

10

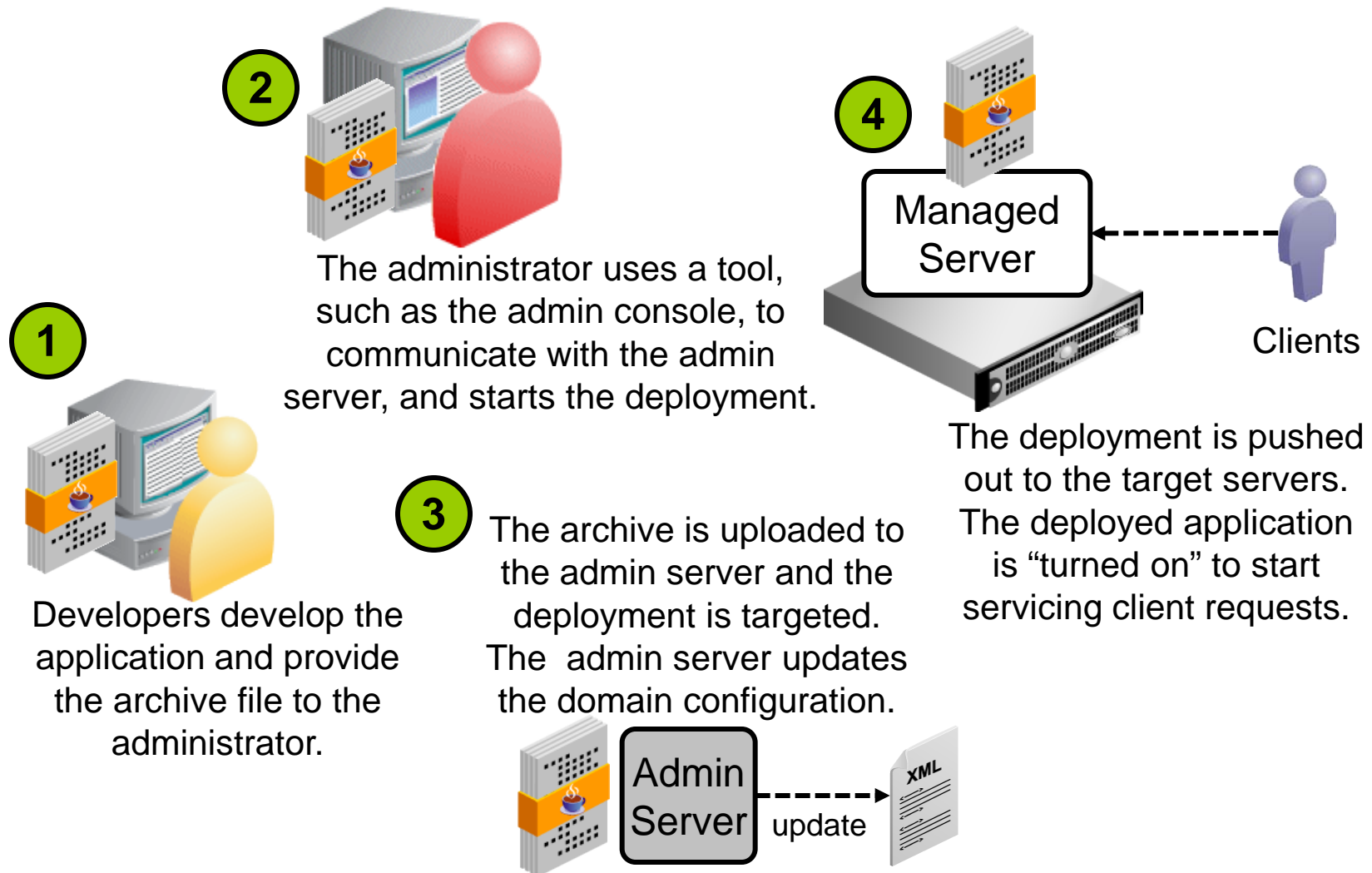
Deploying Applications

Objectives

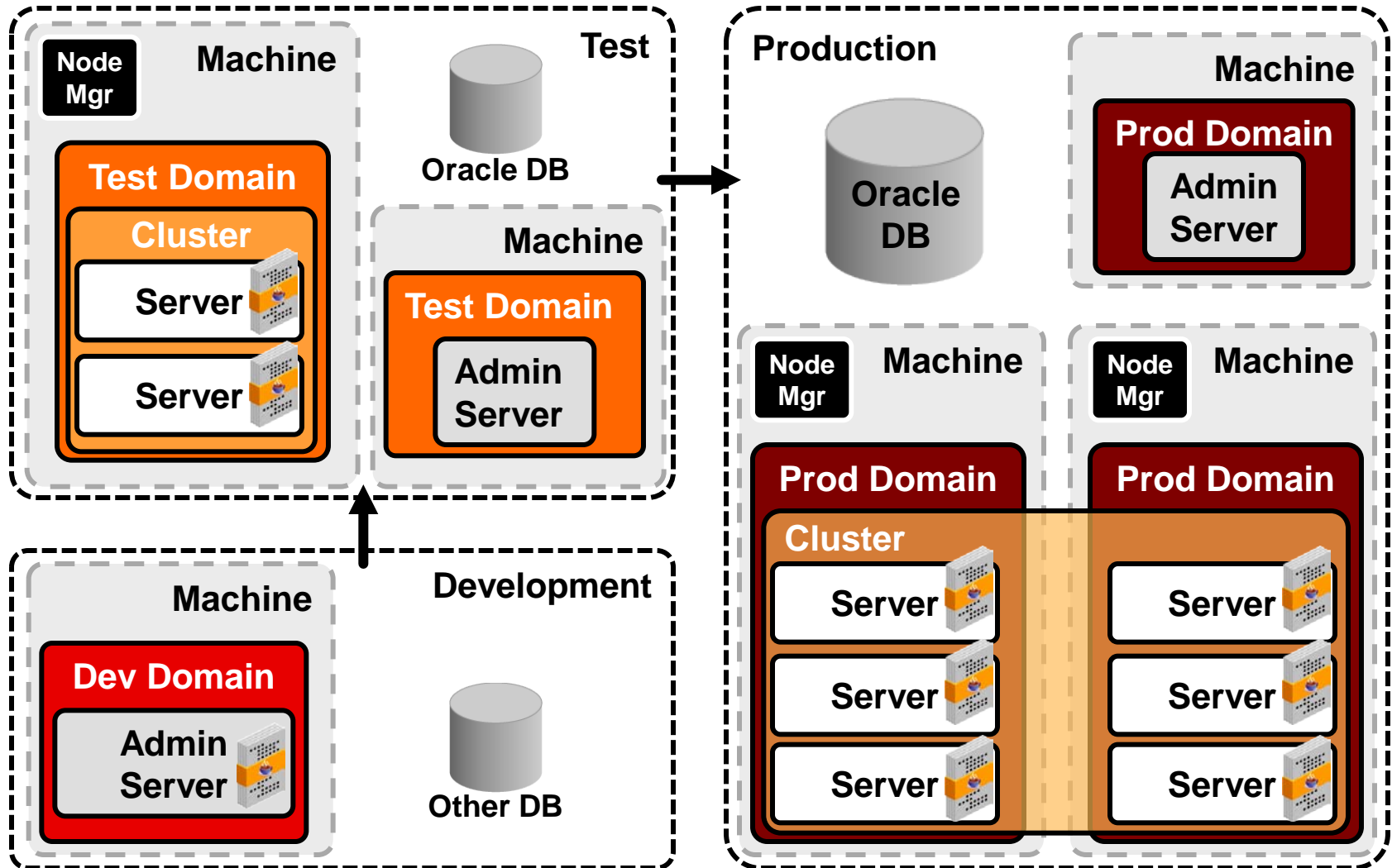
After completing this lesson, you should be able to:

- Deploy an application
- Test a deployed application
- Monitor a deployed application
- Load test an application

Deploying Applications to WebLogic Server



Software Life Cycle and WebLogic Server



Java EE Deployments

- Java Platform, Enterprise Edition deployment units:

Java EE Deployment	Archive Name	Archive File Extension
Web application	Web archive	.war
Enterprise JavaBeans (EJBs)	EJB Java archive	.jar
Web service	Web archive or EJB JAR	.war / .jar
Resource adapter	Resource archive	.rar
Optional package	Java archive	.jar
Enterprise application	Enterprise archive	.ear

A collection of web applications, EJBs, and resource adapters

WebLogic Server Deployments

- WebLogic-specific deployment units:
 - Enterprise applications, web applications, or Java archives as shared libraries
 - JMS and JDBC modules

WLS Deployment	Archive Name	File Extension
Shared library	Enterprise application, web application, or Java archive	.ear / .war / .jar
JMS module/JDBC module	XML file	.xml

These can be included in an enterprise application.

Other Deployments

Product	Deployment	Archive Name	Archive File Extension
Oracle Application Development Framework (ADF)	ADF application	Enterprise application	.ear
Oracle SOA Suite	SOA Suite composite application (SAR stands for SOA archive)	Single composite application or multiple composite applications	Single: .jar or .sar Multiple: .zip
Oracle WebCenter	WebCenter application	Enterprise application	.ear
Oracle Coherence	Coherence artifacts	Grid archive	.gar

Deployment Terms

- **Deploy:**

1. Developers provide application files, usually as an archive, to an administrator, who moves the file to a desired location.
2. An administrator adds the application to the domain configuration and target servers (or clusters). The deployment is distributed to those servers.
3. An administrator starts the application (so the application starts servicing requests).

- **Undeploy:**

1. An administrator stops the application (making it unavailable to clients).
2. An administrator removes the application from the configuration.

Deployment Terms

- **Redeploy:**
 1. Developers provide a new version of the application.
 2. An administrator copies over the deployment files (or archive file) of the deployed application with the new version.
 3. An administrator deploys the new version of the application.
 - Note that with redeployment there is no explicit “start the application” step, because the application is automatically started.
 - WebLogic Server has a strategy called “Production Redeployment” that allows both versions of the application to be active simultaneously.
- **Distribute:** Similar to **deploy**, but the application is not started. The application is pushed out to its targets, ready to be started later.

Deployment Descriptors

- **Deployment descriptor:** An XML file packaged within a deployment that sets properties of the application
 - Each Java EE deployment has both a standard and a WebLogic-specific deployment descriptor.
 - Since Java EE 5, deployment descriptors can be replaced with annotations in the application code.

web.xml

```
...  
<web-app ... >  
  <servlet>  
    <servlet-name>  
      BenefitsServlet  
    </servlet-name>  
    <servlet-class>  
      stcurr.Benefi  
  ...
```

weblogic.xml

```
...  
<weblogic-web-app ...>  
  <session-descriptor>  
    <cookie-path>  
      benefits  
    </cookie-path>  
  </session-descriptor>  
  <web-root>  
    ...
```

Developers are responsible for creating the deployment descriptors (or code annotations) as part of application development.

Deployment Plans

- A WebLogic Server deployment plan is an XML document that can override an application's configuration (deployment descriptors). A deployment plan is:
 - Optional
 - Used to update deployment descriptor values
 - Useful when moving an application from one environment to another (such as from development to test or test to production)
 - A separate file, outside of the deployment archive
`plan.xml`

```
<?xml version='1.0' encoding='UTF-8'?>
<deployment-plan ... >
  <application-name>timeoff.war</application-name>
  <variable-definition>
    <variable> ...
```

Exploded Versus Archived Applications

- An application can be deployed as a set of directories and files. This is called an “exploded directory” application.
- An application file and directory structure can be placed within the appropriate archive file, and that single file can be deployed.
- Exploded directory applications are most often used during development, and archived applications during test and production.
 - There is nothing that prevents exploded application deployments in test and production or archive file deployments during development, however.

Autodeploy

- A development mode domain can automatically deploy applications:
 1. Place the application's exploded directories and files or archive file in the domain's `autodeploy` directory.
 2. The administration server watches that directory. When it detects a new application, it automatically:
 - A. Adds the application to the configuration
 - B. Targets the application to itself, the admin server
 - C. Starts the application (the application starts servicing requests)
- Autodeploy is a convenient feature for developers, which allows them to quickly deploy and test an application.
- Autodeploy is disabled in a production mode domain.

Server Staging Mode

The Staging Mode of a server determines how the server accesses deployed applications:

- **stage:** During the deployment process the application is pushed out to the remote server's Staging Directory.
- **nostage:** The server accesses the application from an accessible location (often shared storage). You must ensure the application is placed in this location. The location is specified during deployment.
- **external_stage:** Similar to `stage`, except the application must be copied into the Staging Directory before beginning the deployment process (manually or by some other tool).

The Staging Mode's default is `stage`.



WebLogic Server Deployment Tools

The following tools can be used to deploy applications:

- Administration console
- WLST
 - Can be used interactively or to run a script
 - Example:

```
deploy('app', '/apps/app.ear', targets='cluster1')
```

- The `weblogic.Deployer` class
 - For command-line deployment
 - Example:

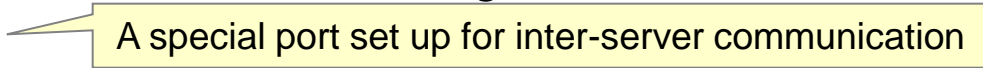
```
java weblogic.Deployer  
  -adminurl http://host01.example.com:7001  
  -username weblogic -password Welcome1  
  -deploy -source /apps/app.ear -targets cluster1
```

WebLogic Server Deployment Tools

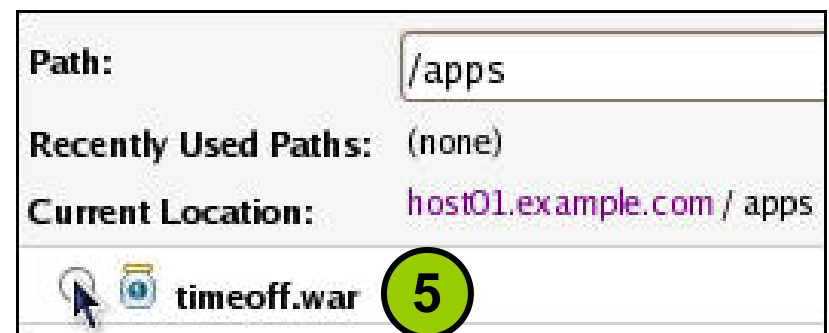
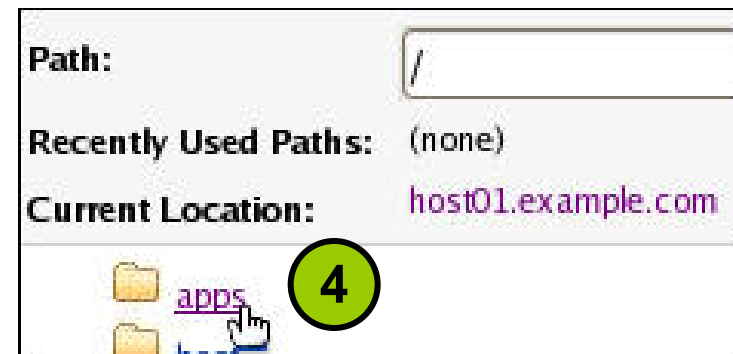
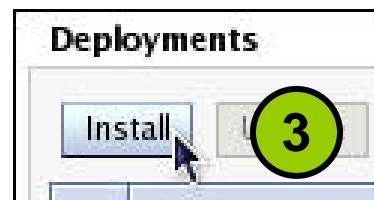
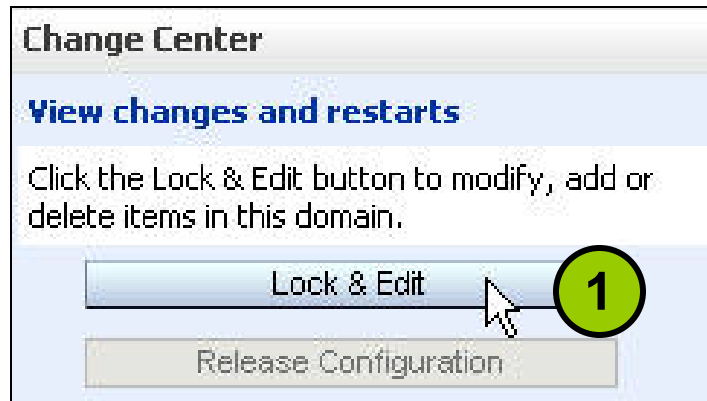
The following tools can be used to deploy applications:

- Ant
 - It is an Apache open-source, Java-based “make utility”
 - Ant and WebLogic-specific Ant tasks come with the WebLogic Server installation.
 - The `wldeploy` task is the Ant version of the `weblogic.Deployer` utility.
- Maven
 - It is an Apache open-source tool for managing Java projects.
 - A plug-in is available with the WebLogic Server installation.
- Enterprise Manager Cloud Control
 - Cloud Control can deploy, undeploy, and redeploy applications to WebLogic Server.

Starting and Stopping an Application

- Newly deployed applications must be started. There are two possible “start levels:”
 - Start servicing all requests: This gives regular users access to the application
 - Start servicing only administration requests: The application is available only to administrators through the domain administration port A special port set up for inter-server communication
- Applications must be stopped before they are undeployed.
 - When work completes: Allows current users of the application to complete their work and disconnect
 - Force stop now: Stops the application immediately, whether or not it is being used

Deploying an Application



Deploying an Application

Path:

Recently Used Paths: (none)

Current Location:

  timeoff.war

Back Next **6** Cancel

Install Application Assistant **7**

Back Next Finish Cancel

Choose targeting style

Targets are the servers, clusters, and virtual hosts of the application.

☒ **Install this deployment as an application**

Servers **8**

☐ AdminServer

Clusters

☒ cluster1

☒ All servers in the cluster

☐ Part of the cluster

☐ server2

☐ server1

Back Next Finish Cancel

Deploying an Application

Security roles (who) and policies (what they can do) for the application are determined by the deployment descriptors.

Security roles are created by using the admin console. Security policies are defined in the deployment descriptors.

Security roles and policies are created by using the admin console.

This option is provided for backward compatibility with WebLogic Server versions before 9.0.

Install Application Assistant 9

Back Next Finish Cancel

Optional Settings
You can modify these settings or accept the defaults

General

What do you want to name this deployment?

Name: timeoff

Security

What security model do you want to use with this application?

- ☒ **DD Only: Use only roles and policies that are defined in the deployment descriptors**
- ☐ **Custom Roles: Use roles that are defined in the application**
- ☐ **Custom Roles and Policies: Use only roles and policies defined in the application**
- ☐ **Advanced: Use a custom model that you have defined**

Source Accessibility

How should the source files be made accessible?

- ☒ **Use the defaults defined by the deployment's deployment descriptor**

Deploying an Application

The screenshot displays the Oracle WebLogic Deployer interface during the deployment of an application. The 'Install Application Assistant' window is the primary focus, with a green circle labeled '10' highlighting the 'Finish' button. Below the buttons, the text 'Review your choices and click Finish' is visible. The 'Additional configuration' section is expanded, showing a radio button selection for 'Yes, take me to the deployment's configuration screen.' The 'Summary' section shows the deployment path '/apps/t' and the application name 'timeoff'. The 'Change Center' window is also visible, with a green circle labeled '11' highlighting the 'Activate Changes' button. The 'Deployments' window is open, showing a table with columns for 'Name' and 'Status'. The 'timeoff' deployment is listed with a status of 'Servicing all requests', which is highlighted by a green circle labeled '12'.

Install Application Assistant

Back | Next | **Finish** | Cancel **10**

Review your choices and click Finish

Click Finish to complete the deployment. This may take a few minutes.

Additional configuration

In order to work successfully, this application may require additional configuration after completing this assistant?

☒ Yes, take me to the deployment's configuration screen.

☐ No, I will review the configuration manually.

Summary

Deployment: /apps/t

Name: timeoff

Change Center

View changes and restarts

Pending changes exist. They must be activated to take effect.

11 **Activate Changes**

Undo All Changes

Deployments **12**

Install | Update | Delete | Start ▼ | Stop ▼

	Name ^
<input type="checkbox"/>	
<input checked="" type="checkbox"/>	timeoff

Servicing all requests **12**

Servicing only administration requests

Undeploying an Application

Note: A running application cannot be undeployed.

The screenshot illustrates the process of undeploying an application in the Oracle WebLogic Console, with four numbered steps:

- Step 1:** In the **Domain Structure** tree on the left, the **Deployments** link under **wlsadmin** is selected.
- Step 2:** In the **Deployments** main panel, the **Stop** button is clicked for the application **timeoff**. The dropdown menu shows the option **Force Stop Now** is selected.
- Step 3:** The **Force Stop Application Assistant** dialog box appears, and the **Yes** button is clicked to confirm the stop action.
- Step 4:** In the **Change Center** panel, the **Lock & Edit** button is clicked to allow configuration changes.

The **Deployments** panel shows a table with the following data:

	Name
<input checked="" type="checkbox"/>	timeoff

The **Change Center** panel includes the following text and buttons:

View changes and restarts

Click the Lock & Edit button to modify, add or delete items in this domain.

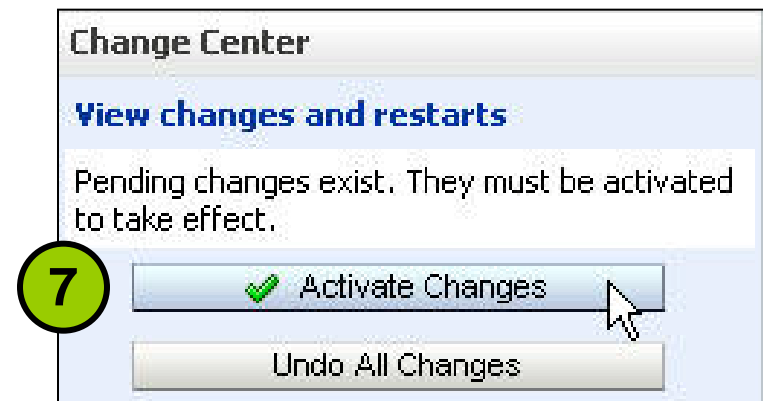
Lock & Edit (highlighted)

Release Configuration

Undeploying an Application



Note that although the deployment is no longer in the configuration, the deployment files (or the archive file) have not been deleted.



Redeploying an Application

1

```
$> cp /uploaded/timeoff.war /apps/timeoff.war
```

Change Center

View changes and restarts

Click the Lock & Edit button to modify, add or delete items in this domain.

Lock & Edit

Release Configuration

2

Domain Structure

wlsadmin

+ Environment

Deployments

+ Services

3

4

Deployments

Install

Update

Delete

Start v

Stop v

<input type="checkbox"/>	Name ^v	State
<input checked="" type="checkbox"/>	+  timeoff	Active

Redeploying an Application

Update Application Assistant 5

Back Next **Finish** Cancel

Locate new deployment files

You have elected to update the timeoff application.

Source path: /apps/timeoff.war Change Path

Deployment plan path: (No value specified) Change Path

It is an error when redeploying to move the deployment files, so do not use this.

You can move (or add) a deployment plan, however.

Change Center

View changes and restarts


Pending changes exist. They must be activated to take effect.

✓ **Activate Changes** 6

Undo All Changes

7

The application does not need to be manually started.

<input type="checkbox"/>	Name ^	State
<input type="checkbox"/>	+  timeoff	Active

Monitoring Deployed Applications: Admin Console

The administration console allows you to monitor applications:

1. Select the name of the deployed application in the Deployments table.
2. Select the **Monitoring** tab.
3. Select the subtab of interest.

Settings for timeoff

Overview Deployment Plan Configuration Security Targets Control Testing **Monitoring** Notes

Web Applications Servlets Sessions PageFlows Workload Web Service Clients JAX-RS Applications

Customize this table

Web Applications

Showing 1 to 2 of 2 Previous | Next

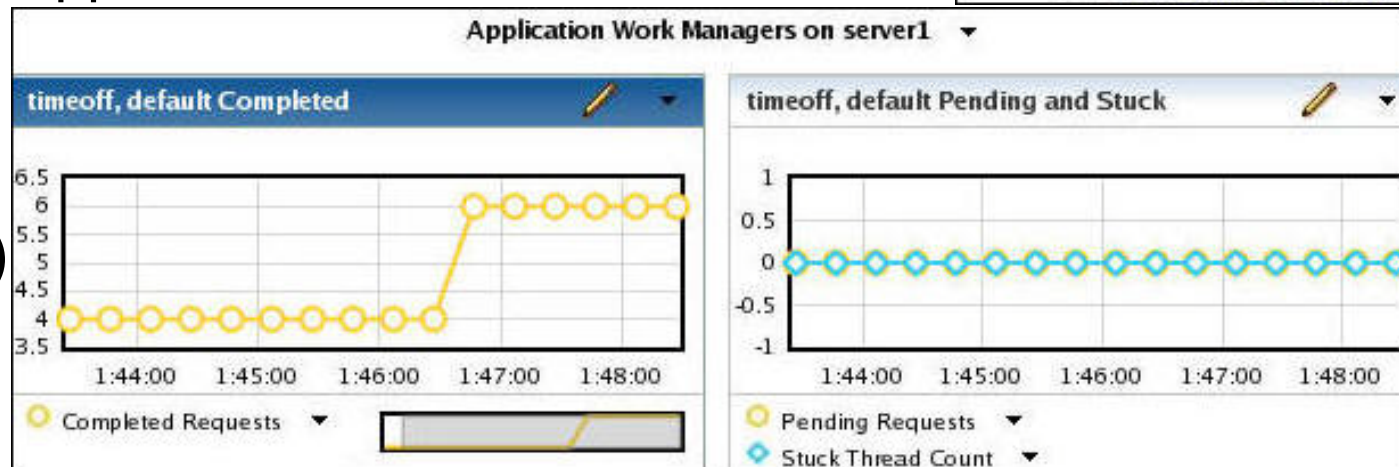
Context Root	Application	Server	Machine	State	Active Server Count	Source Information	Servlets	Sessions
/timeoff	timeoff	server1	machine1	Active	1	timeoff.war	5	0

Monitoring Information Available from the Admin Console

Application Type	Monitoring Information Available
Web application	Targeted servers, context root, number of Servlets, the number of times each Servlet has been invoked, average execution time of each Servlet, the number of active HTTP sessions, creation time of each session, work manager and thread information, and more
EJB	Targeted servers, total number of EJBs that have been activated, current number of beans in use from the pool, total number of times a bean has been accessed from the cache, current number of beans in the cache, for a message-driven bean if it is connected to its destination, and more
Web service	Web service name, number of servers where the service is active, number of service errors, total number of times the service has been invoked, average service response time, and more
Enterprise application	Web application, web service, and EJB monitoring as described above, as well as JDBC and JMS module monitoring

Monitoring Deployed Applications: Monitoring Dashboard

1. In the View List, expand the server and select the built-in view called **Application Work Managers on *servername***.
2. Click the start button.
3. Find the charts for the application of interest.



Application Errors

- WebLogic Server errors generally show up in the server log as messages from some troubled subsystem.
- Application errors often show in the server log as a Java Exception followed by a stack trace.
 - A stack trace shows all the methods that were interrupted when the error occurred.
 - Stack traces are used by developers to track down their faulty code.

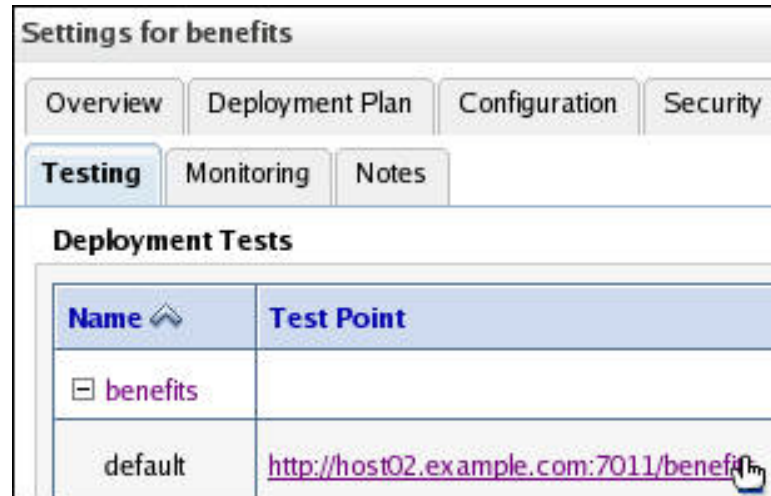
The diagram shows a sample Java stack trace within a gray rectangular box. A yellow callout box labeled "Exception" points to the first line of the trace. Below the trace, five yellow callout boxes with leader lines identify the components of the stack trace entries: "Package" points to "jsr_servlet", "Class" points to "_vision", "Method" points to "_jspService", "Source file" points to "vision.java", and "Line number" points to "7".

```
####<...> <Error> <HTTP>...Servlet failed with  
an Exception  
java.lang.ArrayIndexOutOfBoundsException: 3  
at stcurr.BadClass.causeError(BadClass.java:9)  
at jsr_servlet._vision._jspService(vision.java:7)  
...
```

Package	Class	Method	Source file	Line number
jsr_servlet	_vision	_jspService	vision.java	7

Application Testing

- The administration console displays test URLs based on the application configuration. Select the deployment from the Deployments table, and then click the **Testing** tab.
 - If a link works:
 - The application is deployed and started
 - Remember, however, that this is a minimal test, and it is possible that there are still issues with the application
 - If a link does not work:
 - It could indicate a deployment problem
 - It is also possible that the application is OK and accessed through some other URL



Performance Testing Methodology

1. Define the expected workload.
2. Define performance objectives.
3. Select the subsystems to study.
4. Create and perform a test to create an initial benchmark (baseline performance data).
5. Modify one system attribute.
6. Perform the test again.
7. Analyze the results.
8. Repeat steps 5–7.



Load and Stress Testing

- Load testing measures performance for a system at different levels of concurrent request loads.
- Stress testing measures a system's limits. For example:
 - Extremely high number of concurrent users
 - Extremely high data volume
- Performance testing a WebLogic Server application requires:
 - Measurable performance goals, often stated as Service Level Agreements (SLAs)
 - Example SLA: "The 90th percentile response time for the application under two times a normal load should be within 5 seconds."
 - Load testing tools that can generate the data needed

Load Testing Tools

Many commercial and open-source load testing tools are available, including:

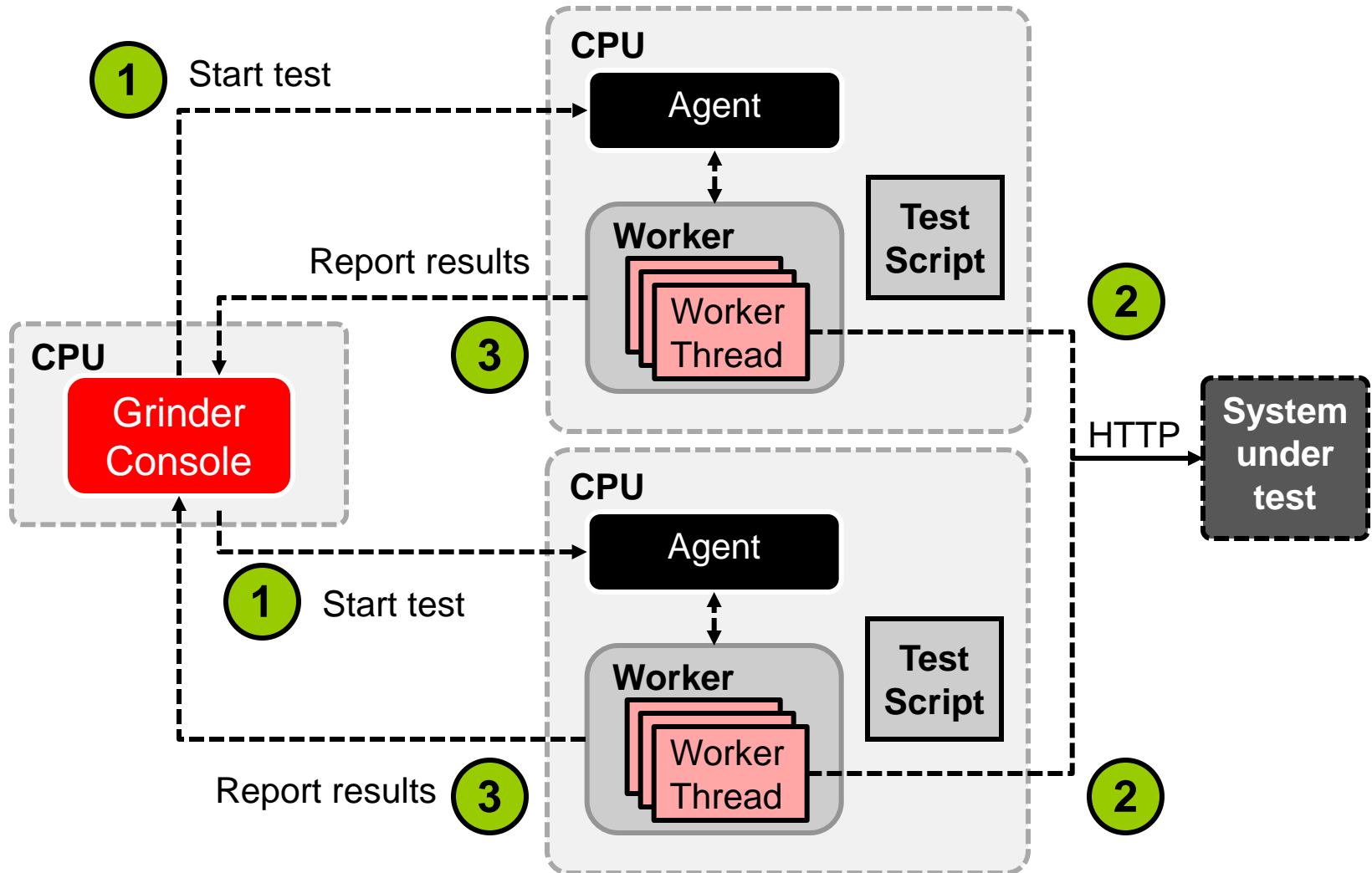
- The Grinder
- JMeter
- HP LoadRunner
- RadView WebLOAD
- Oracle Load Testing (part of Oracle Application Testing Suite)



The Grinder

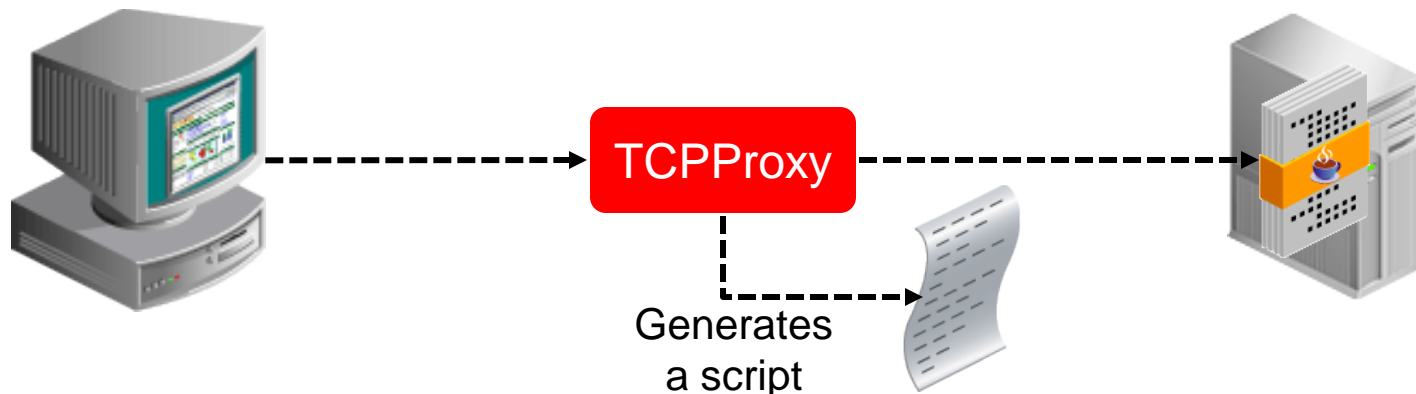
- The Grinder:
 - Is an open source load-testing tool based on Java and Python
 - Supports a distributed, agent-based load simulation model
 - Provides a graphical console to manage agents and view results
 - Supports HTTP/S (links, forms, cookies, and so on) but can be extended to support additional protocols
- A Grinder agent:
 - Uses a specific number of processes and worker threads
 - Executes a supplied test script with each thread

The Grinder Architecture



The Grinder Proxy

- The Grinder tests are Python scripts, which can be:
 - Coded manually
 - Recorded using the TCPProxy
- To use TCPProxy:
 - Configure your web browser to proxy requests through the TCPProxy and then use your web application
 - TCPProxy creates a script that includes all GET and POST requests, cookies, and user “think times”



Agent Properties

- Agents are configured by using the `grinder.properties` file, which has settings for:
 - The location of The Grinder console
 - The test script to run
 - The number of worker processes to start
 - The number of threads to start in each worker process
 - The number of times that each thread should run the test
 - Think time adjustments (speed up or slow down)
 - Output and logging levels
- If The Grinder console is not available when an agent is started, it starts running tests immediately.
- Test script files can also be distributed to agents by using the console.

The Grinder Console

Manage agents

Distribute test scripts

Sample interval: 1000 ms

Ignore 0 samples

Collect samples forever

Collecting samples: 118

36.0 TPS

Total
 120 ms (mean)
 8.94 TPS (mean)
 600 TPS (peak)
 1047 tests
 0 errors

Results for individual requests

Overall results for the entire test

The Grinder Console

File Action Distribute Help

Graphs Results Processes Script

Accumulated test statistics

Test	Description	Successful Tests	Errors	Mean Time	Mean Time Standard Deviation	TPS	Peak TPS	Mean Response Length	Response Bytes Per Second	Response Errors	Mean time to resolve host	Mean time to establish connection	Mean time to first byte
Test ...	Page 1	200	0	614	78.1	1.71	200	0.00	0.00	0			0.00
Test ...	GET c...	200	0	180	86.5	1.71	200	277	473	0	0.0100	2.70	174
Test ...	GET /	200	0	194	79.0	1.71	200	405	692	0			190
Test ...	GET s...	200	0	198	60.9	1.71	200	532	909	0			195
Test ...	Page 2	200	0	41.6	61.8	1.71	96.0	0.00	0.00	0			0.00
Test ...	GET d...	200	0	41.3	61.8	1.71	96.0	2970	5070	0			40.7
Test ...	Page 3	200	0	14.5	23.4	1.71	70.0	0.00	0.00	0			0.00
Test ...	GET d...	200	0	14.2	23.4	1.71	70.0	1480	2520	0			13.9
Test ...	Page 4	47	0	12.3	3.03	0.402	36.0	0.00	0.00	0			0.00
Test ...	GET d...	47	0	12.1	3.08	0.402	36.0	2970	1190	0			11.9
Total		1047	0	120	104	8.94	600	1210	10900	0	0.0100	2.70	118

Finding Bottlenecks

- A CPU-bound system cannot process additional workload because the processor is too busy (at or near 100%).
 - Possible causes include:
 - Too frequent garbage collection
 - Excessive memory allocation—resulting in paging
- An I/O bound system's processor is not fully utilized (< 75%). The performance remains the same regardless of the client load.
 - Common culprits include:
 - Accessing remote disks too frequently
 - Database issues
 - Too few connections, poorly written queries
 - Insufficient network bandwidth



Correcting Bottlenecks

Issue	Resolution
Garbage collection	Try changing JVM garbage collection options, such as the type of collector or the size of the generations.
Memory	Try modifying JVM memory arguments.
Code performance	Use a Java profiler to find the methods that run most often and take the longest to run. Developers should make those methods more efficient.
Web application performance	Precompile JSPs. Ensure that Servlet Reload Check, Resource Reload Check, and JSP Page Check all have the value of -1. Session replication takes resources, developers should use the session sparingly.
Database performance	Developers should increase the efficiency of their queries. DBAs should tune the database.

Quiz

An application must be stopped to be undeployed.

- a. True
- b. False

Quiz

A redeployed application must be manually restarted before it can be used.

- a. True
- b. False

Summary

In this lesson, you should have learned how to:

- Deploy an application
- Test a deployed application
- Monitor a deployed application
- Load test an application

Practice 10-1 Overview: Deploying an Application

This practice covers the following topics:

- Deploying an application
- Redeploying an application
- Undeploying an application

Practice 10-2 Overview: Load Testing an Application

This practice covers the following topics:

- Using The Grinder to load test WebLogic Server
- Viewing the load test results in The Grinder console