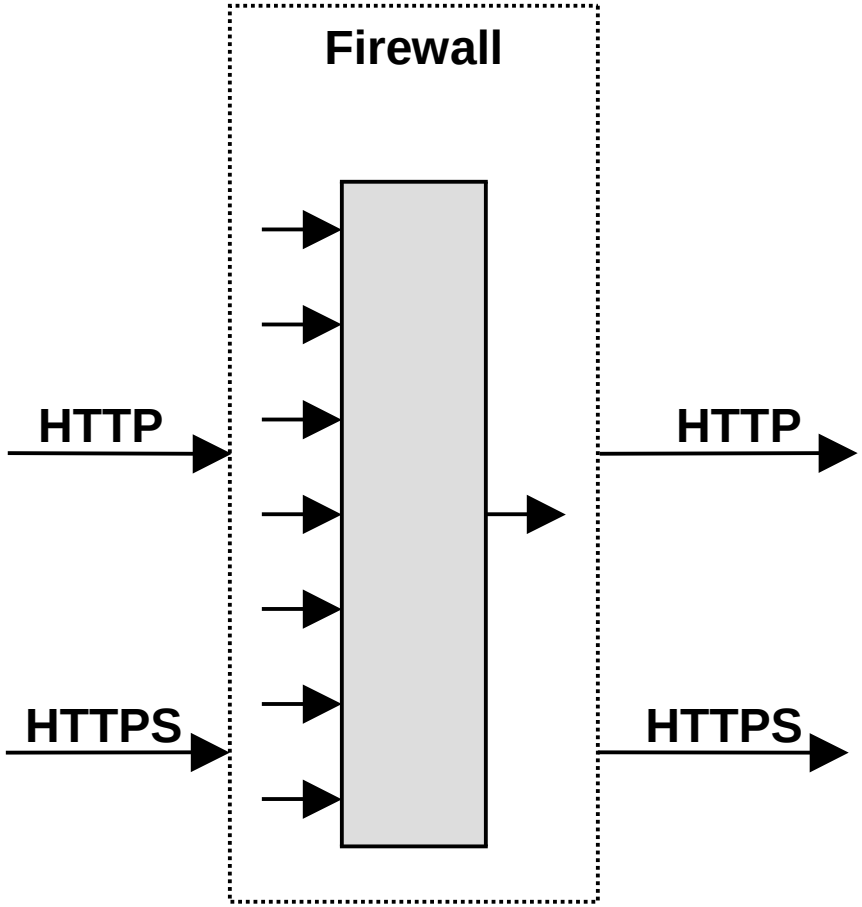
Proxy Server

- A proxy server:
  - forwards requests to other machines
  - can be used as a level of indirection and security
  - can be used to load balance a system

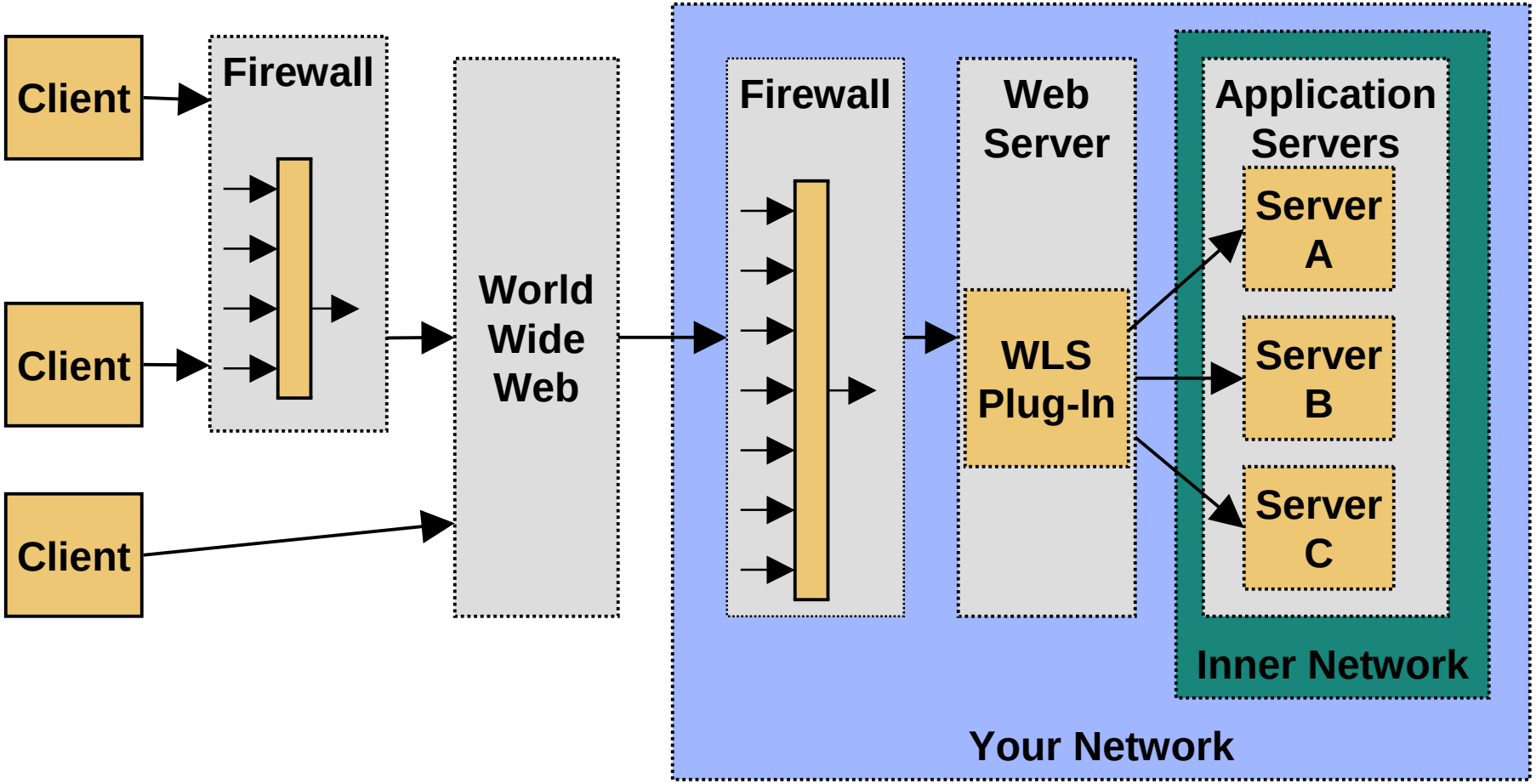


Firewall

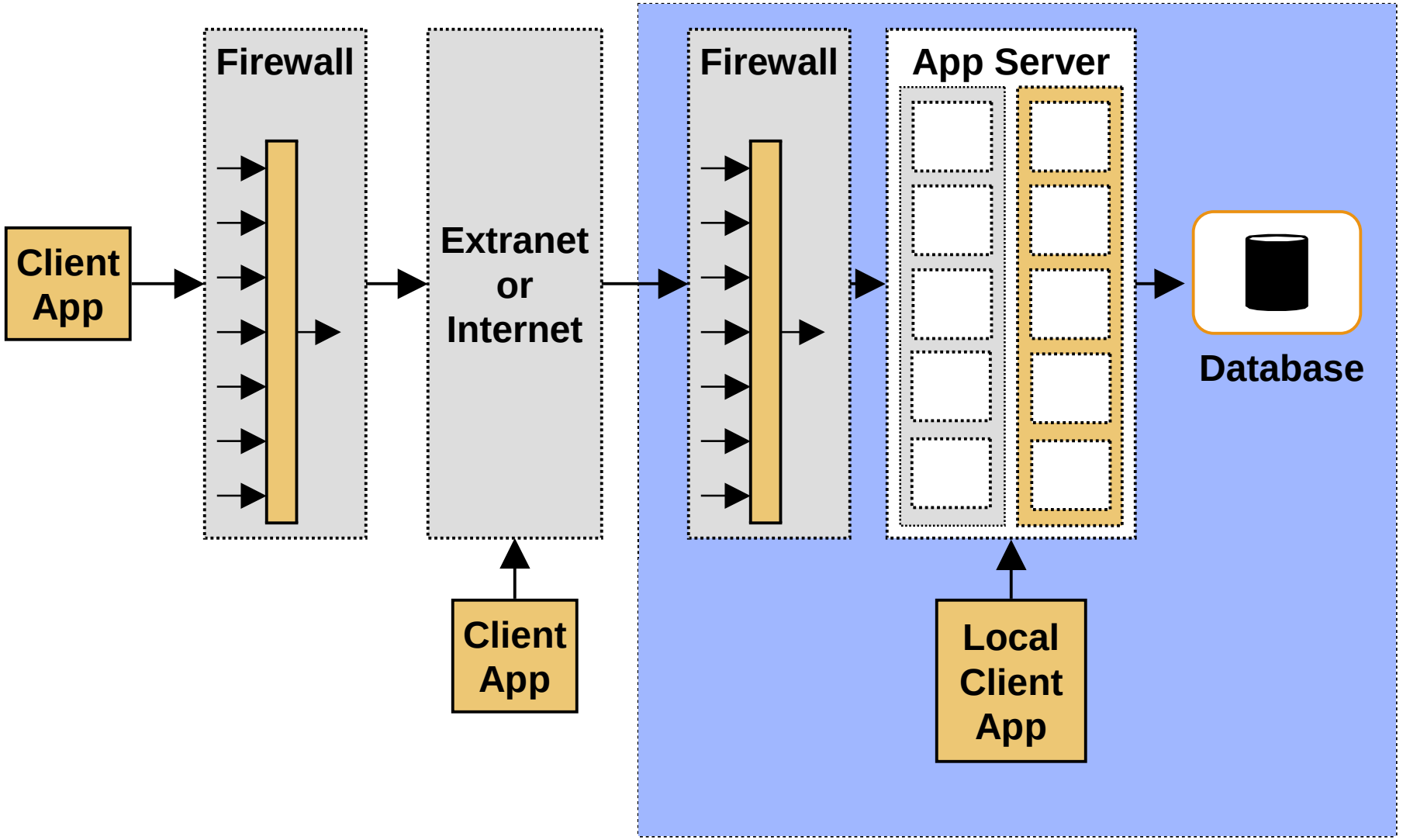
- Firewalls:
  - provide filtering, authorization, and authentication services
  - help keep out hackers
  - map port requests
  - can act as proxy servers
  - can decrease back end network activity



A Web App Server Configuration

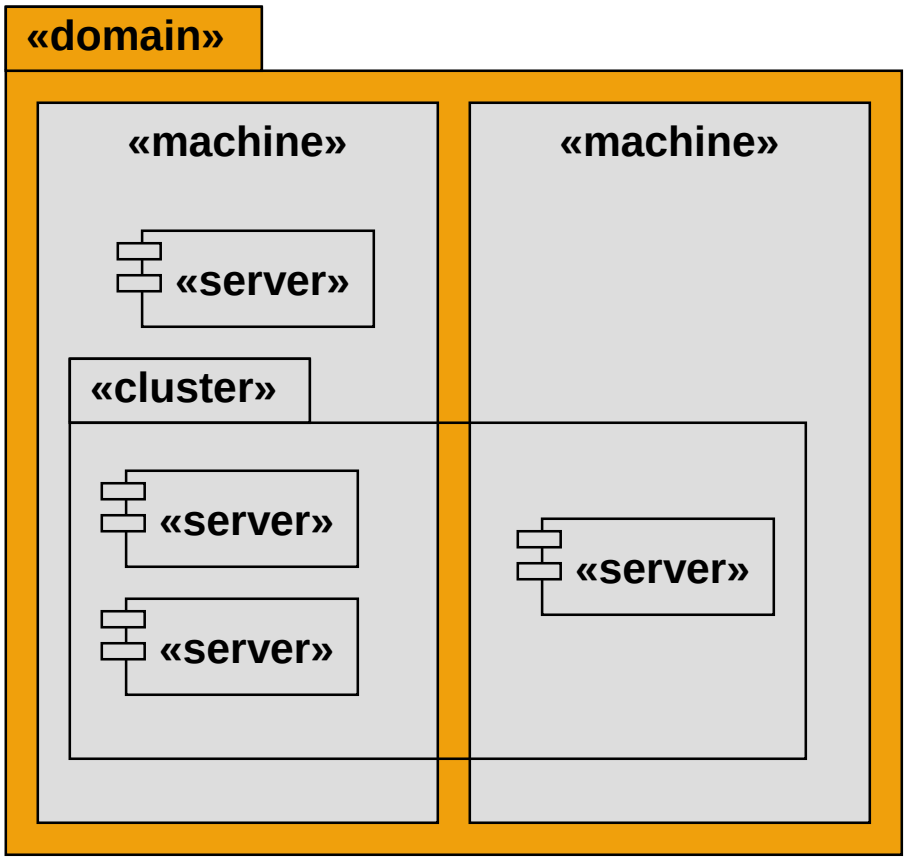


An Application Server Configuration



Definition: Domain

- A *domain* is a logically-related group of WebLogic Server resources that you manage as a unit.
- A domain provides *one* point of administration.
- A WebLogic Server domain can logically separate:
  - Development, test, and production applications
  - Organizational divisions

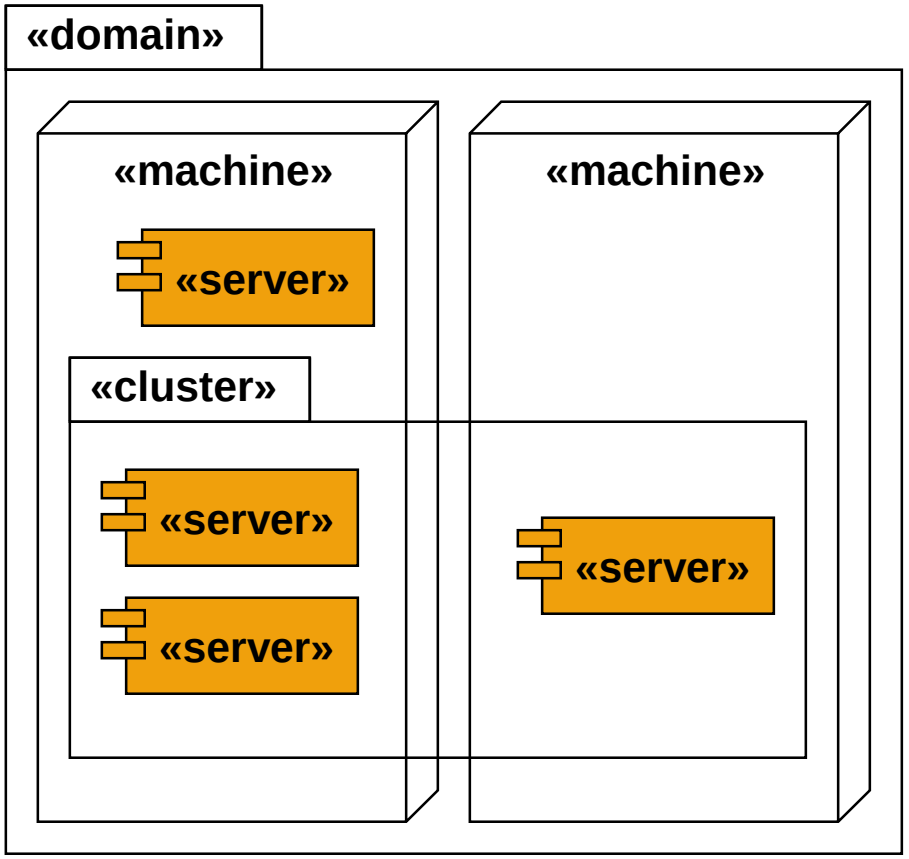


Why Use Domains?

- A *domain* is an administration feature that:
  - Is transparent to applications
  - Can be configured and administered, for technical or business reasons, even after applications are developed or in production
- WebLogic Server domains can be used to separate:
  - Development, test and production applications
  - Administration and operational responsibilities
  - Organizational or business divisions

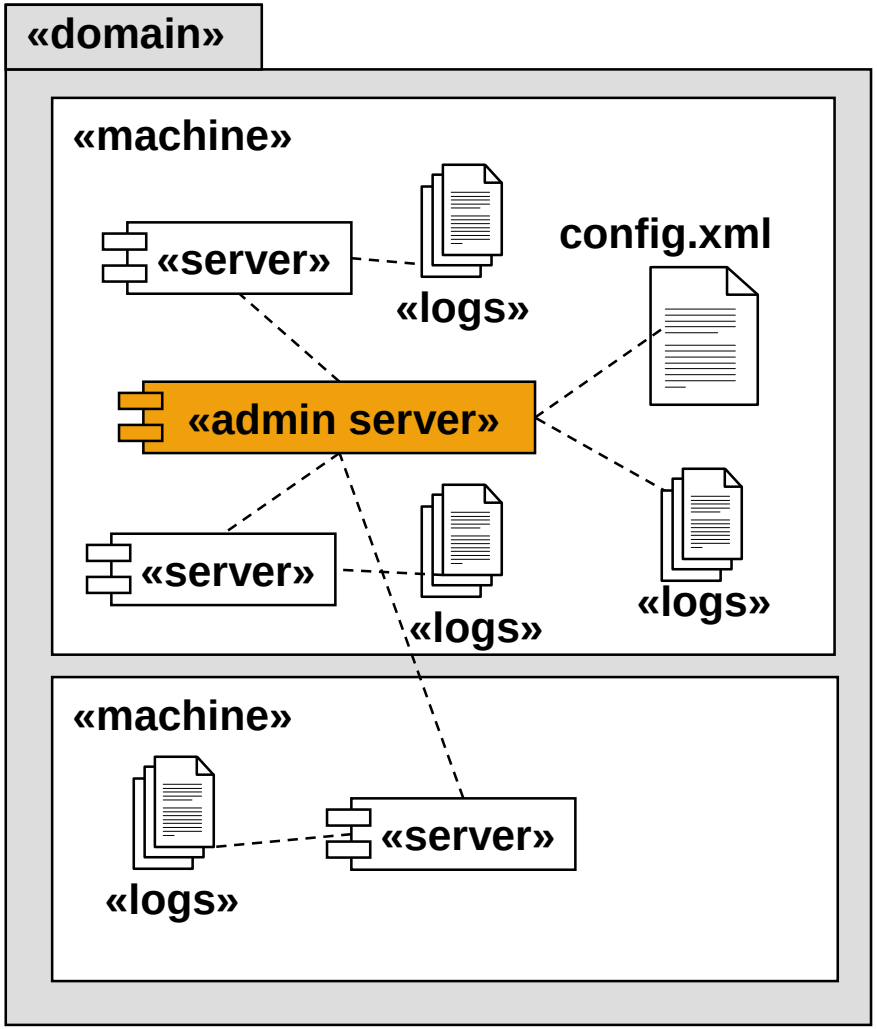
Definition: Server

- A *server* is an instance of `weblogic.Server` executing in a JVM.
- A server:
  - Runs on a designated WLS machine
  - Has a dedicated amount of RAM
  - Is multi-threaded



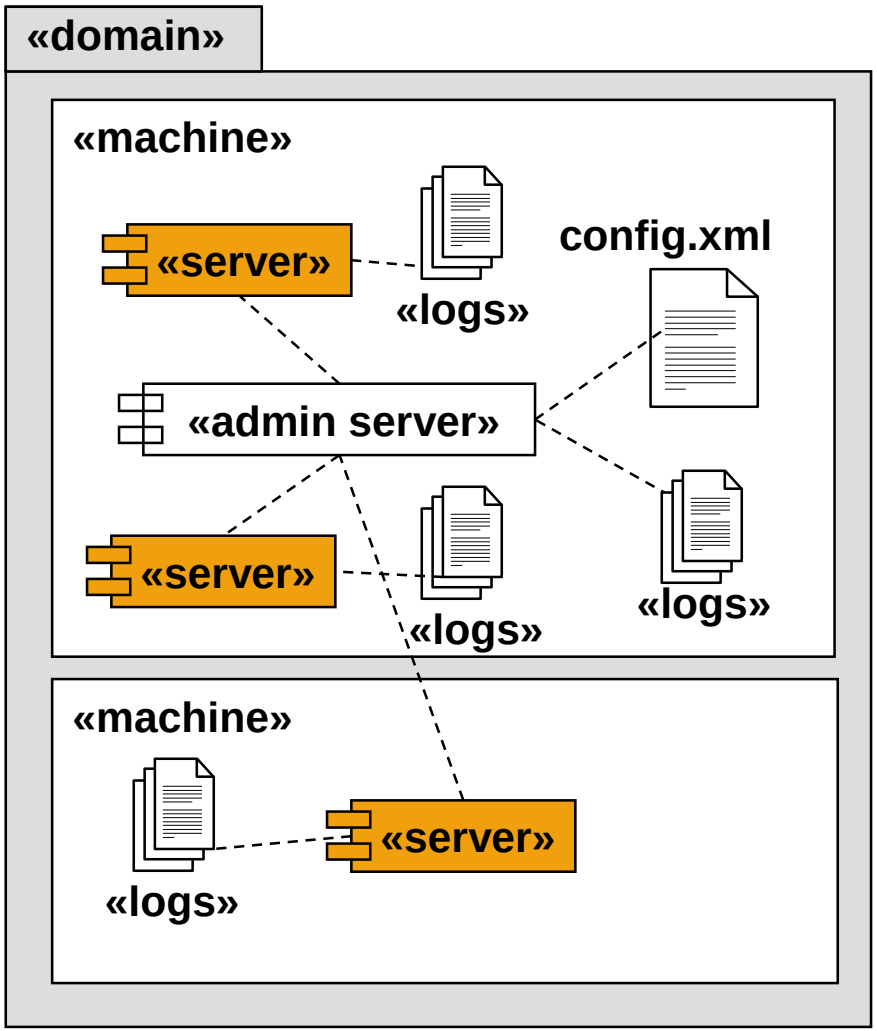
Definition: Administration Server

- An *administration (admin) server* is the central point of control for a domain.
- An admin server:
  - Stores the configuration information and logs for a domain
  - Runs the WebLogic administration console



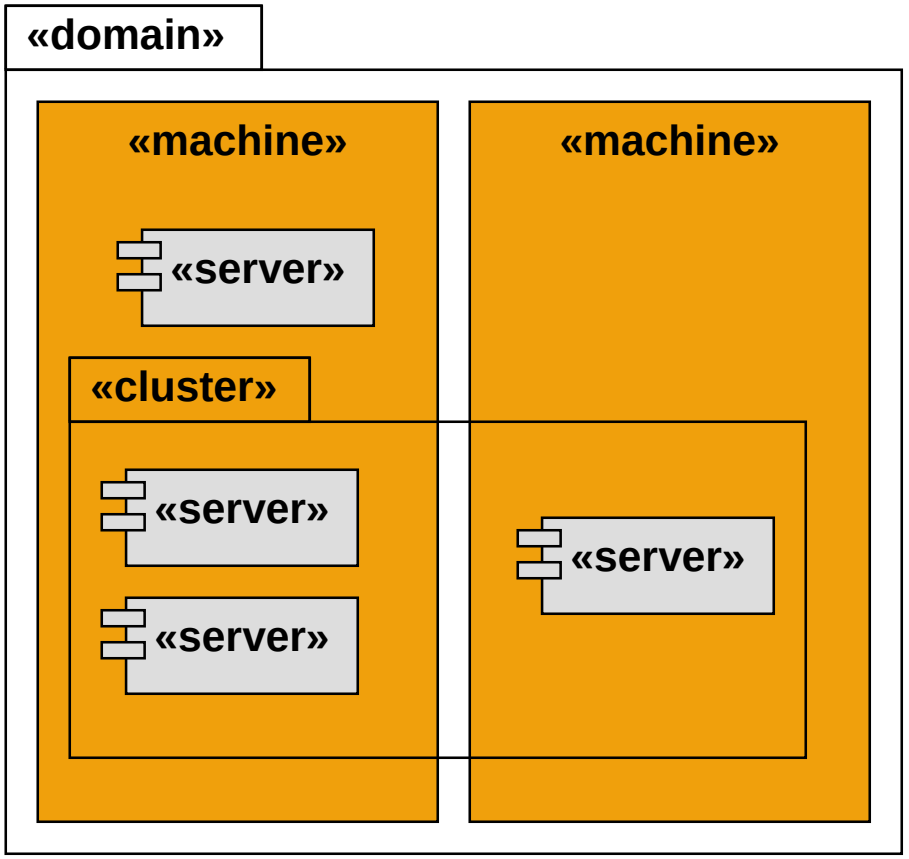
Definition: Managed Server

- A *managed server* is any server in a domain that is *not* the admin server.
- A managed server:
  - contacts the admin server for configuration information
  - Runs business applications in a production environment



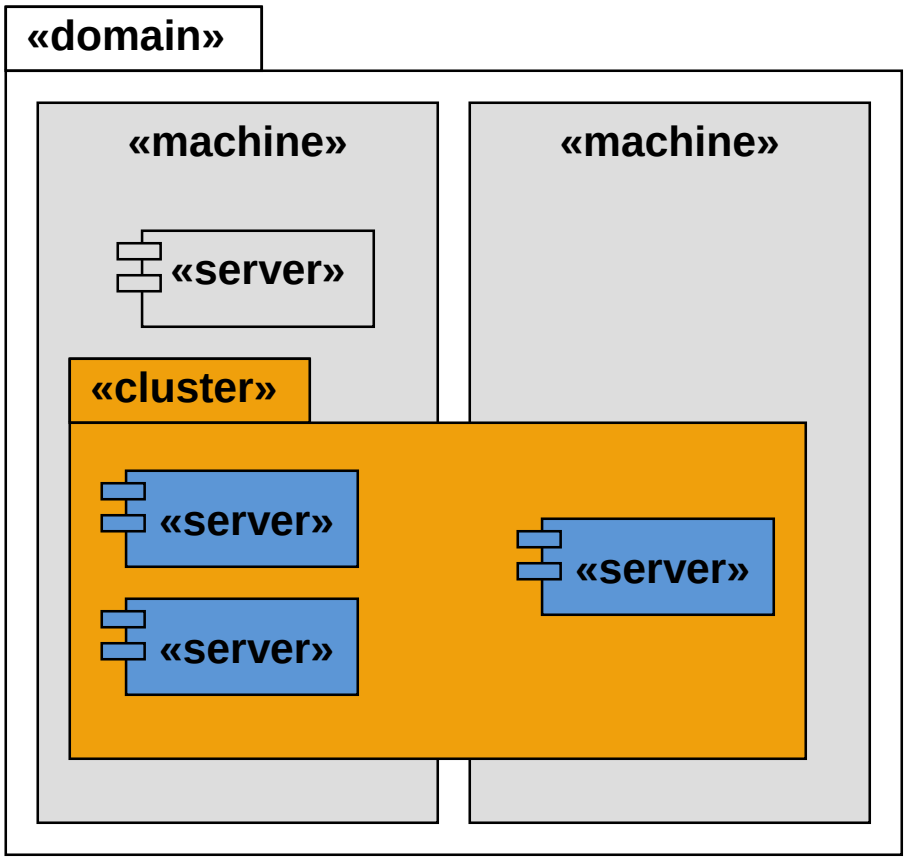
Definition: Machine

- A *machine* is a computer that hosts WebLogic Server(s).
- A machine:
  - Runs a supported operating system platform
  - Can host multiple WebLogic Server instances



Definition: Cluster

- A *cluster* is a logical group of WLS servers.
- WebLogic clusters provide automatic:
  - Fault tolerance
  - High Availability
  - Load-balancing
- A cluster is transparent to a client.



Section Review

In this section we discussed:

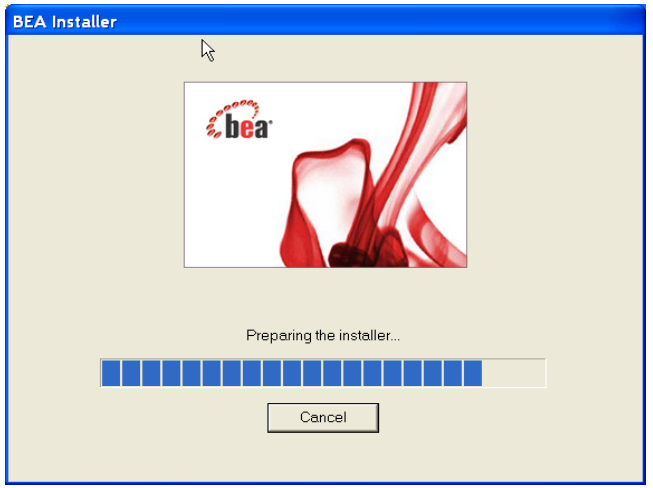
- ✓ How distributed systems improve availability, scalability, and maintainability
- ✓ How standards for distributed systems improve cost effectiveness of software development projects
- ✓ The J2EE architecture & many J2EE technologies
- ✓ Terms used to discuss Web architectures
- ✓ Terms used to describe WebLogic Server features

Road Map

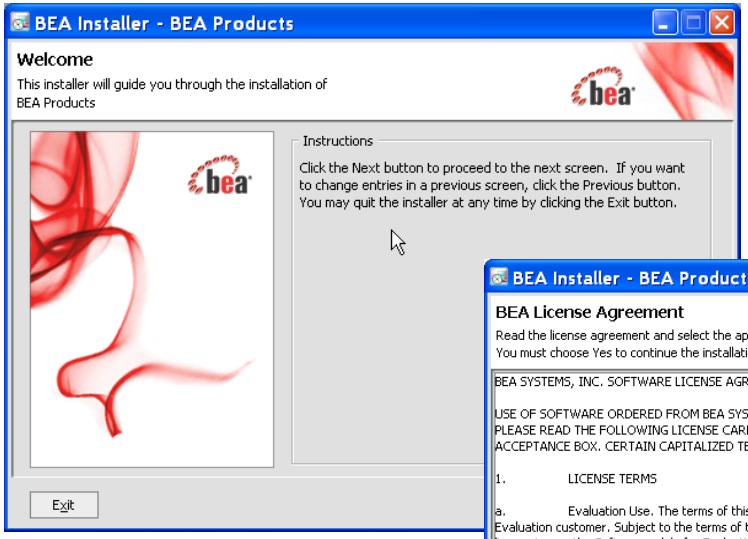
1. Distributed Architecture
2. **Setting Up a WebLogic Server Environment**
   - WebLogic Server Architecture
   - Installing & Running WebLogic Server

WebLogic Server Installation

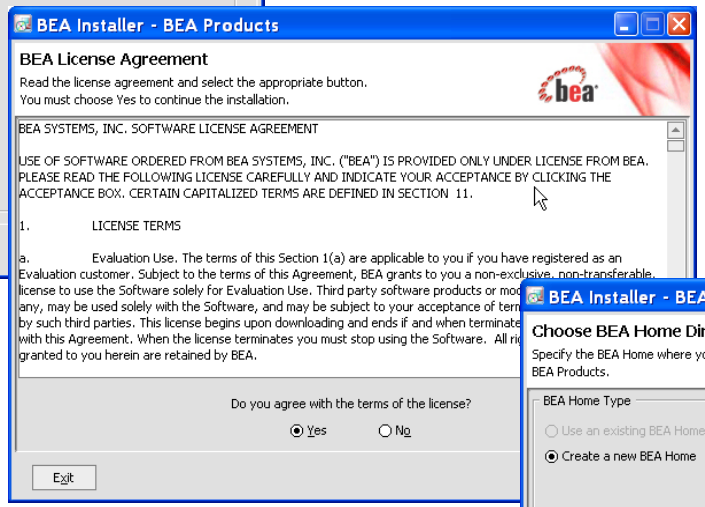
- WebLogic Server can be installed in three different ways:
  - Graphical User Interface Mode (GUI)
  - Console Mode
  - Silent Mode
- The BEA installer program supports a number of platforms including:
  - Windows 2000, 2003 Server, XP
  - Sun Solaris
  - HP-UX
  - Linux



GUI Mode Installation



Read the welcome screen

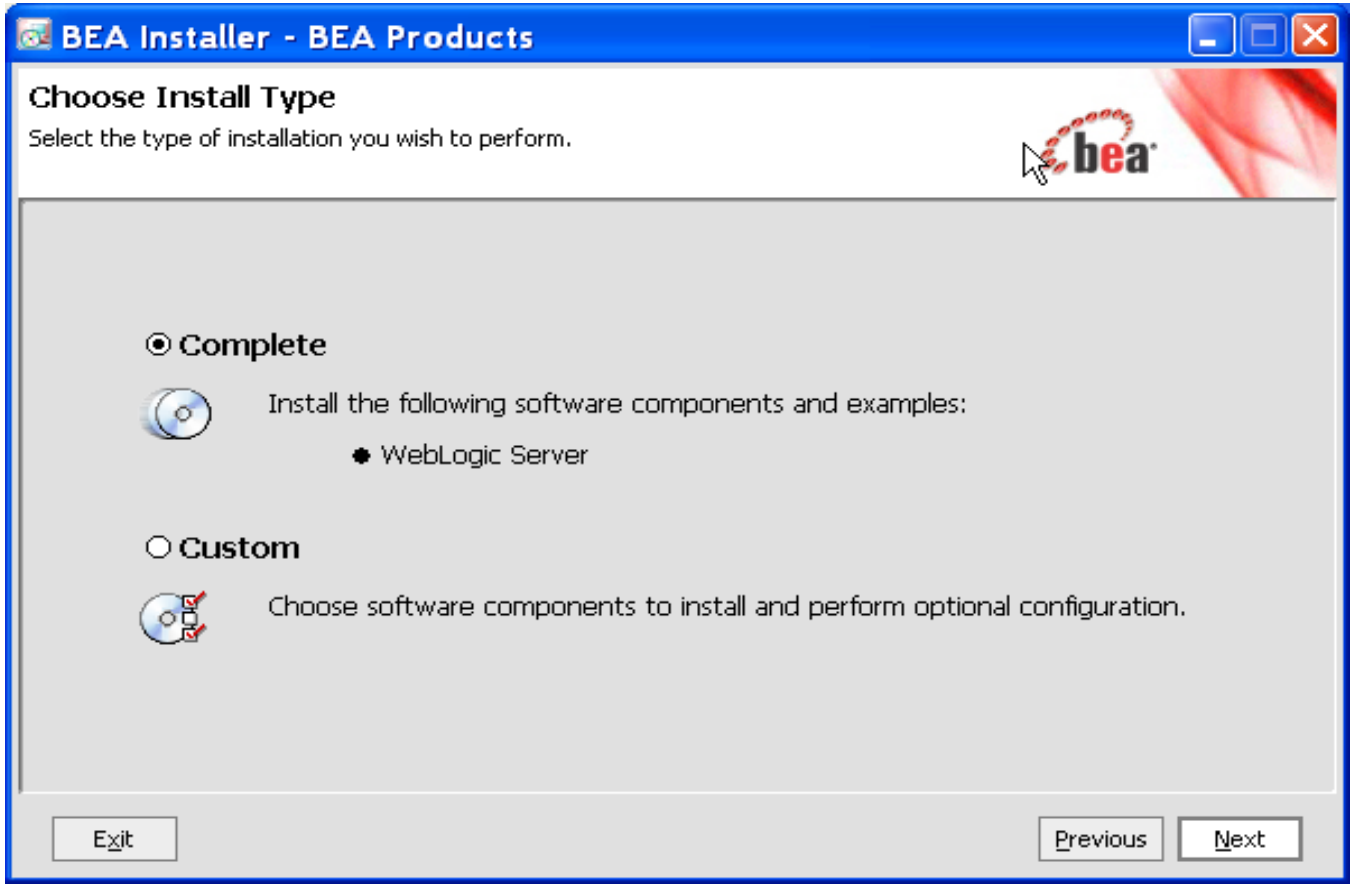


Accept the License Agreement

Choose the Home directory

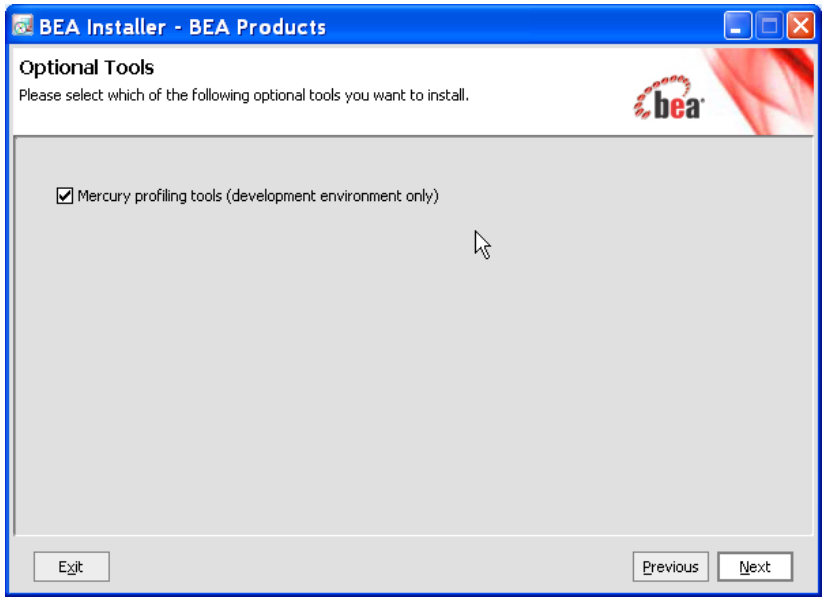
Choose an Install Type

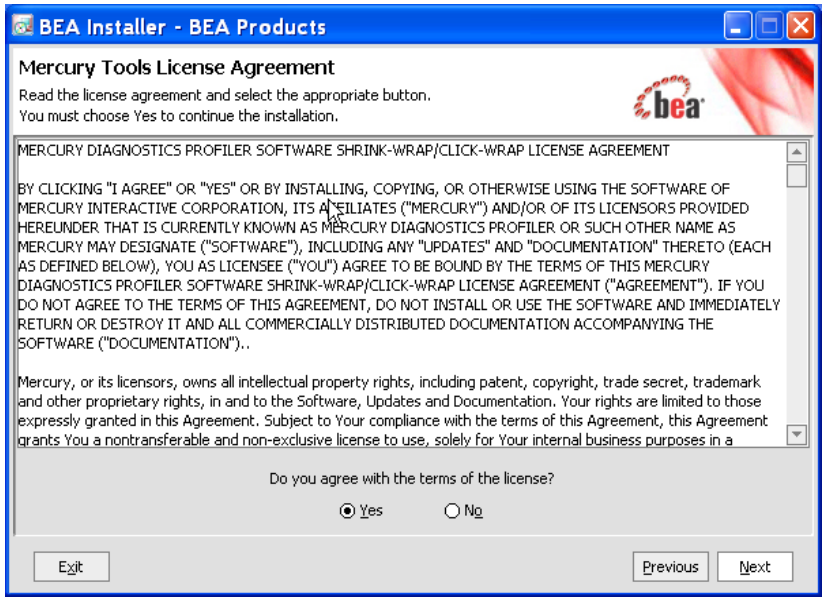
- Select the software to be installed on your system.



Select Optional Tools

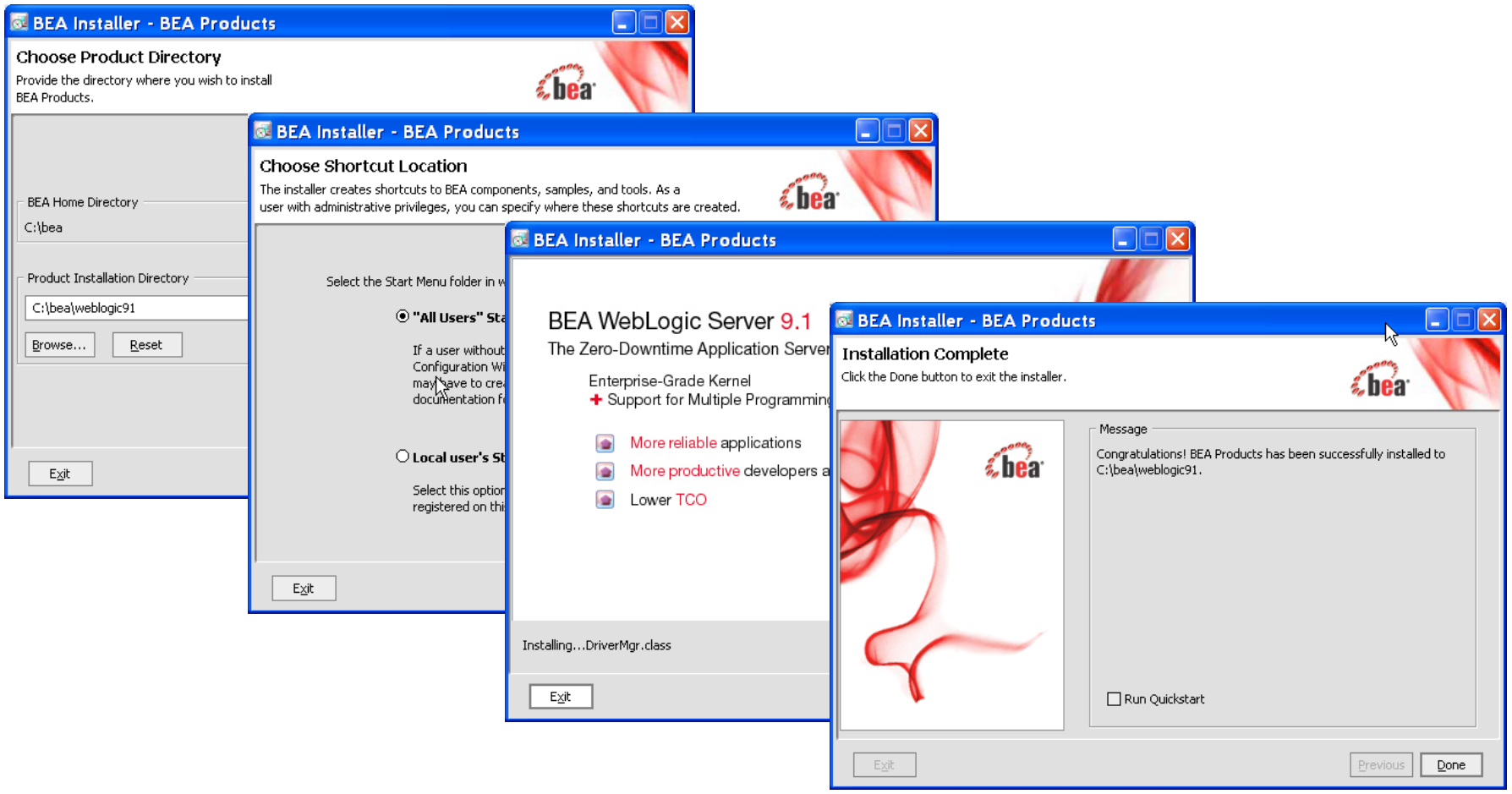
- Choose Mercury profiling tool as an option only for the development environment





Choose the Product Directory

- Choose the target directory for WebLogic Server 9.1

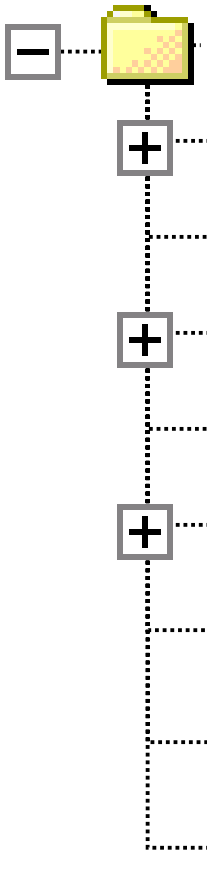


Installation complete!

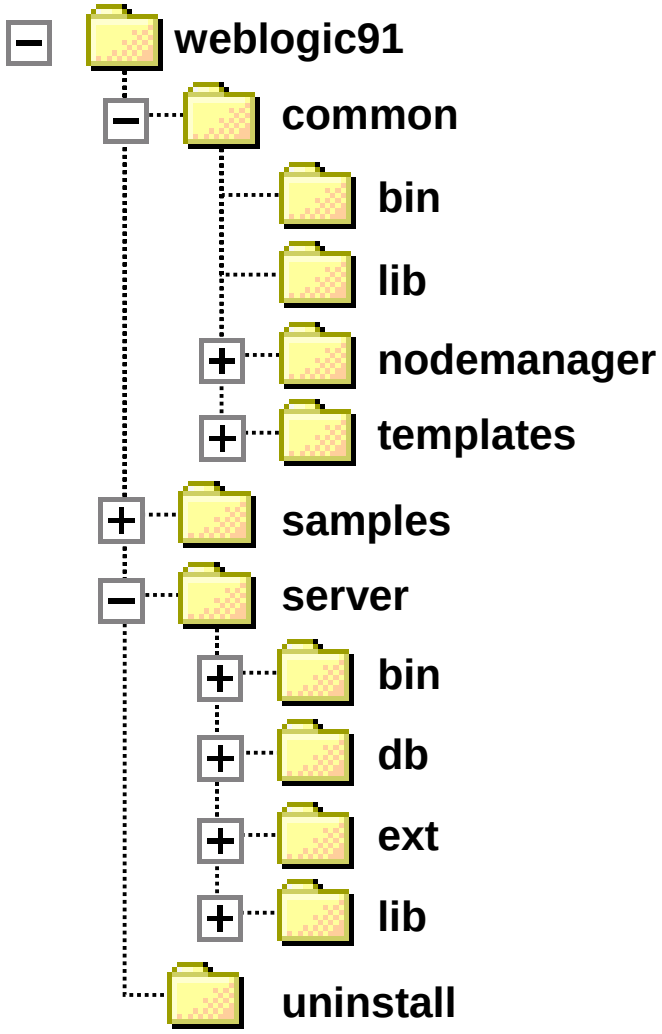
Console Mode Installation

- Console-mode installation is the text-based method of executing the BEA installation program.
- A WebLogic Server installer for a UNIX platform takes one of two forms:
  - a UNIX-specific Java installer which includes JDK 1.5.0_04
    - has a filename ending in .bin
  - a platform-independent Java installer without a JDK
    - has a filename ending in .jar
- The installation steps are similar to GUI-based installation.

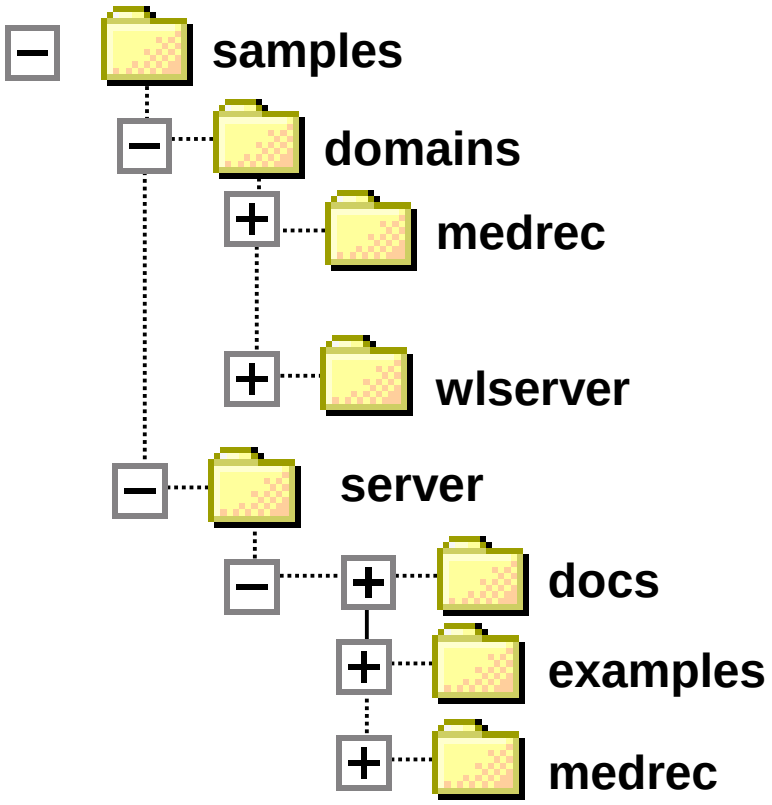
Post Installation: BEA Directory

Directory and Files	Description
 bea	BEA Home directory
jdk150_04	Prepackaged 1.5.0_04 JDK/JRE
logs	Install history of BEA products
user_projects	Default location of user domains
utils	Additional/utility JAR files
weblogic91	WebLogic Server home directory
license.bea	License file
registry.xml	Record of all installed BEA products
UpdateLicense.cmd	Updates license.bea file

WebLogic Directory Structure

Directory / File	Description
	Contains WLS 9.1 product components
	Contains files shared by WLS 9.1 components including template JAR files used by the Configuration Wizard when creating domains
	Contains sample code and resources
	Contains server software components
	Contains executables
	Oracle Database ddl files for v8.1.7 & v9.2.0
	XML JAR files
	WebLogic Server JAR files
	Code required for uninstalling WLS 9.1

Samples Directory Structure

Directory / File	Description
	Contains sample code and resources Sample domains Sample domain for medrec application Sample domain for wl_server application Contains source code for sample domain examples installed with WebLogic Server

JVM Run-Time Arguments

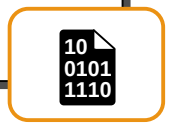
- WebLogic Server can be executed with most Java Virtual Machines.
- WebLogic Server supports JDK 1.5.0

Syntax for running a virtual machine:

```
java options FullyQualifiedJavaClass ProgramOptions
```

Some virtual machine options:

`-Xms`	The minimum size of the dynamic heap.
`-Xmx`	The maximum size of the dynamic heap.
`-D`*`prop=val`*	Defines an environment variable accessible by the program.
`-classpath` *`classpath`*	Specifies list of files/directories containing dependent classes.



WebLogic Server Dependencies

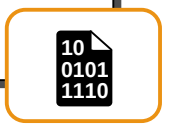
- To run WLS, you must configure:
  - PATH to include all executable programs (including the Java interpreter)
  - CLASSPATH to include dependencies
- These parameters can be set:
  - in your computer's environment settings
  - in a custom batch file or shell script

To see exhaustive list of DOS environment properties:

```
set
```

To set a DOS environment variable:

```
set VAR_NAME=VALUE
```



Configuring Your *CLASSPATH*

- The WLS CLASSPATH:
  - is completely configured by the Java system CLASSPATH environment variable

Files that must be in the CLASSPATH:

```
%WL_HOME%/server/lib/weblogic.jar
```

Any additional service pack jar files (See release notes)

Files that can be in the CLASSPATH:

```
%WL_HOME%/common/eval/pointbase/lib/pbclient51.jar
%WL_HOME%/common/eval/pointbase/lib/pbtools51.jar
%WL_HOME%/common/eval/pointbase/lib/pbembedded51.jar
%WL_HOME%/server/lib/xmlx.jar
```

JDBC drivers
Startup classes, shutdown classes
3rd-party libraries
Other common classes

Starting WebLogic Server

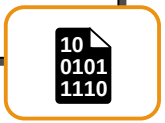
- WebLogic Server is started by running the `weblogic.Server` class.

Minimal syntax:

```
java -server –Xms256m –Xmx512m -classpath "%CLASSPATH%"
  -Dweblogic.Name=%SERVER_NAME% -Dplatform.home=%WL_HOME%
  -Dweblogic.management.username=%WLS_USER%
  -Dweblogic.management.password=%WLS_PW%
  -Dweblogic.ProductionModeEnabled=%STARTMODE%
  -Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy
   weblogic.Server
```

Arguments:

%SERVER_NAME% - The name of the server to start.



Example: Starting WLS Directly

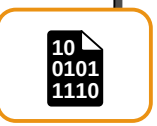
To start WebLogic Server by hand (Windows):

```
java -server –Xms256m –Xmx512m -classpath "%CLASSPATH%"
  -Dweblogic.Name=myServer –Dplatform.home=C:\bea\weblogic91
  -Dweblogic.management.username=system
  -Dweblogic.management.password=weblogic
  -Dweblogic.ProductionModeEnabled=false
  -Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy
   weblogic.Server
```

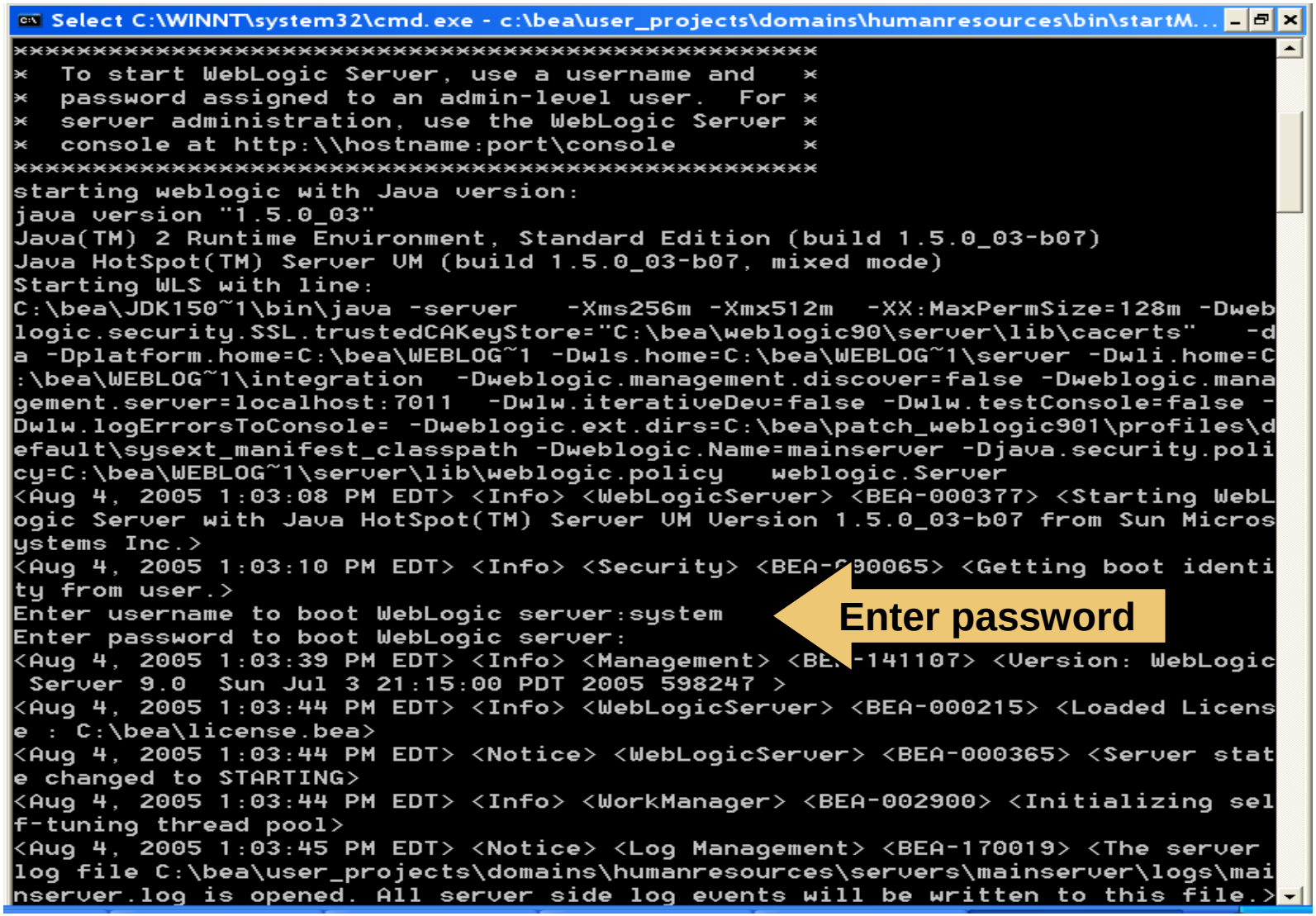
To start WebLogic using the default domain script (Windows):

```
c:\>cd bea\user_projects\domains\someDomain
c:\...>startWebLogic.cmd
```

Execute the start script from the appropriate directory!



Initial Output



Section Review

In this section we discussed:

- ✓ WebLogic Server product overview
- ✓ Installing & Running WebLogic Server
- ✓ Installing WebLogic Server
- ✓ The WebLogic Server directory structure
- ✓ How to configure the CLASSPATH variable for WebLogic Server
- ✓ How to start WebLogic Server from the command-line

Exercise

Install BEA Software

- In this lab will install WLS and set up you working environment. You will also run WLS for the first time.
- For details on the exercise, refer to the Lab Guide.
- If questions arise, ask the instructor.
- The instructor will determine the stop time.

Exercise

Setup the BEA Ed.Lab Environment

- For details on the exercise, refer to the Lab Guide.
- If questions arise, ask the instructor.
- The instructor will determine the stop time.

Module Review

In this module we discussed:

- ✓ Distributed architecture and J2EE technologies
- ✓ Web and WebLogic terminology
- ✓ WebLogic Server in a Web-based distributed system
- ✓ How to Install and configure WebLogic Server