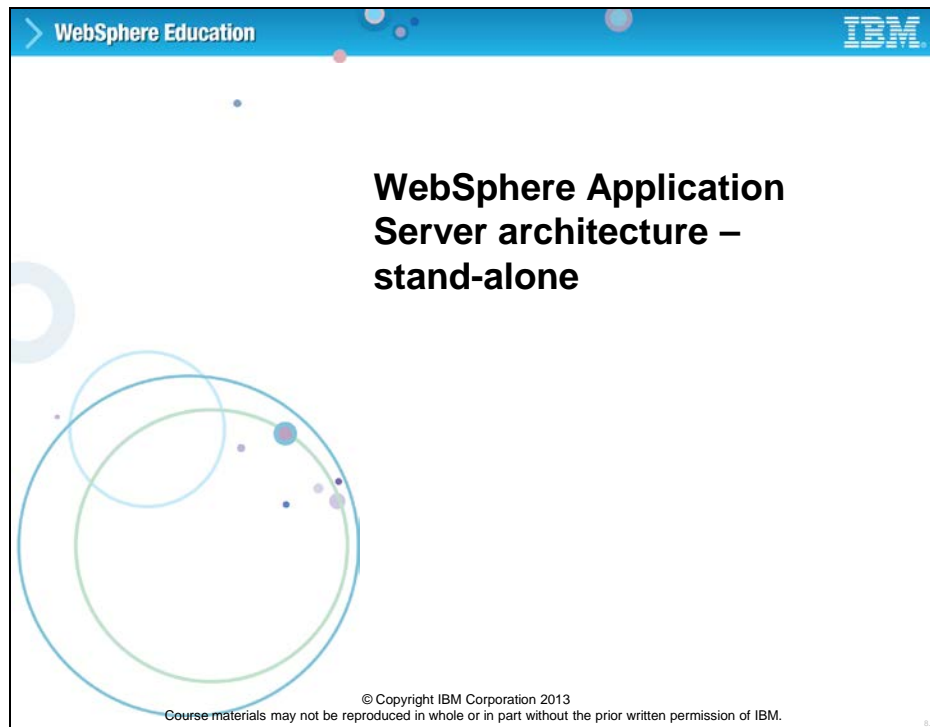
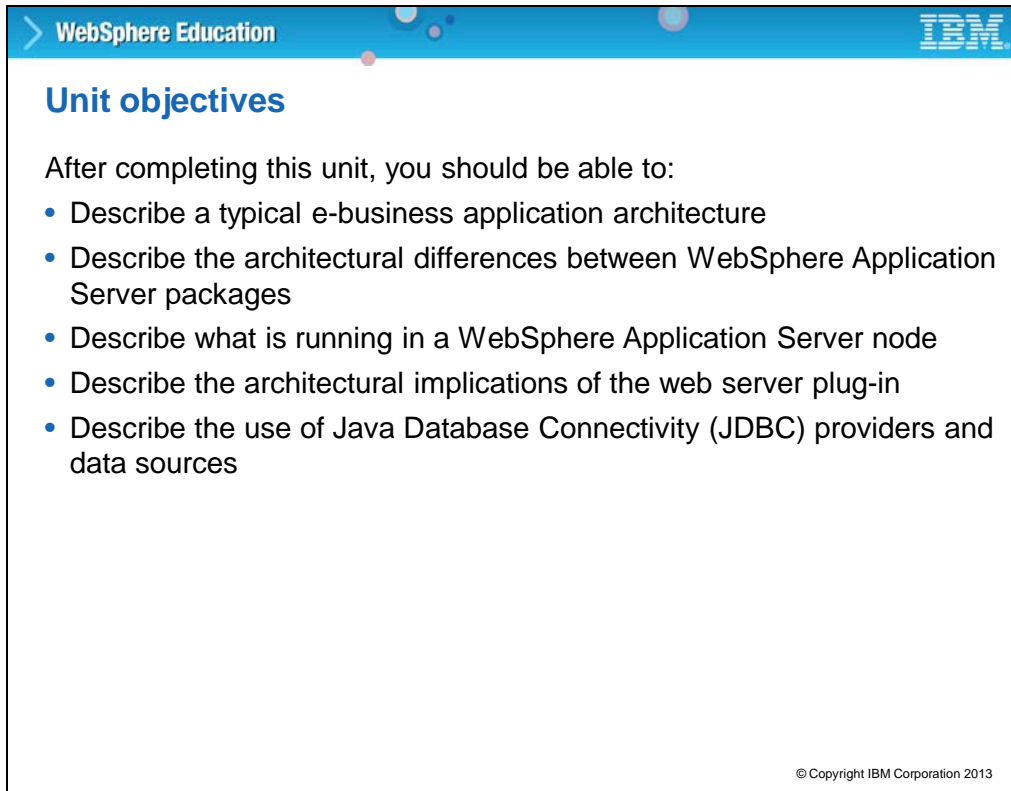



Slide 1



This unit provides an architectural overview of WebSphere Application Server V8.5 at run time.

Slide 2

The slide features a blue header bar with a white right-pointing arrow, the text 'WebSphere Education', and the IBM logo on the right. The main content area is white with a blue title 'Unit objectives'. Below the title, a paragraph states 'After completing this unit, you should be able to:', followed by a bulleted list of five objectives. The footer contains the copyright notice '© Copyright IBM Corporation 2013'.

> WebSphere Education 

Unit objectives

After completing this unit, you should be able to:

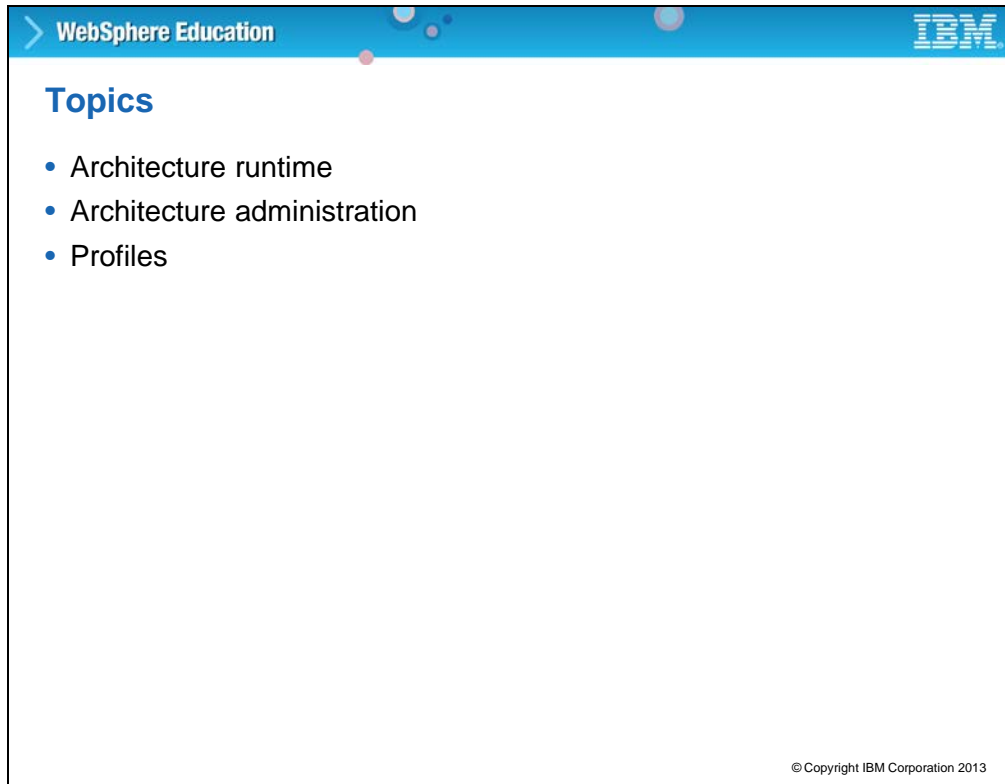
- Describe a typical e-business application architecture
- Describe the architectural differences between WebSphere Application Server packages
- Describe what is running in a WebSphere Application Server node
- Describe the architectural implications of the web server plug-in
- Describe the use of Java Database Connectivity (JDBC) providers and data sources

© Copyright IBM Corporation 2013

After completing this unit, you should be able to:

- Describe a typical e-business application architecture
- Describe the architectural differences between WebSphere Application Server packages
- Describe what is running in a WebSphere Application Server node
- Describe the architectural implications of the web server plug-in
- Describe the use of Java Database Connectivity (JDBC) providers and data sources

Slide 3

A presentation slide with a blue header bar. The header bar contains a white right-pointing arrow, the text 'WebSphere Education', and the IBM logo. The main content area is white and contains the title 'Topics' in blue, followed by a bulleted list of three items: 'Architecture runtime', 'Architecture administration', and 'Profiles'. A small copyright notice is in the bottom right corner of the slide.

> WebSphere Education IBM

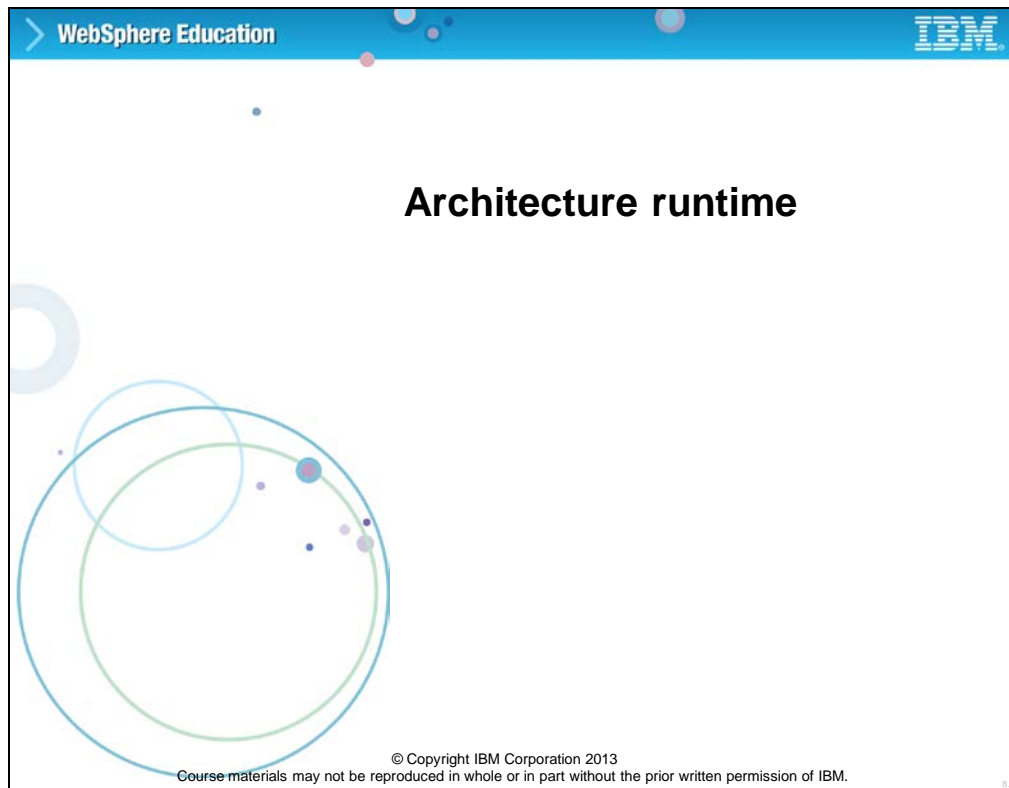
Topics

- Architecture runtime
- Architecture administration
- Profiles

© Copyright IBM Corporation 2013

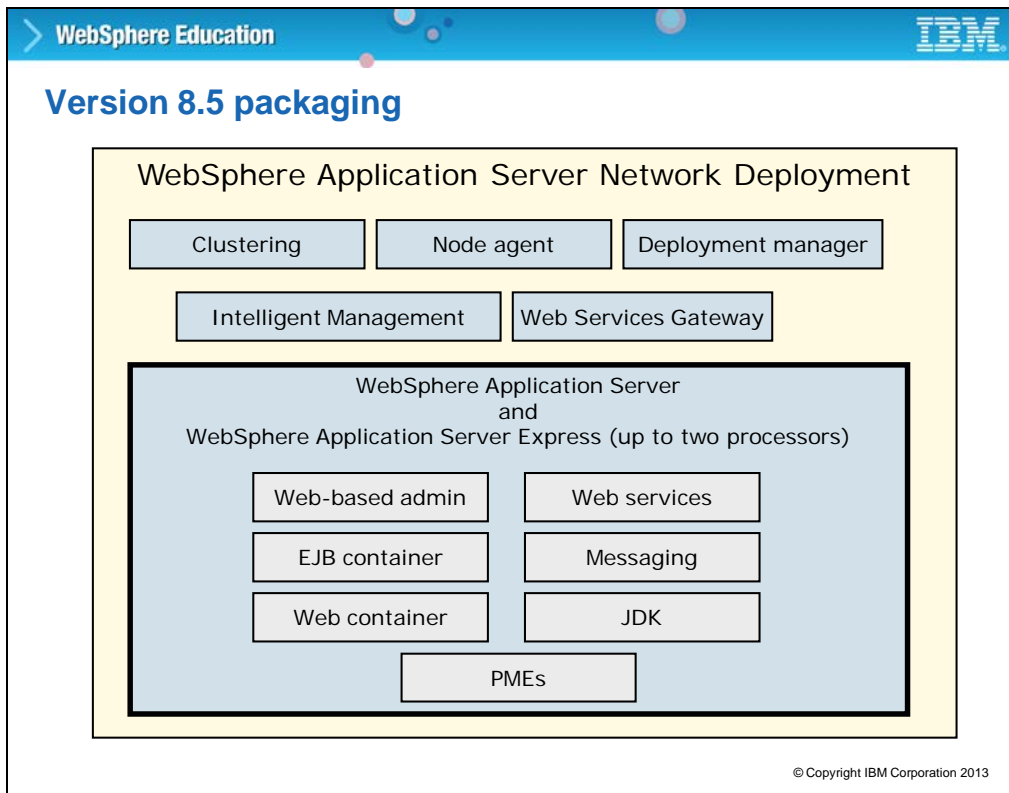
This unit is divided into three topics.

Slide 4



Topic one: Architecture runtime. In this topic, you learn about the architecture runtime of a stand-alone application server.

Slide 5



This unit focuses on the stand-alone version of WebSphere Application Server. All of the concepts that are contained within this unit are applicable to both versions: Stand-alone and Network Deployment. The next unit describes the Network Deployment edition of WebSphere.

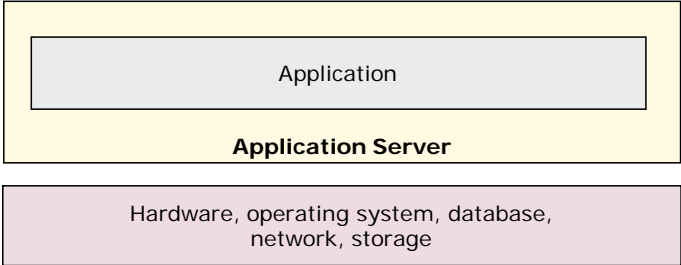
Slide 6

WebSphere Education

IBM

WebSphere Application Server basics

- WebSphere Application Server
 - Is a platform on which Java based business applications run
 - Is an implementation of the Java Platform, Enterprise Edition (Java EE) specification
 - Provides services (database connectivity, threading, workload management) that the business applications can use

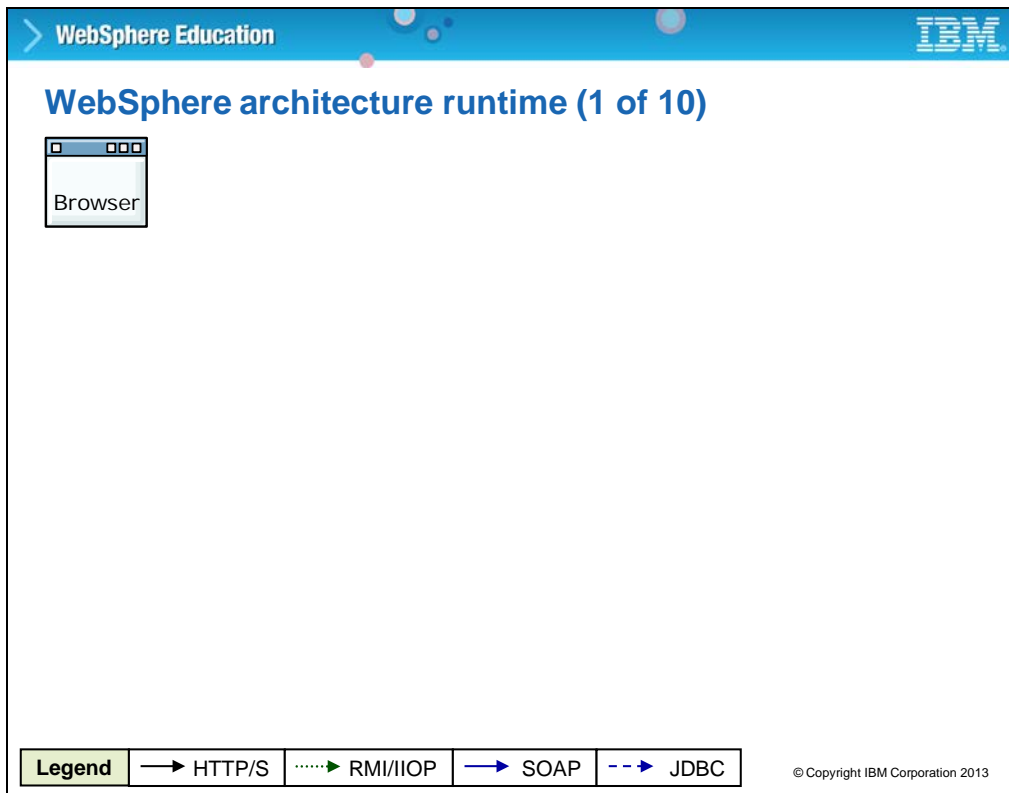


© Copyright IBM Corporation 2013

WebSphere Application Server is a platform on which Java based business applications run. It is an implementation of the Java EE specification, and provides many services that business applications use.

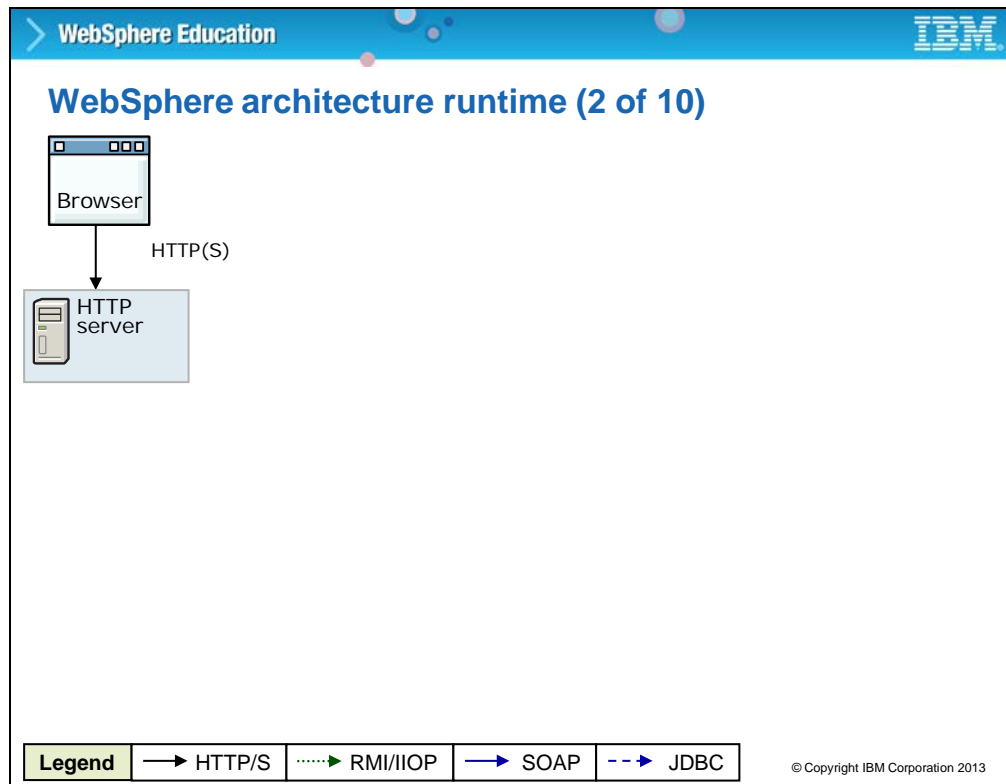
The diagram illustrates the differences between the application, the application server, and the hardware and operating system layers. Basically, the application server runs on the operating system, which provides network connectivity, storage, and other services. The applications run on the application server, which provides database connectivity, transaction processing, threading, and more.

Slide 7



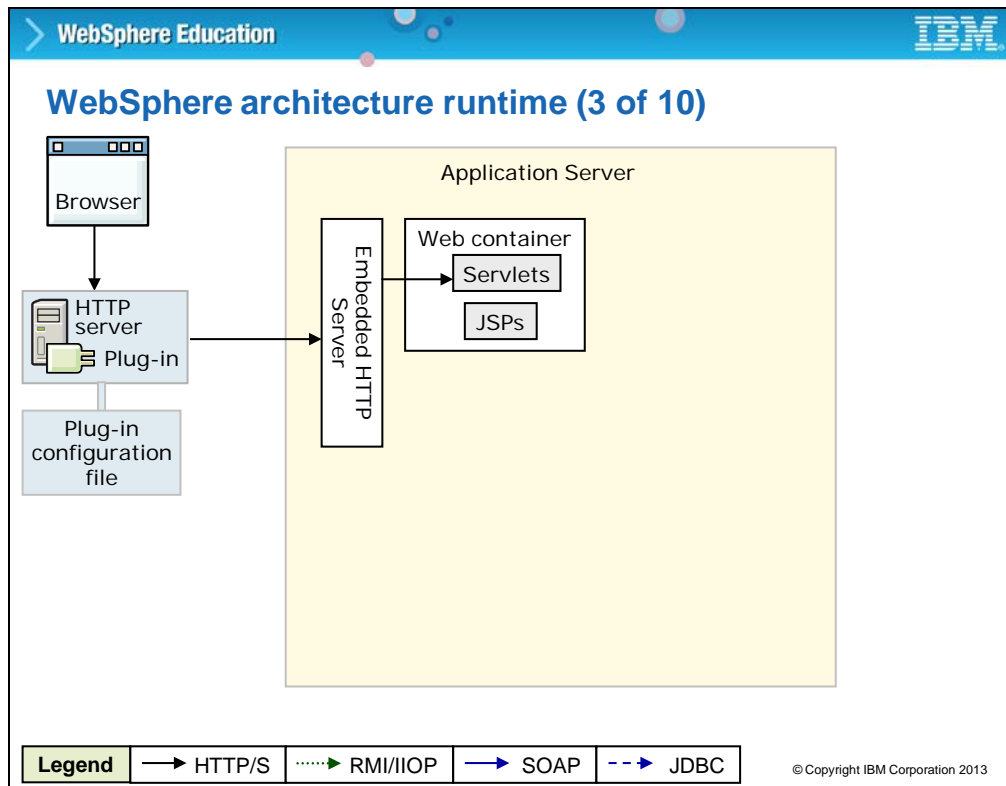
This series of slides illustrates the basic architecture of WebSphere Application Server, including several of the larger components. The first component that you see represents a web browser. Other types of application clients are also supported, but users typically access web applications through a browser. The browser is the first component in a request flow.

Slide 8



The first step for this client request is the external HTTP server. Depending on how this enterprise is set up, the external HTTP server can be inside or outside a firewall, or in a DMZ. The communication between the browser and the server can be HTTP or HTTPS, which uses security.

Slide 9



In this configuration, the external HTTP server is configured to work with the plug-in component. The plug-in component in the HTTP server determines where the client request goes from that point. In this example, there is just one option, the application server.

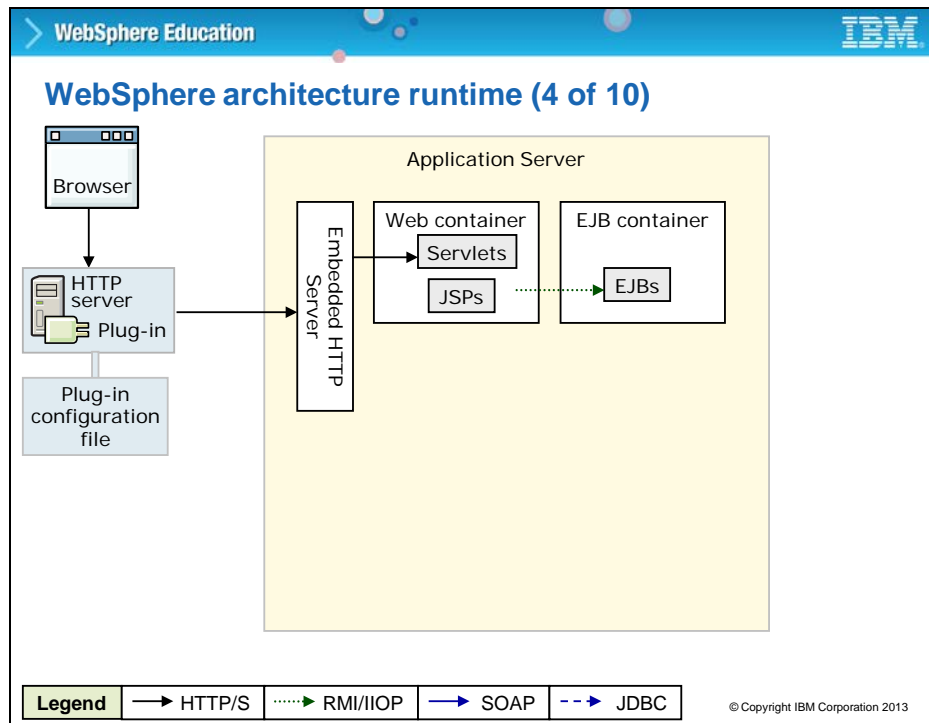
The plug-in is also a critical player in workload management of HTTP requests. In a network deployment environment, where there are multiple clustered application servers, the plug-in can be set up to distribute requests to the multiple application servers. The plug-in can also steer traffic away from unavailable servers.

Configuration information for the plug-in is stored locally in the plug-in configuration file, and the communication between the HTTP server and the application server can be either HTTP or HTTPS. The plug-in configuration file is critical as it tells the plug-in if it is to handle a request and where to send the request.

You can see the main element in this architecture is the application server, a Java process that encapsulates many services, including the containers, where business logic runs. If you are familiar with Java EE, you recognize the web container. The web container runs servlets and JavaServer Pages (JSPs), both of which are Java classes that generate markup that a browser examines.

When the request reaches the application server, an embedded HTTP server receives it, which then forwards the request appropriately within the application server.

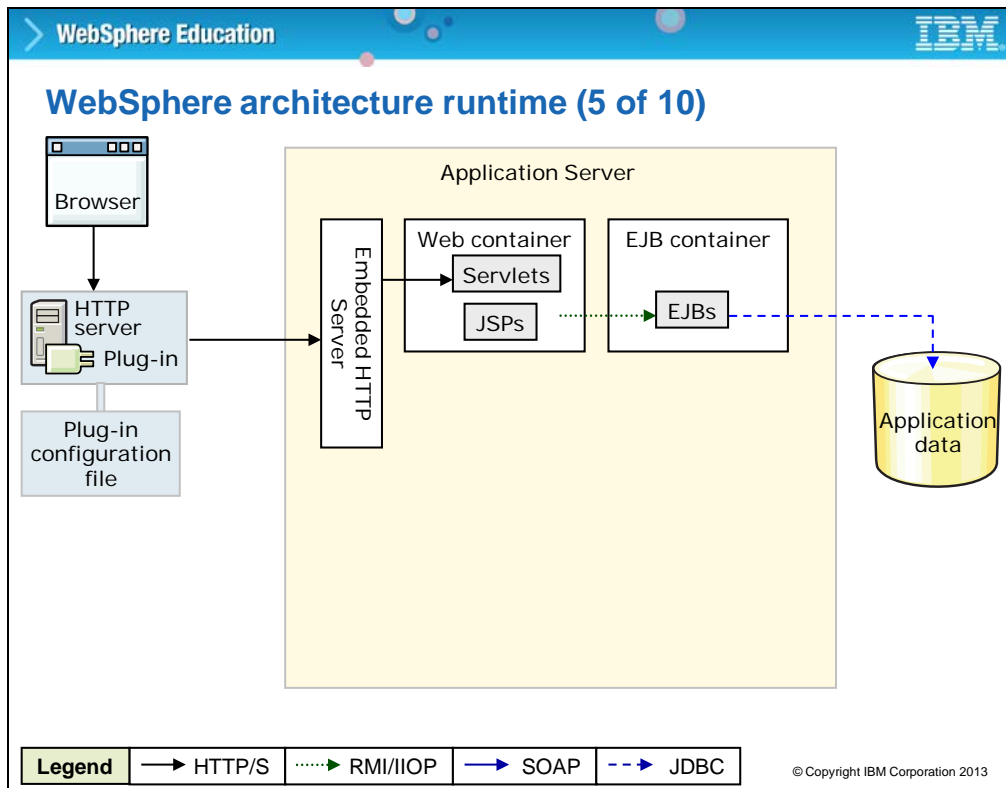
Slide 10



While servlets and JSPs can act independently, they most commonly make calls to EJBs to run business logic or access data. EJBs, which run in the EJB container, are easily reusable Java classes. They most commonly communicate with a relational database or other external source of application data, either returning that data to the web container or changing the data on behalf of the servlet or JSP.

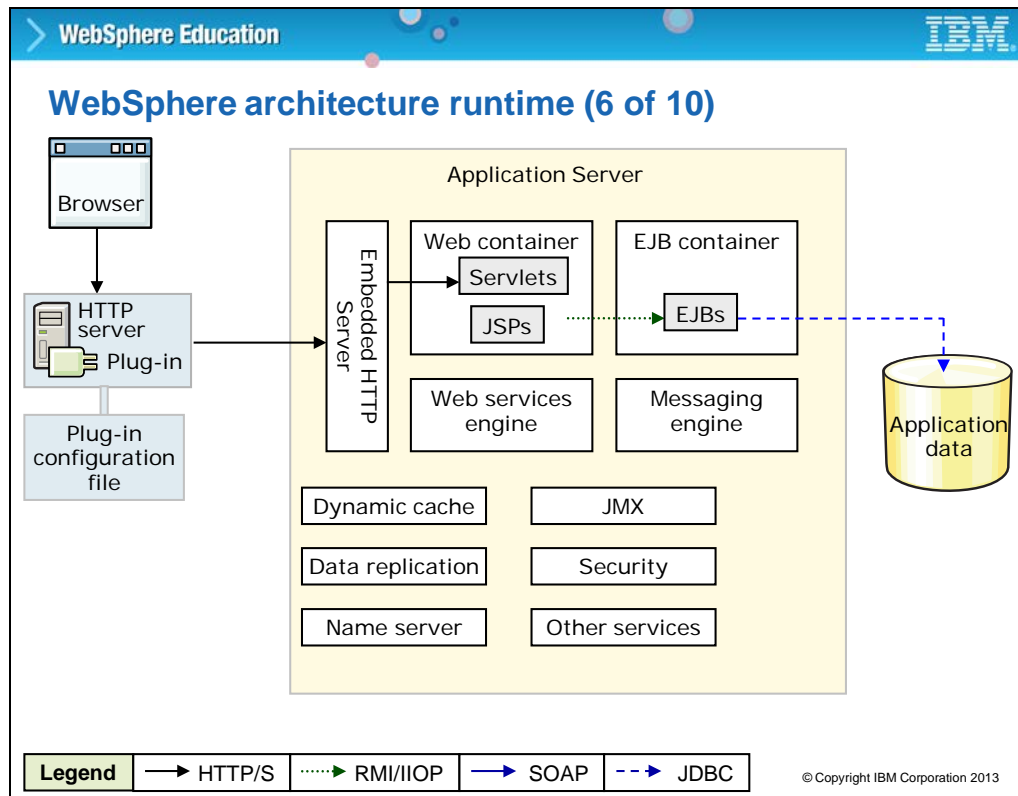
If the application uses Enterprise JavaBeans, or EJBs, the request goes from the web container to the EJB container, where the EJBs are located. Communication between the web container and the EJB container is done with RMI over IIOP.

Slide 11



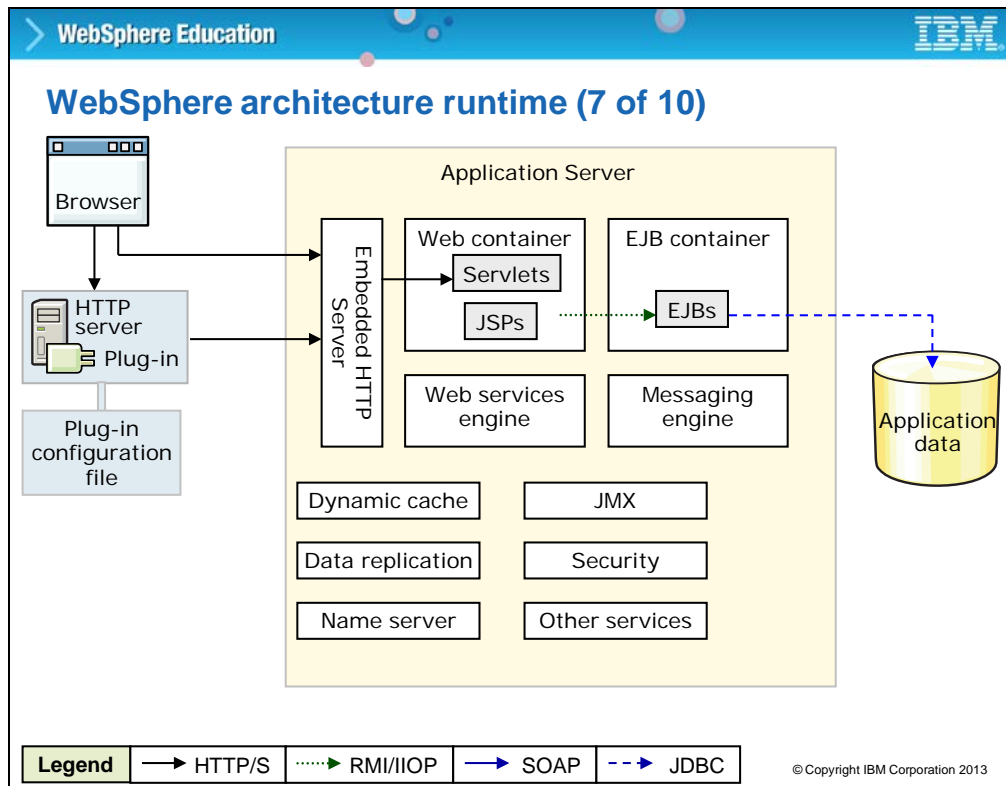
Any application data that the EJBs access is stored in a database. JDBC is used to access the database. The steps that were mentioned so far represent a typical client request flow through the WebSphere Application Server architecture, but there are many more possibilities.

Slide 12



The application server provides a number of services, some of which are shown here: naming, security, messaging, and web services. Some of these services are covered in more detail in later units.

Slide 13

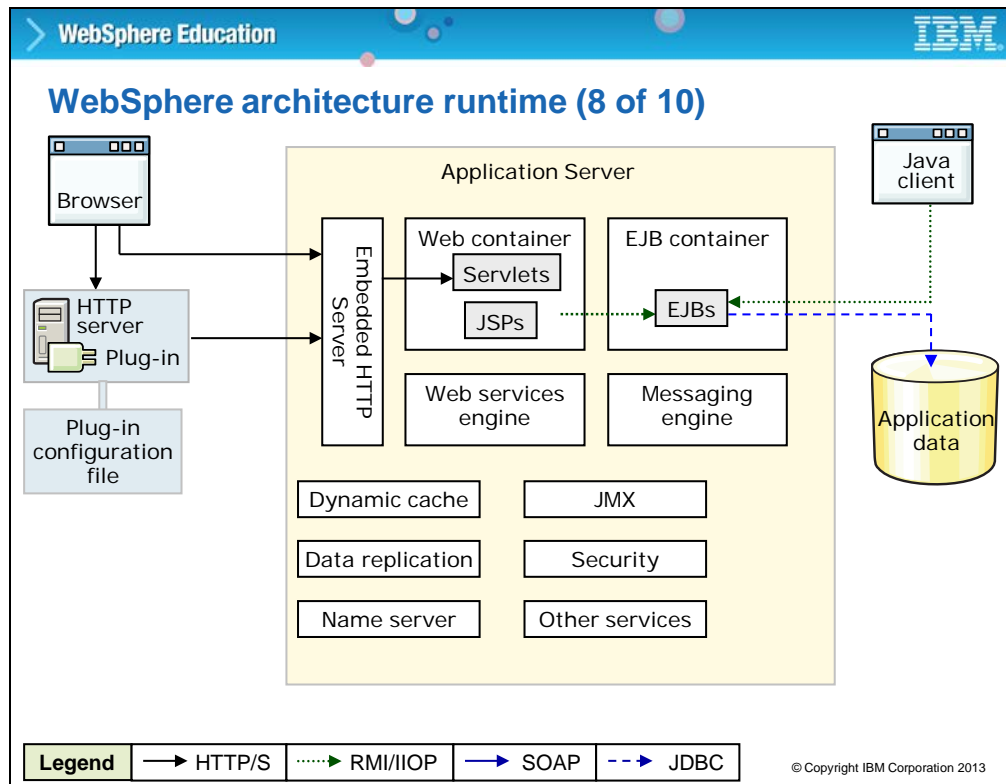


Next, you see some variations on the request flow. By using a browser, a client can access an embedded HTTP server directly, without going through the external HTTP server. It is done by modifying the URL that is used and going directly to the application server.

This method can be used for administering, testing, or troubleshooting the application server.

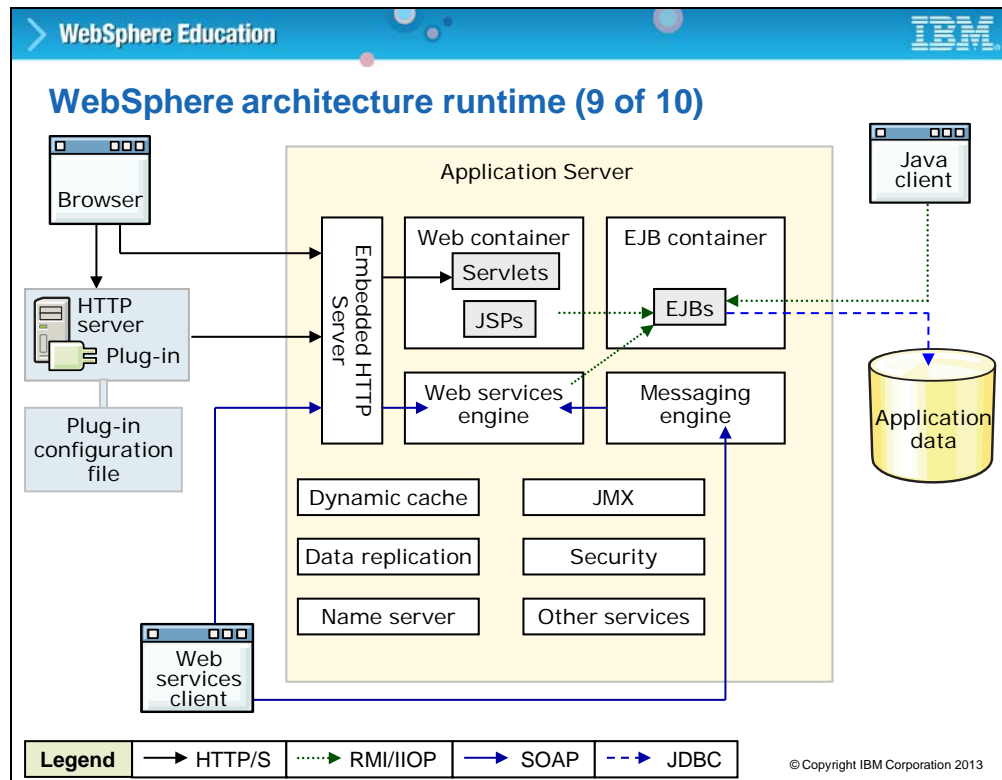
Keep in mind ~~by that~~ passing the plug-in, results in a loss of features that the plug-in provides such as security, high availability, and failover capabilities.

Slide 14



EJBs can be accessed directly by using a stand-alone Java client instead of a web browser, which is useful for testing and troubleshooting EJBs.

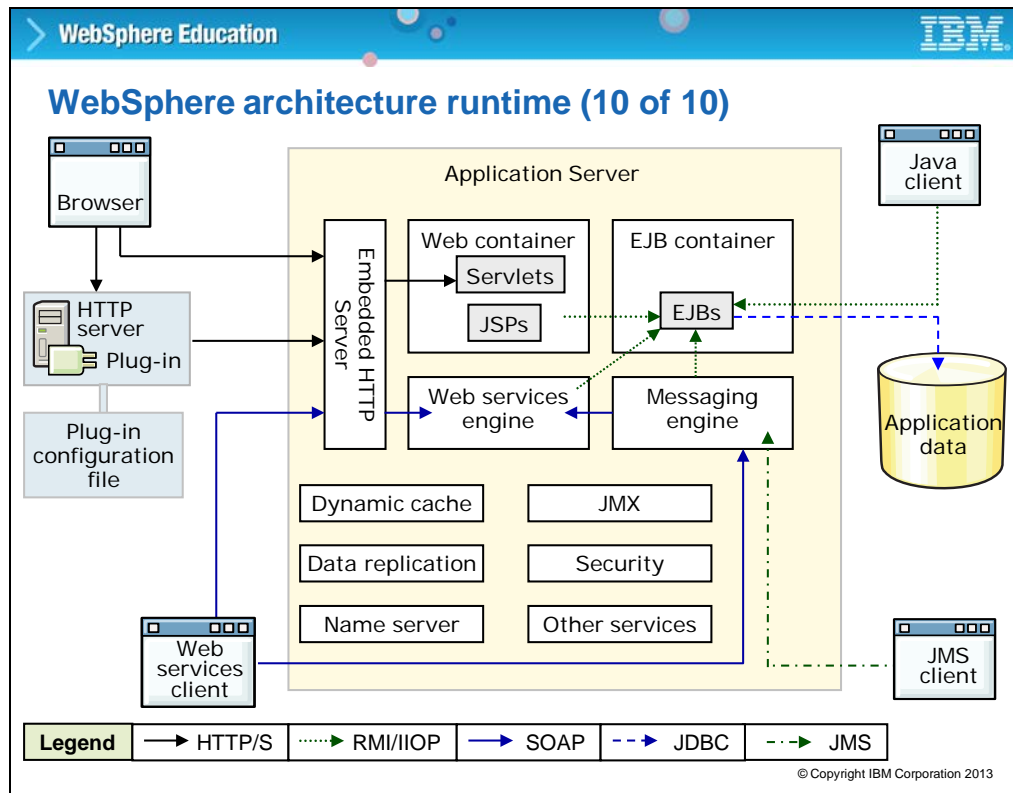
Slide 15



Some other types of applications might use web services or the messaging capabilities that are built into the WebSphere Application Server. A web service client can make requests by using SOAP over HTTP or SOAP over JMS.

The web services engine handles requests for web services, and the messaging engine handles JMS requests.

Slide 16



The messaging engine can also handle requests from a JMS client. The JMS messaging engine is built into the application server. This messaging engine is pure Java. JMS destinations, which are known as queues and topics, provide asynchronous messaging services to the code that runs inside the containers. JMS is covered in more detail later in this course.

Slide 17

> WebSphere Education

JDBC providers

- Provide the JDBC driver implementation for database access
 - Type 2 JDBC drivers (thick): require the database client software on the client node to connect to the database server
 - Type 3 JDBC drivers (net protocol): require server-side code to map net protocol to native database
 - Type 4 JDBC drivers (native protocol): connect directly to the database by using its native protocol
- XA drivers support transaction recovery

Client node

```

graph LR
    subgraph Client_node [Client node]
        subgraph Data_source [Data source]
            JNDI
            JDBC_driver[JDBC driver]
            Connection_pool((Connection pool))
        end
        Database_client[Database client]
    end
    Data_source -. JDBC .-> Database_client
    Database_client --> Database_server[Database server]
      
```

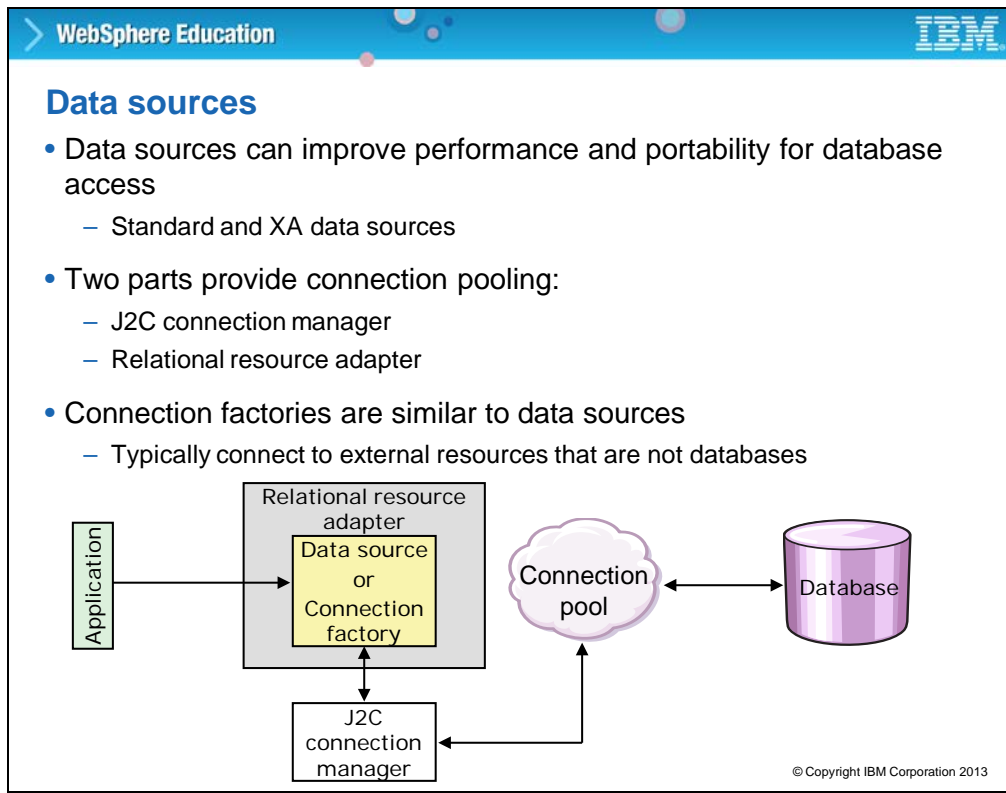
© Copyright IBM Corporation 2013

EJBs use the JDBC protocol to access application data that is stored in a database server. The JDBC driver implementation for accessing a database is specified in a JDBC provider. There are different types of JDBC drivers and the one you use depends on the requirements of your application and your database.

The Type 2 driver is considered to be a "thick" driver because it requires database client software on the client node to connect to the database server. The Type 3 driver requires server-side code to map net protocol to the local database. The Type 4 driver is considered to be a "thin" driver because it can connect directly to the database by using its own protocol. If available, Type 4 drivers are advised over using Type 2 or 3, but you use the type of driver that your application or database requires. Some JDBC drivers are also XA-compliant, which means that they support transaction recovery.

The diagram shows how this information is used to access a database. In WebSphere Application Server, you create a data source that specifies the JDBC driver. The data source also defines a unique JNDI name, which is used to identify and locate the data source. You can configure connection pool settings for the data source to optimize performance between a database client and server.

Slide 18



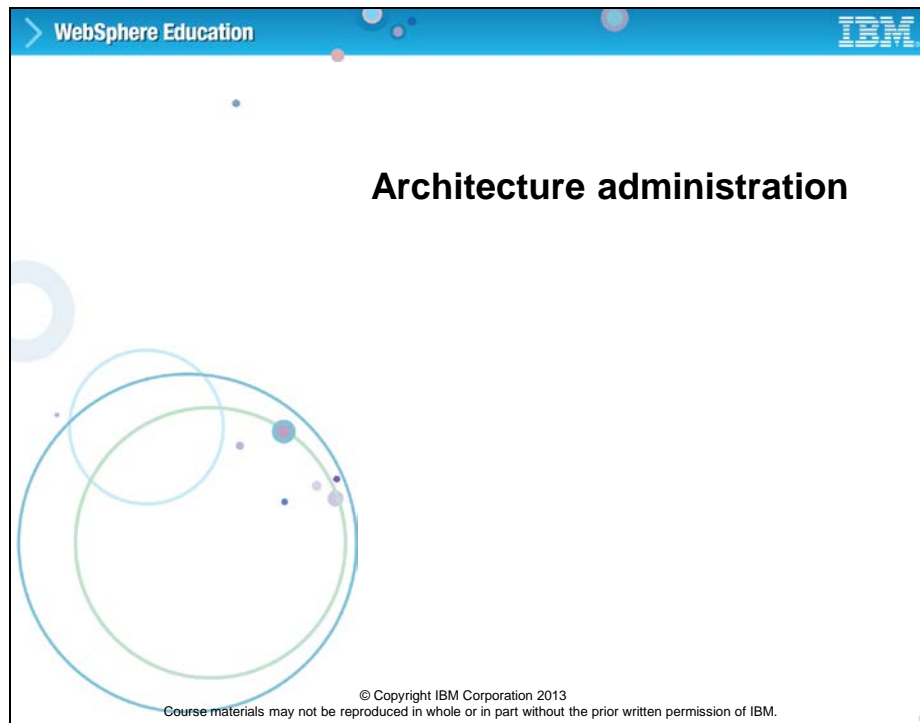
Application components use a data source to access connection instances to a relational database. A data source is associated with a JDBC provider, which supplies the driver implementation classes that are required for JDBC connectivity with your specific vendor database. There are two types of data sources: standard and XA data sources. An XA data source supports application participation in any single-phase or two-phase transaction environment.

When this data source is involved in a global transaction, the product transaction manager provides transaction recovery. The connection pool that corresponds to each data source provides connection management.

Each time an application attempts to access a data store (such as a database), it requires resources to create, maintain, and release a connection to that data store. To mitigate the strain that this process can place on overall application resources, the application server enables administrators to establish a pool of connections that applications can share on an application server.

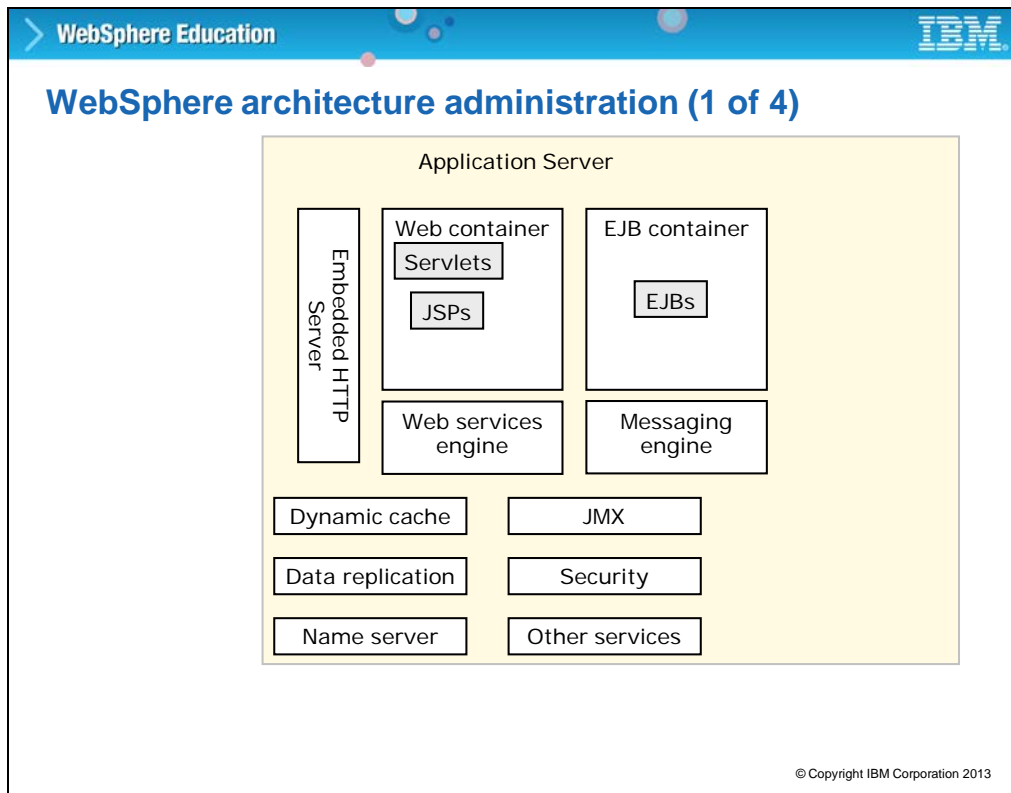
Connection pooling spreads the connection usages across several user requests, conserving application resources for future requests. The Java EE Connector, or J2C connection manager enables allocation of connection handles. The resource adapter plugs into an application server and provides connectivity between the enterprise information system, the application server, and the enterprise application.

Slide 19



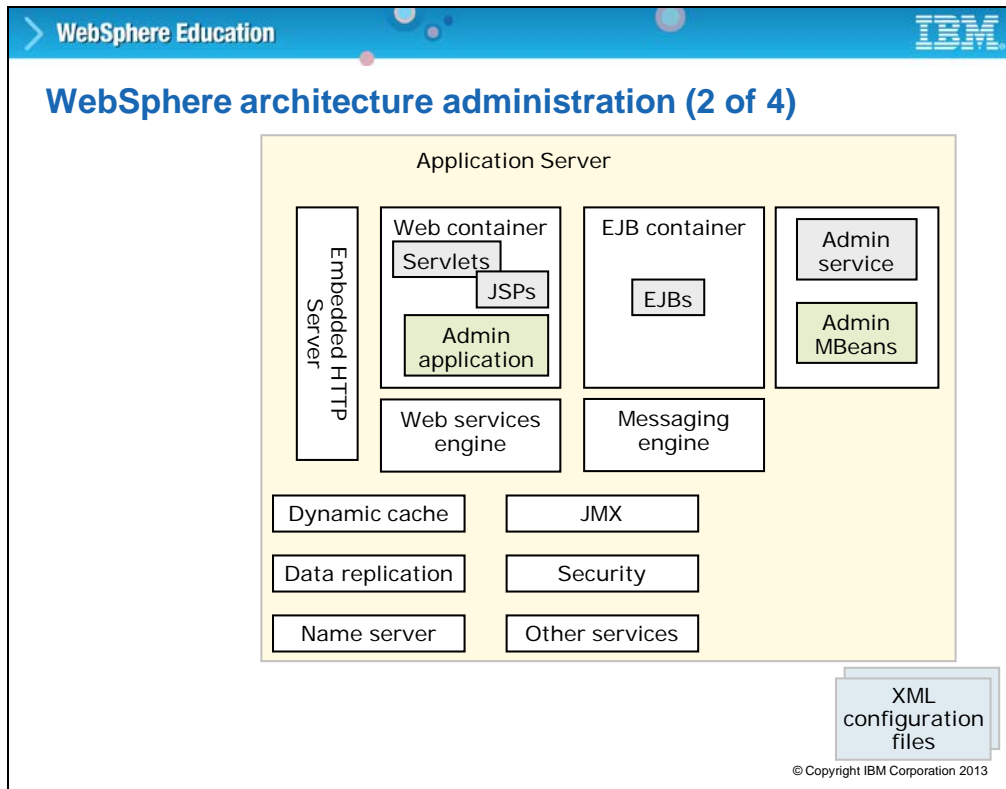
Topic two: Architecture administration. In this topic, you learn about administration concepts for an application server.

Slide 20



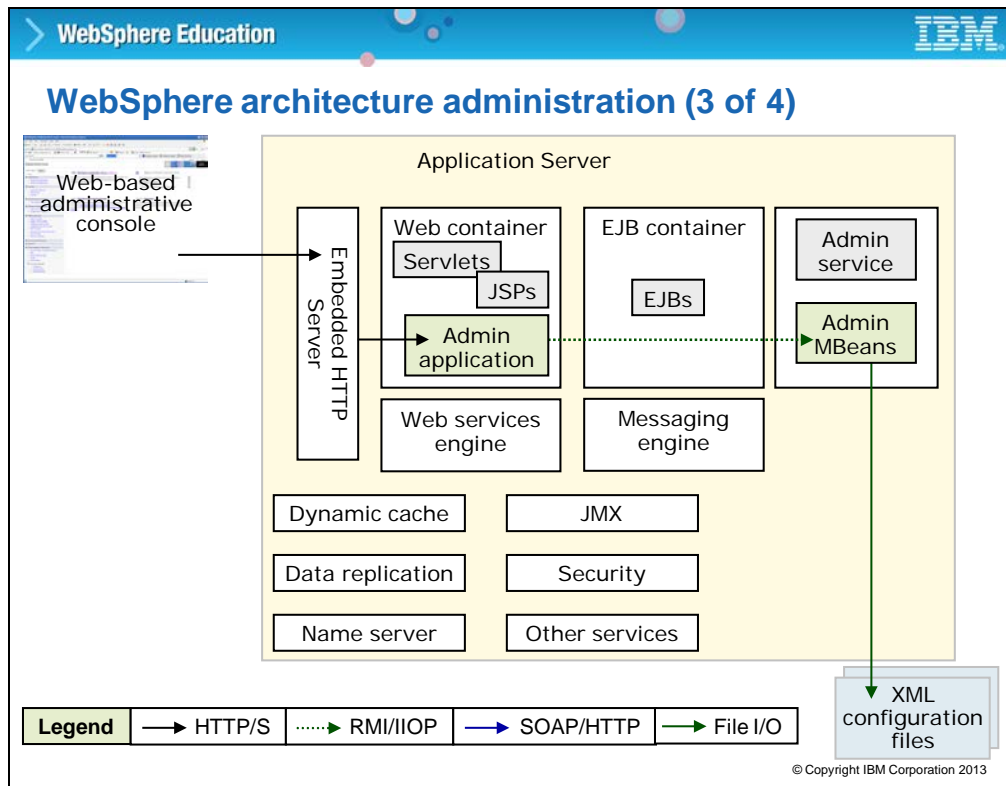
Next, examine some of the components of the WebSphere administrative architecture. Earlier, you saw the runtime depiction of a WebSphere Application Server. This diagram illustrates the basic architecture of administering WebSphere Application Server. These tools are the two main administrative tools that are used to administer WebSphere Application Server: the administrative console and the wsadmin command-line tool.

Slide 21



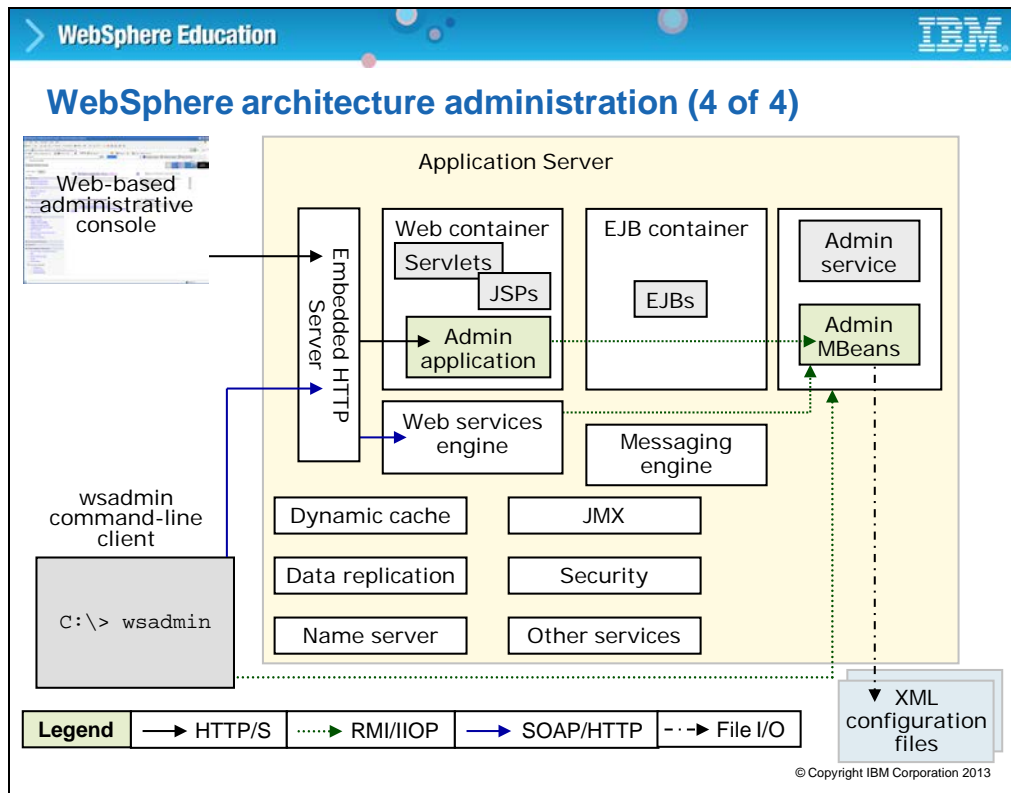
The configuration information of the server is stored in a set of XML files that are known as the configuration repository. These files define the server itself, and resources and services that it provides. This configuration information for an application server is stored in the local file system. One of the services available within the application server is the administrative service, or admin service. This service allows for configuration of the application server. The admin service manages changes to the configuration repository through a set of interfaces called Mbeans. An application that runs within the web container provides users the ability to administer the application server through a web application. The application is listed in the diagram as the admin application, which is the administrative console. The admin application works with the admin service to update the configuration files in the repository.

Slide 22



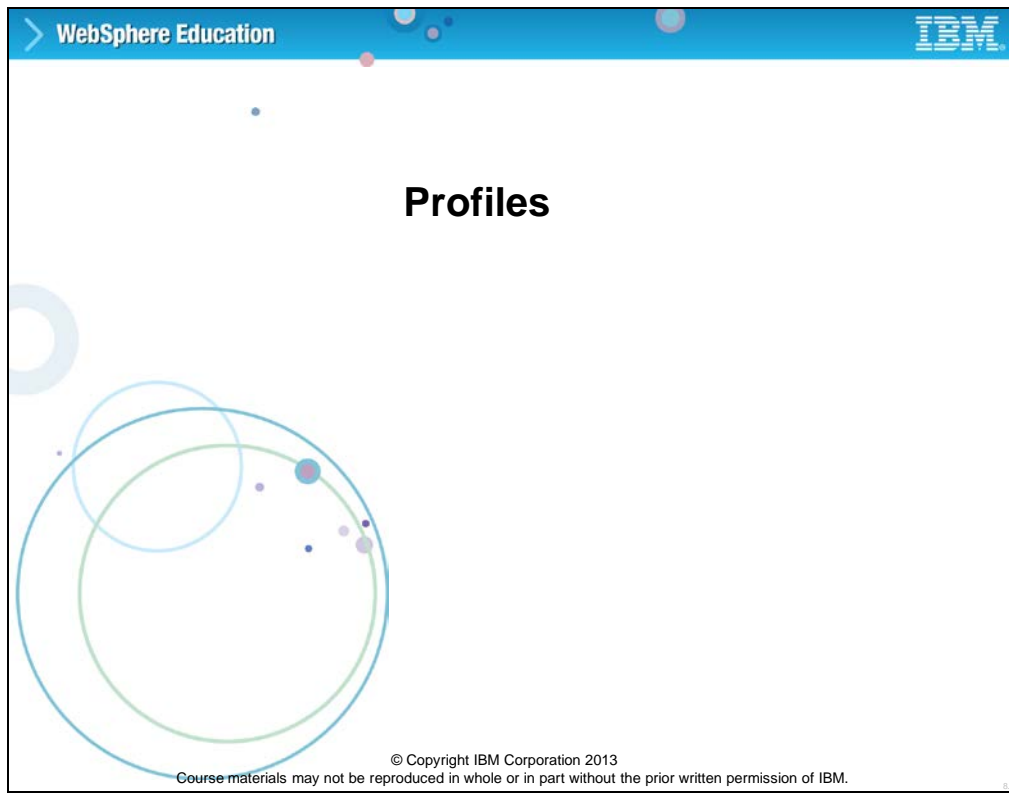
Changes to the configuration can be made through the web-based administrative console, which accesses the application server through its embedded HTTP server. The server in turn forwards requests through the admin application and to the admin service, which ultimately updates the configuration files. The administrative console allows an administrator to use a graphical tool to update the XML configuration repository instead of manually updating the repository. Manually editing of the XML configuration repository is not advised.

Slide 23



Changes can also be made to the configuration through the wsadmin command-line interface. Basically, anything that can be done through the web-based administrative console, can also be done by using wsadmin commands. With wsadmin, you **are** also capable of automating multiple tasks with scripts. The wsadmin interface gives you the option of accessing the application server through the embedded HTTP server, or accessing the Admin service directly by using RMI over IIOP or SOAP. SOAP is the default protocol.

Slide 24



Topic three: Profiles. In this topic, you learn about the profiles.

Slide 25

The slide is titled 'WebSphere profile overview' and is part of the 'WebSphere Education' series, as indicated by the header. It explains that profiles are sets of files representing a WebSphere Application Server configuration. It then categorizes the files into two groups: Product files and Profiles (configuration files). To the right of the text, there are two overlapping boxes: a green one labeled 'WebSphere product files (binary files)' and a yellow one labeled 'Profile files' which lists 'DmgrProfile', 'profile1', and 'profile2'.

WebSphere Education

WebSphere profile overview

Profiles are sets of files that represent a WebSphere Application Server configuration

WebSphere Application Server files are split into two categories:

- **Product files**
 - Set of shared read-only static files or product binary files
 - Shared among any instances of the WebSphere Application Server product
- **Profiles (configuration files)**
 - Set of user-customizable data files
 - Files include WebSphere configuration, installed applications, resource adapters, properties, and log files

WebSphere product files (binary files)

Profile files

- DmgrProfile
- profile1
- profile2

© Copyright IBM Corporation 2013

Two categories of files make up WebSphere Application Server: product files and configuration files. Product files include the application binary files that are required to run the application server. The files are read-only files which any instances of the product share. Configuration files include the configuration information that you define for your runtime environment, such as installed applications, data sources, properties files, and other definitions. Configuration files are also known as profiles. Profiles are the configuration mechanism that allows you to run more than one application server on a single installation of WebSphere product files.

Slide 26

WebSphere Education **IBM**

WebSphere profile benefits

- Benefits of profiles:
 - Each profile uses the same product files
 - Simpler than multiple WebSphere installations
 - Less disk space
 - Simplifies application of product updates

The diagram illustrates the WebSphere profile architecture. It shows two profiles, profile1 and profile2, each with its own set of configuration files (bin, config, configuration, consolepreferences, etc., firststeps, installableApps, installedConnectors, installedFilters, logs, properties, staticContent, temp, tranlog). A separate tree view shows the WebSphere directory structure, highlighting the 'profiles' directory which contains 'DmgrProfile', 'profile1', and 'profile2'.

© Copyright IBM Corporation 2013

Each profile uses the same set of binary files, rather than requiring a new installation for each runtime configuration, which is a benefit of separating the configuration information from the product files.

After installing the core product files for the product, you must create a profile. Each profile uses the same product files, regardless of type. The configuration files are a set of user-customizable data files. Collectively, these files make up a profile in a profile directory. The `<was_root>/profiles` directory is the default directory for creating profiles.

In the example that is shown, two application servers are each configured according to the files that exist within their own profile directory.

Slide 27

WebSphere Education **IBM**

Managing profiles

2 Command Prompt

```
C:\Program Files\IBM\WebSphere\AppServer\profiles\DmgrProfile\bin>manageprofiles.bat -help
```

Function:
Creates, lists, alters or deletes profiles

Syntax:
manageprofiles -<mode> -<argument> <argument parameter> ...

The available modes are:

```
create
augment
delete
unaugment
unaugmentAll
deleteAll
listProfiles
listAugments
backupProfile
restoreProfile
getName
getPath
validateRegistry
validateAndUpdate
getDefaultName
setDefaultName
response
help
```

Note: Command-line arguments are case-sensitive.
Note: If argument accepts a parameter, it must be specified.
Note: The default profile template is 'profile1'.
Note: Mode specific arguments are listed in the help text.
To see the detailed help text, type 'manageprofiles.bat -help'.

1 WebSphere Customization Toolbox 8.5

File Window Help

Profile Management Tool Welcome

Profiles

| Profile name | Environment | Profile path |
|--------------|--------------------|---------------------------------------|
| profile1 | Application server | /opt/IBM/WebSphere/AppServer/profiles |

Create... Augment...

Profiles are managed through one of the tools provided:

- 1. Profile Management Tool**
 - Accessed through the WebSphere Customization Toolbox
 - Gathers user input and starts the **manageprofiles** command-line tool to create the profiles
- 2. manageprofiles script**
 - Command-line interface for profile management functions

© Copyright IBM Corporation 2013

An initial profile can be created at the end of the WebSphere Application Server installation, or profiles can be created any time after by using one of the two tools provided: the profile management tool or the manageprofiles script.

The profile management tool is an eclipse-based GUI that walks you step-by-step through the profile creation process. The profile management tool is part of the WebSphere Customization Toolbox.

The manageprofiles script is a command-line tool that allows you to specify the parameters for creating a profile and can also be used to manage or delete a profile.

Slide 28

WebSphere Education
IBM

Profile types

- Cell
 - Deployment manager with a federated application server
- Management
 - Administrative agent
 - Deployment manager
 - Job manager
- Application server
 - Stand-alone
- Custom profile
 - Federated node
(no application server)
- Secure proxy

Environment Selection

Select a specific type of environment to create.

Environments:

WebSphere Application Server

Cell (deployment manager and a federated application server)

Management

Application server

Custom profile

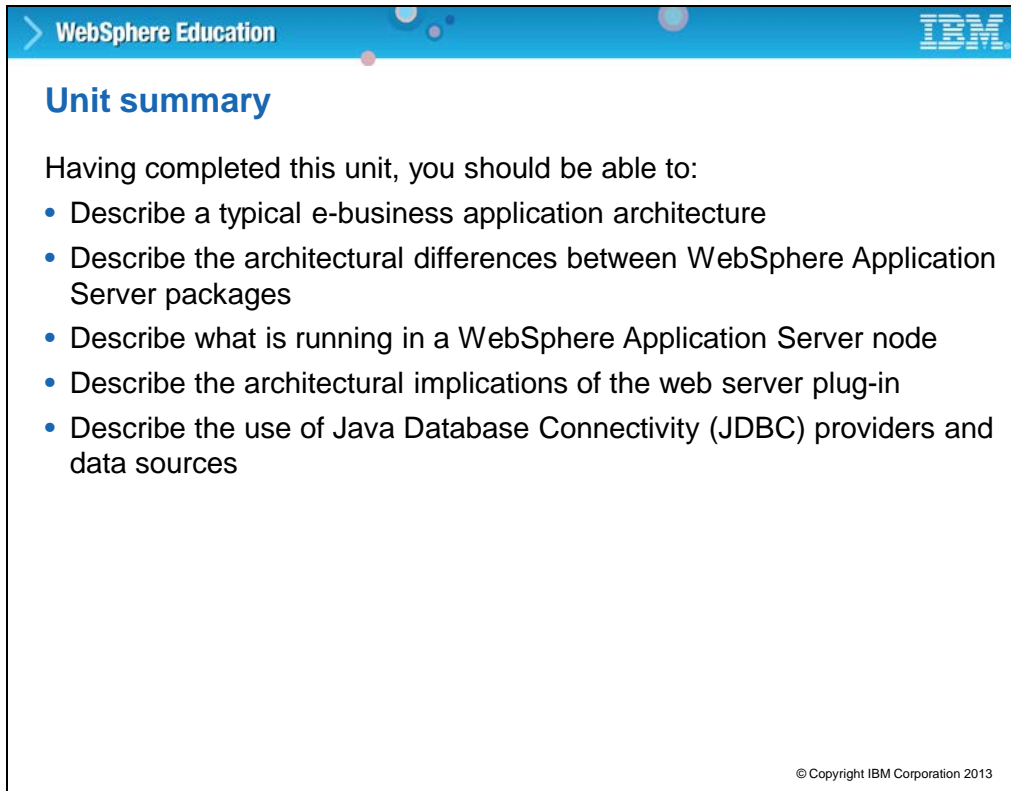
Secure proxy (configuration-only)


© Copyright IBM Corporation 2013

You can create several types of profiles. Initially, for a stand-alone application server, you would choose the application server profile. The other types of profiles that are listed are covered in detail later in the course. In short, if you are building a network deployment cell, you would create a deployment manager or a cell profile. The job manager and administrative agent profiles are for a flexible management environment.

A custom profile can be used to create a run time with a node agent (federated into a cell), but without an application server. There is also a profile type for creating a secure proxy.

Slide 29

The slide features a blue header bar with a white right-pointing arrow, the text 'WebSphere Education', and the IBM logo on the right. The main content area is white with a blue title 'Unit summary'. Below the title, it states 'Having completed this unit, you should be able to:' followed by a bulleted list of five items. The footer contains the copyright notice '© Copyright IBM Corporation 2013'.

> WebSphere Education 

Unit summary

Having completed this unit, you should be able to:

- Describe a typical e-business application architecture
- Describe the architectural differences between WebSphere Application Server packages
- Describe what is running in a WebSphere Application Server node
- Describe the architectural implications of the web server plug-in
- Describe the use of Java Database Connectivity (JDBC) providers and data sources

© Copyright IBM Corporation 2013

You completed this unit.

Having completed this unit, you should be able to:

- Describe a typical e-business application architecture
- Describe the architectural differences between WebSphere Application Server packages
- Describe what is running in a WebSphere Application Server node
- Describe the architectural implications of the web server plug-in
- Describe the use of Java Database Connectivity (JDBC) providers and data sources