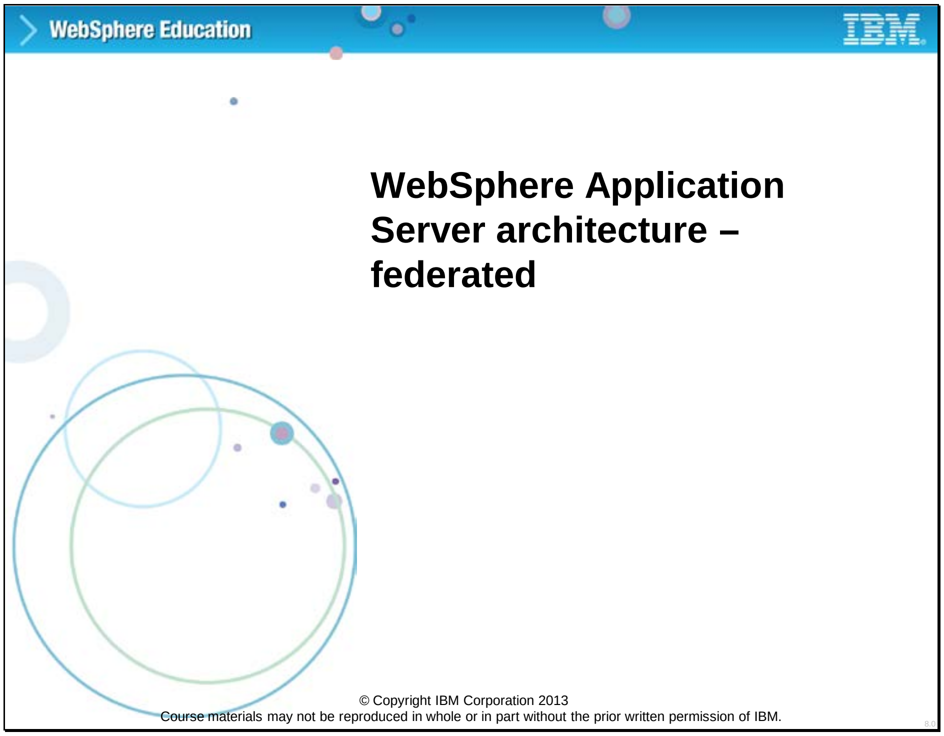
Slide 1



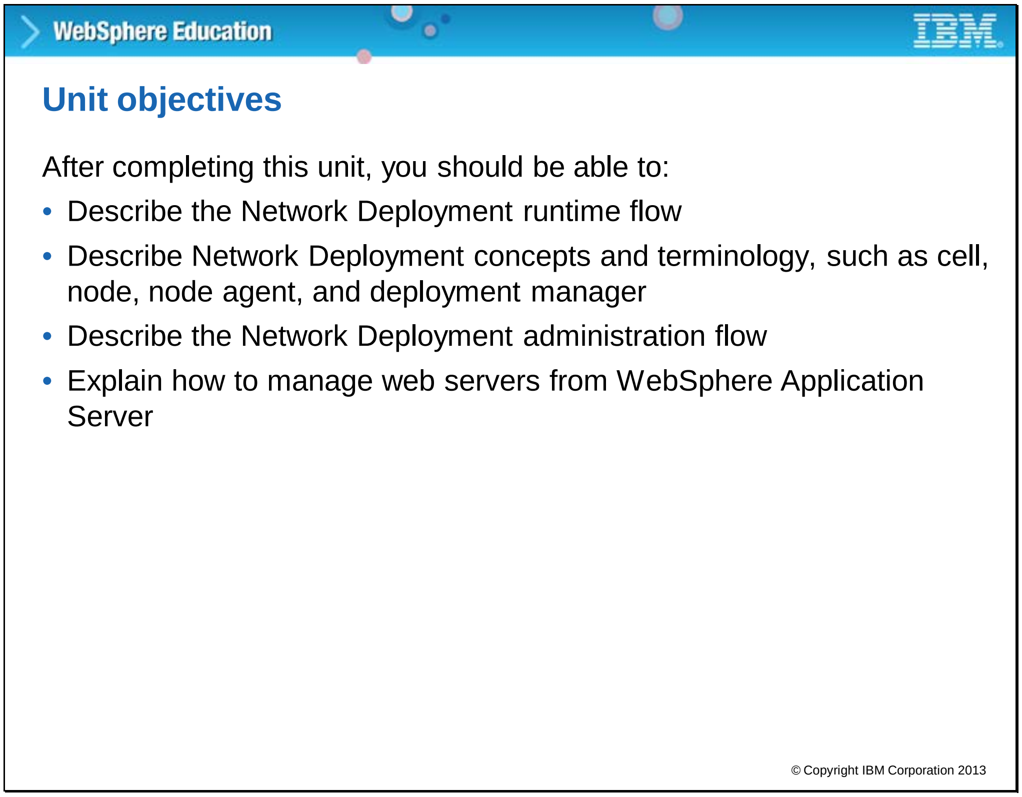
WebSphere Education

WebSphere Application Server architecture – federated

WebSphere Application Server architecture – federated

This unit describes the concepts and terminology of a Network Deployment configuration.

Slide 2



WebSphere Education

Unit objectives

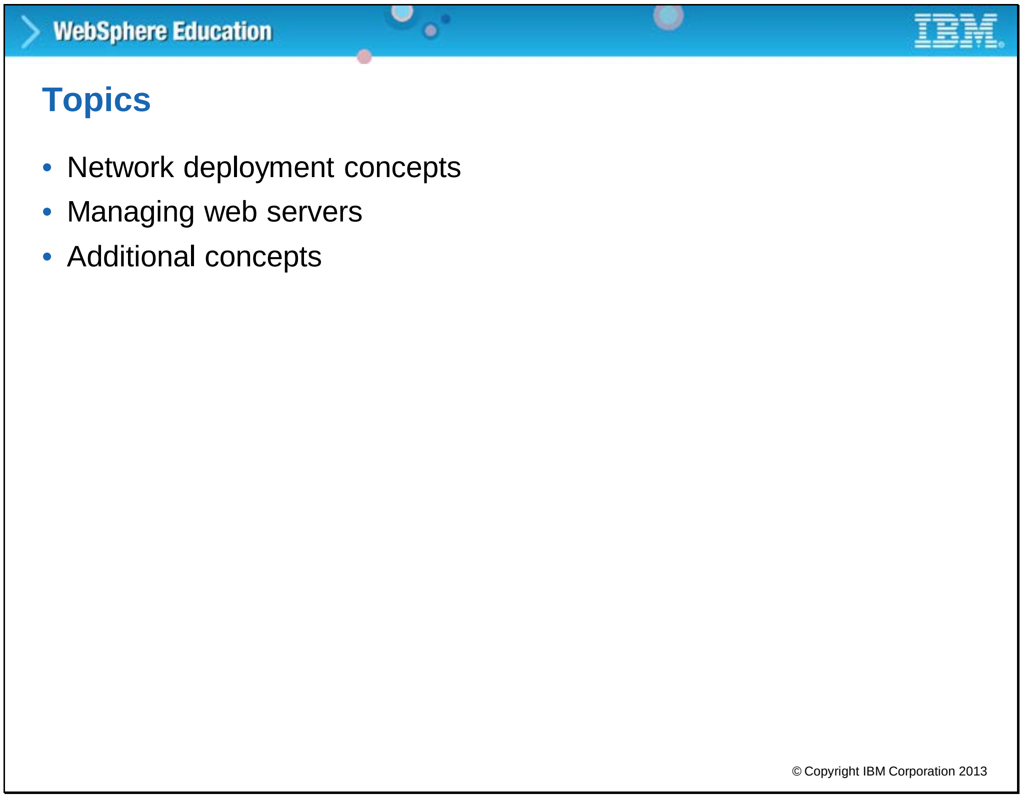
After completing this unit, you should be able to:

- Describe the Network Deployment runtime flow
- Describe Network Deployment concepts and terminology, such as cell, node, node agent, and deployment manager
- Describe the Network Deployment administration flow
- Explain how to manage web servers from WebSphere Application Server

After completing this unit, you should be able to:

- Describe the network deployment runtime flow
- Describe network deployment concepts and terminology such as cell, node, node agent, and deployment manager
- Describe the network deployment administration flow
- Explain how to manage web servers from WebSphere Application Server

Slide 3



WebSphere Education

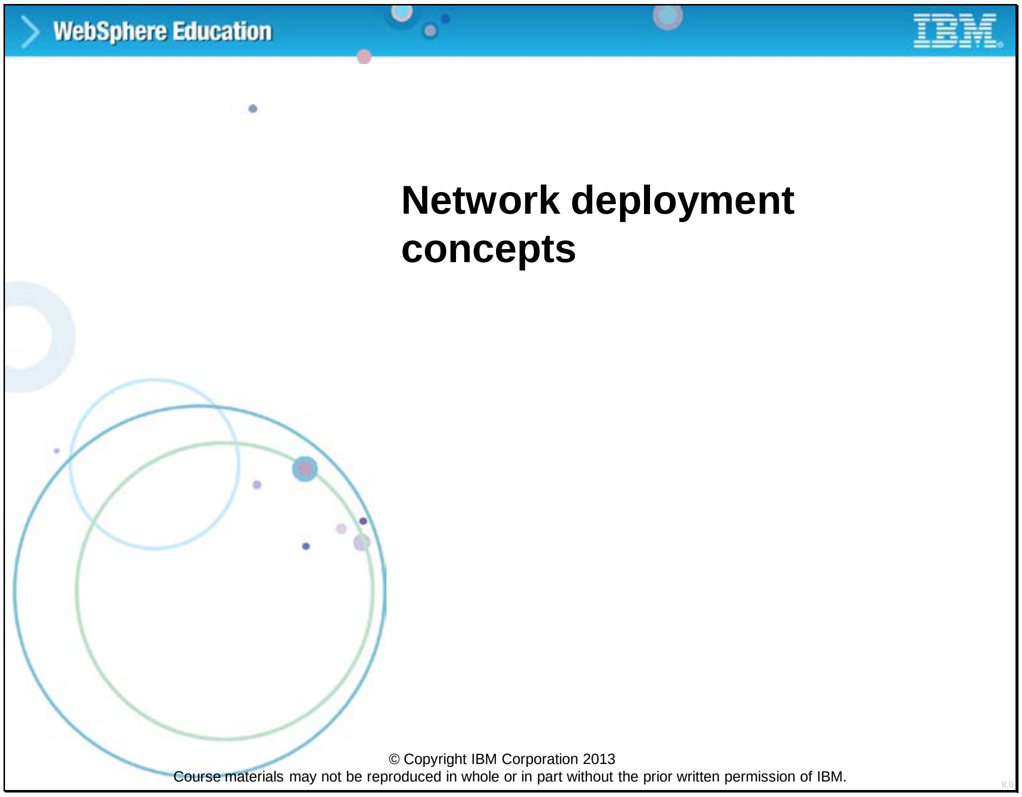
IBM

Topics

- Network deployment concepts
- Managing web servers
- Additional concepts

This unit is divided into three topics.

Slide 4



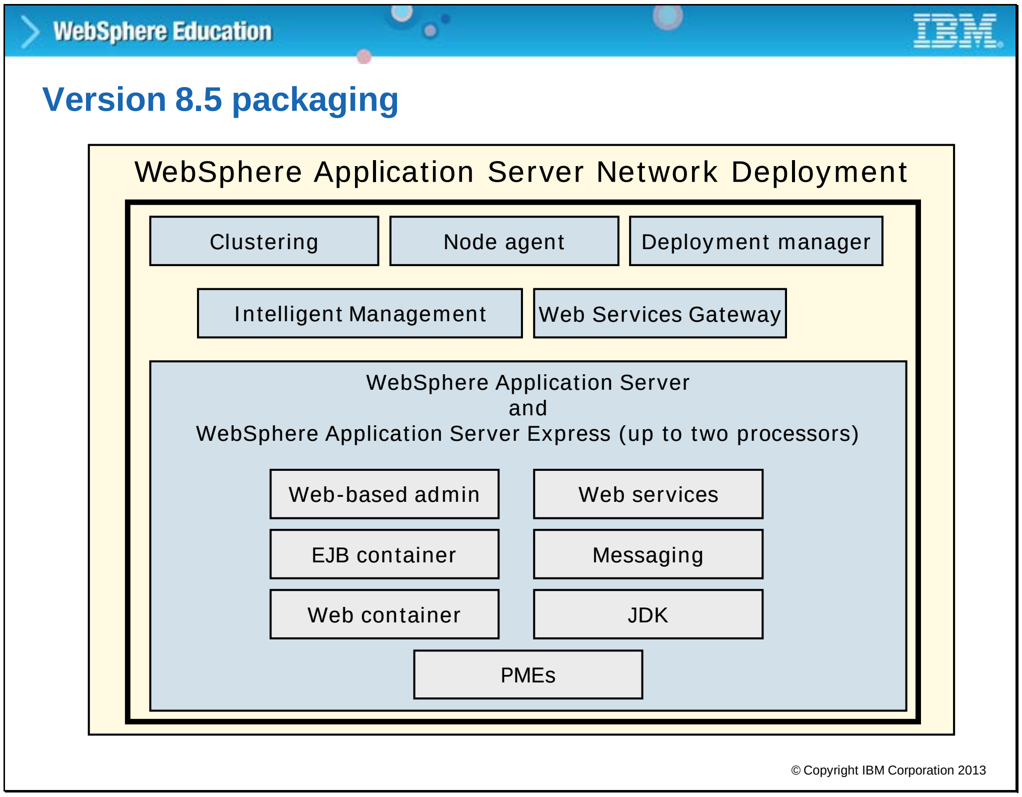
WebSphere Education

Network deployment concepts

8.0

Topic: Network deployment concepts. In this topic, you learn about various network deployment concepts.

Slide 5



WebSphere Education

Version 8.5 packaging

WebSphere Application Server Network Deployment

Clustering

Node agent

Deployment manager

Intelligent Management

Web Services Gateway

WebSphere Application Server
and
WebSphere Application Server Express (up to two processors)

Web-based admin

Web services

EJB container

Messaging

Web container

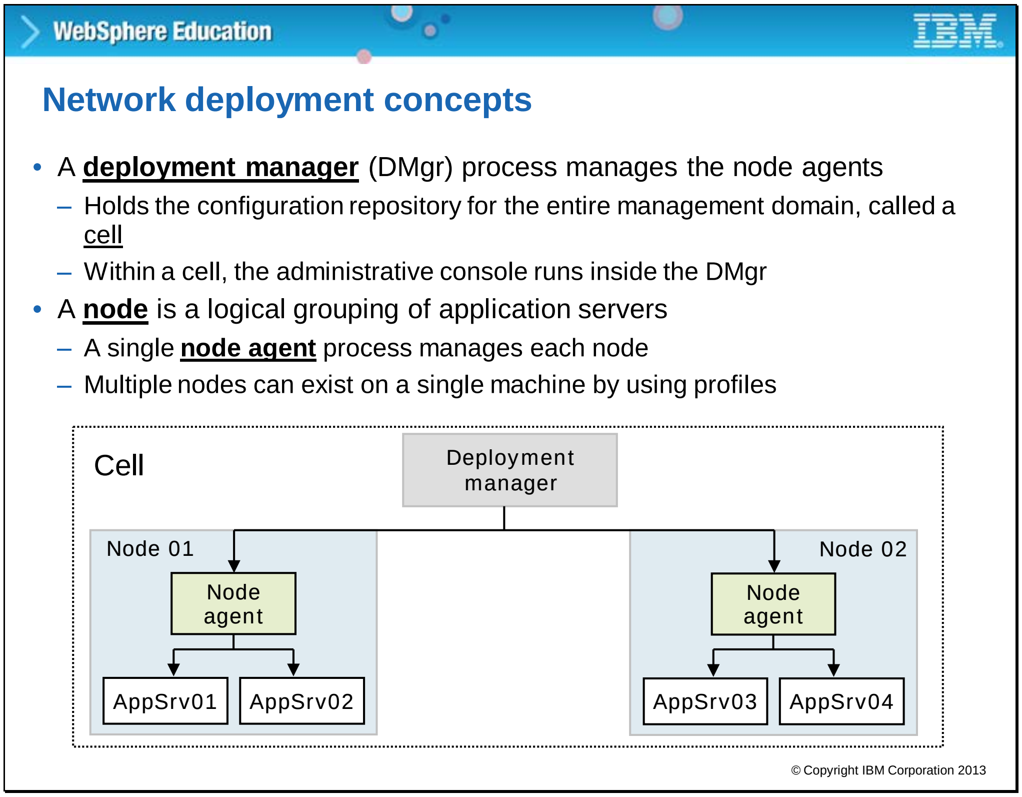
JDK

PMEs

This unit builds on what is presented in the previous unit and extends the WebSphere Application Server architecture into a network deployment environment.

With network deployment, you can administer multiple application servers from a central point, and also apply workload management across a cell, or administrative domain.

Slide 6



WebSphere Education
IBM

Network deployment concepts

- A **deployment manager** (DMgr) process manages the node agents
  - Holds the configuration repository for the entire management domain, called a cell
  - Within a cell, the administrative console runs inside the DMgr
- A **node** is a logical grouping of application servers
  - A single **node agent** process manages each node
  - Multiple nodes can exist on a single machine by using profiles

A couple of special processes create and help manage a network deployment cell: the deployment manager and the node agents.

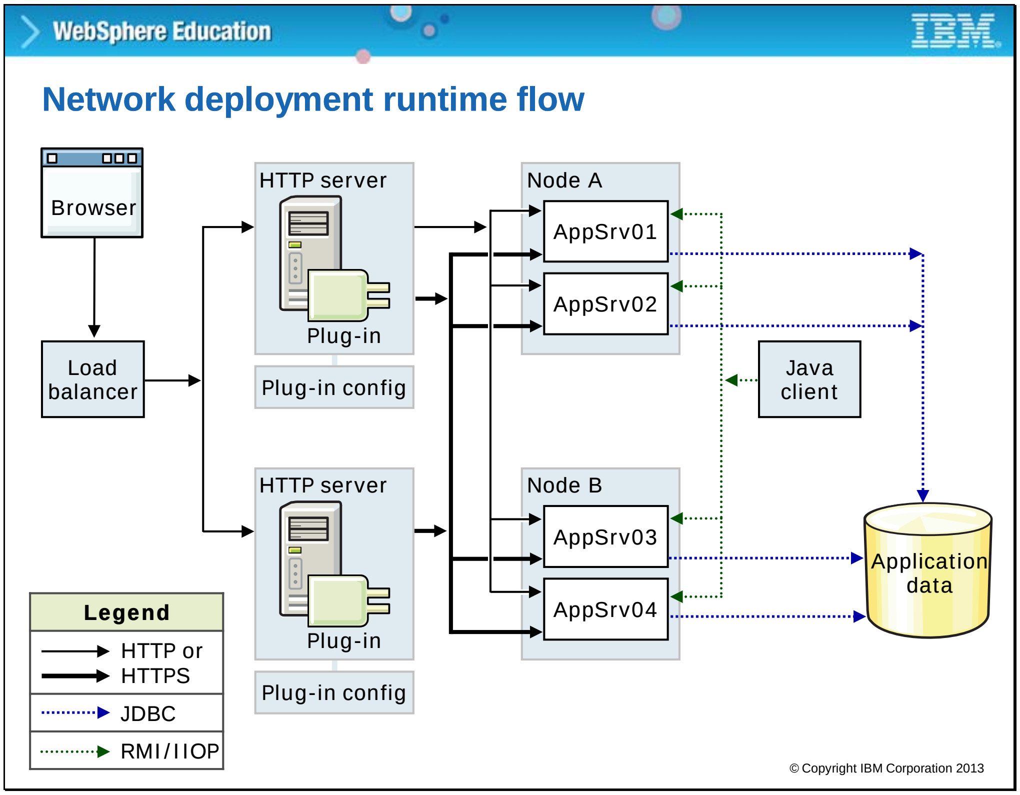
The deployment manager is a special type of server that manages all the nodes in an administrative domain, called a cell. There is only one deployment manager per cell. The deployment manager holds the configuration repository for the entire management domain, called a cell.

A node is a logical grouping of one or more application server processes that share a common configuration repository. You can have one or more nodes that are configured on a system, but nodes do not span multiple systems. You create a node by creating a profile.

Each node has a node agent. A node agent is a process that handles communications with the resources within the node. One node agent process runs on each node to manage the running servers on that node, and to communicate and synchronize changes with the deployment manager of the cell. The node agent is an important process that allows for communication of administrative information, such as commands and configuration files, to reach the applications servers.

One final item to note is that a *node* does not equal a physical *machine*.

Slide 7



WebSphere Education

Network deployment runtime flow

Browser

Load balancer

HTTP server

Plug-in

Plug-in config

HTTP server

Plug-in

Plug-in config

Node A

AppSrv01

AppSrv02

Java client

Node B

AppSrv03

AppSrv04

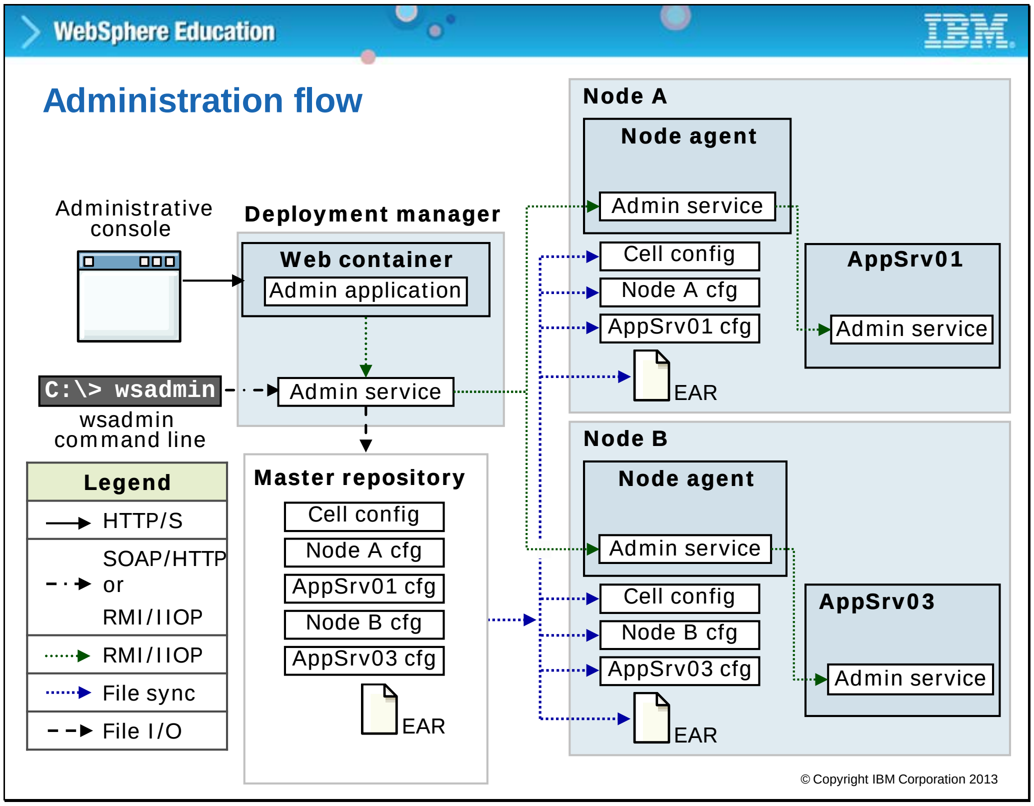
Application data

Legend

→	HTTP or HTTPS
⋯→	JDBC
⋯→	RMI/IIOP

The main theme with network deployment is distributed applications. While the "flow" of an application remains the same, there are significant additions to the run time of an application. This diagram illustrates how the request flow changes with a network deployment environment where you have multiple, clustered application servers that can share the workload.
First, note the load balancer, which allows for multiple external HTTP servers. Users point their browsers to the load balancer and their requests are workload managed to an HTTP server. When a client request comes in to the HTTP server, the HTTP server plug-in routes it to an application server according to configuration settings. Application servers can be clustered to provide scalability, failover, and a combination of both. In this case, the HTTP server plug-in load balances the request between the application servers that it is configured to serve. When the request enters the application server, the flow is identical to how it was in Express and base. Requests can also be load balanced across multiple EJB containers, if applicable. The Java client requests to EJBs can also be workload managed so that the requests do not all ~~hit~~ inundate one application server.

Slide 8



WebSphere Education

Administration flow

Administrative console

C:\> wsadmin

wsadmin command line

Deployment manager

Web container

Admin application

Admin service

Master repository

Cell config

Node A cfg

AppSrv01 cfg

Node B cfg

AppSrv03 cfg

EAR

Node A

Node agent

Admin service

Cell config

Node A cfg

AppSrv01 cfg

EAR

AppSrv01

Admin service

Node B

Node agent

Admin service

Cell config

Node B cfg

AppSrv03 cfg

EAR

AppSrv03

Admin service

Legend	
──▶	HTTP/S
-·▶	SOAP/HTTP or RMI/IIOP
·····▶	RMI/IIOP
·····▶	File sync
- -▶	File I/O

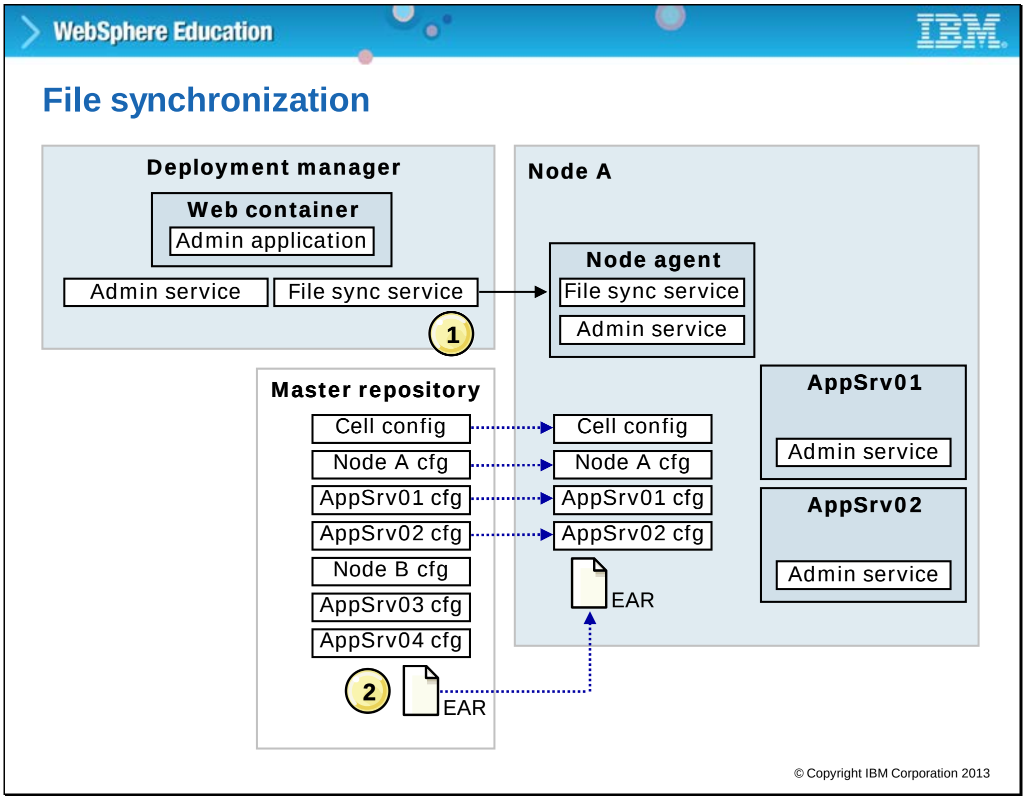
© Copyright IBM Corporation 2013

The administration of a network deployment environment is different than for a stand-alone application server. The administrative console and wsadmin are the two ways that the environment is administered. Rather than administering an application server directly, configuration changes and updates to application servers that are running in a cell are managed from a central point, ~~–~~the deployment manager.

The deployment manager is similar to an application server in that it is also running an Admin service, and it has a web container that is running the Admin application. You use the same web-based administrative console or wsadmin command-line interface. However, instead of using it with an application server, you use it with the deployment manager, which manages the configuration for all the nodes in the cell. The communication of these commands flows from the tools to the deployment manager, to the node agents, and to the application servers. This flow allows for the administration of multiple nodes from a single point, the deployment manager.

There is ***one*** master repository for the configuration files within a cell, and those files are associated with the deployment manager. All updates to the configuration files go through the deployment manager. The deployment manager updates only files that are changed in the master repository unless a full synchronization is used. A node agent, running on each node in the cell, also keeps a copy of the configuration repository locally and periodically synchronizes updates with the deployment manager.

Slide 9



WebSphere Education

File synchronization

Deployment manager

Web container

Admin application

Admin service

File sync service

1

Master repository

Cell config

Node A cfg

AppSrv01 cfg

AppSrv02 cfg

Node B cfg

AppSrv03 cfg

AppSrv04 cfg

2

EAR

Node A

Node agent

File sync service

Admin service

AppSrv01

Admin service

AppSrv02

Admin service

Cell config

Node A cfg

AppSrv01 cfg

AppSrv02 cfg

EAR

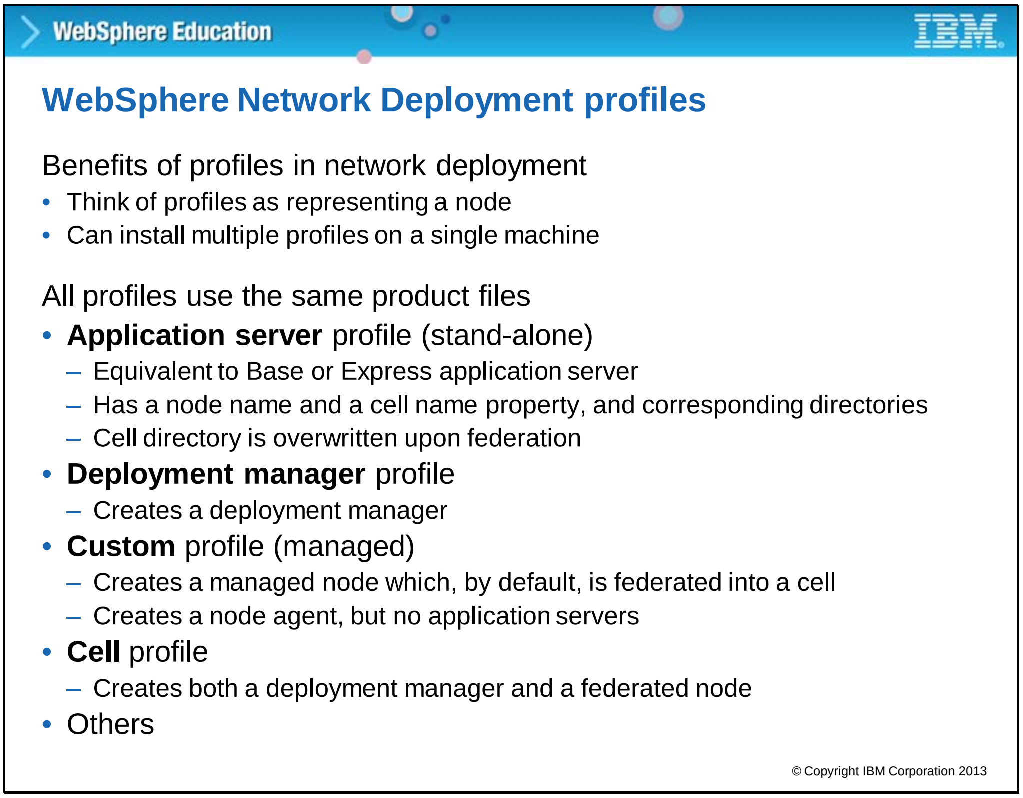
Each managed process, node agent, and deployment manager starts with its own set of configuration files. The deployment manager contains the master configuration. Any changes at the node agent or server level are local, and the MASTER configuration overrides them at the next synchronization.

During synchronization, the node agent asks for changes to master configuration. Then, any new or updated files are copied to the local configuration of the node.

Synchronization between the node agents and the deployment manager can be configured to occur in a number of ways.

Node agents synchronize their files with the master copy either automatically or manually. Automatic synchronization can be done at startup or scheduled periodically. Manual synchronization is done with the administrative console or from the command line.

Slide 10



WebSphere Education

WebSphere Network Deployment profiles

Benefits of profiles in network deployment

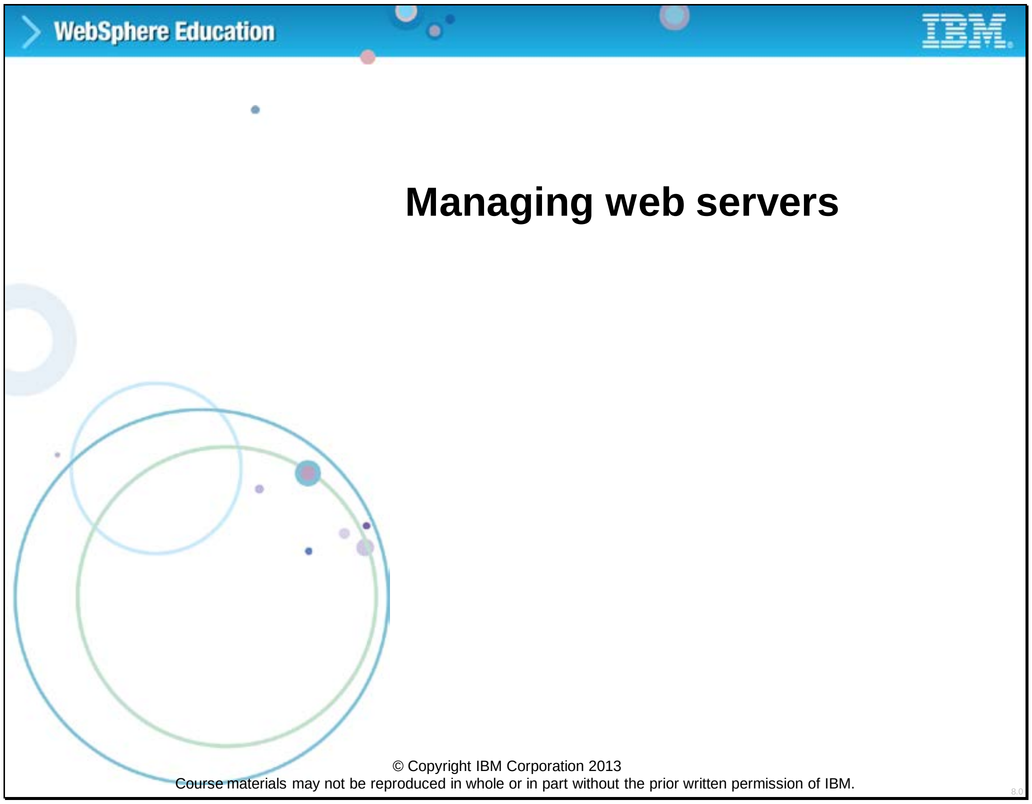
- Think of profiles as representing a node
- Can install multiple profiles on a single machine

All profiles use the same product files

- **Application server** profile (stand-alone)
  - Equivalent to Base or Express application server
  - Has a node name and a cell name property, and corresponding directories
  - Cell directory is overwritten upon federation
- **Deployment manager** profile
  - Creates a deployment manager
- **Custom** profile (managed)
  - Creates a managed node which, by default, is federated into a cell
  - Creates a node agent, but no application servers
- **Cell** profile
  - Creates both a deployment manager and a federated node
- Others

The addition of Network Deployment to this discussion does not change the definition of a profile. The WebSphere configuration is still built by creating profiles, which consist of product binary files and configuration files. The deployment manager profile is a special type of node that manages the administrative domain of a cell. The process of adding a node to a cell is called federation. To set up a network deployment cell, you create profiles for the nodes in the cell. For example, you would create a deployment manager profile for the deployment manager node. Alternatively, you can use the cell profile to create both a deployment manager and the first federated node in the cell. The application server profile is for stand-alone servers. However, after you create the deployment manager, you can federate any existing application servers into the cell, if you want. The custom profile creates a node with a node agent but no application server. The custom profile can be used to add managed nodes to a cell, for example, a web server without an application server.

Slide 11

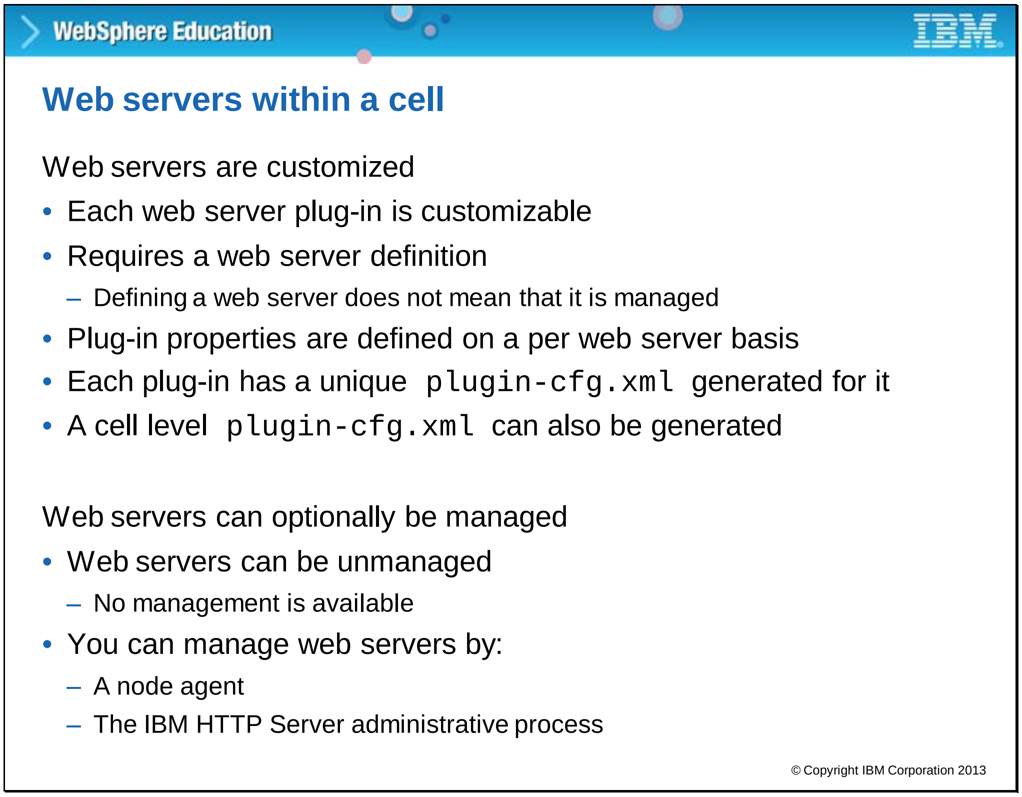


WebSphere Education

Managing web servers

Topic: Managing web servers. In this topic, you learn how to administer web servers from a WebSphere Application Server environment.

Slide 12



WebSphere Education
IBM

Web servers within a cell

Web servers are customized

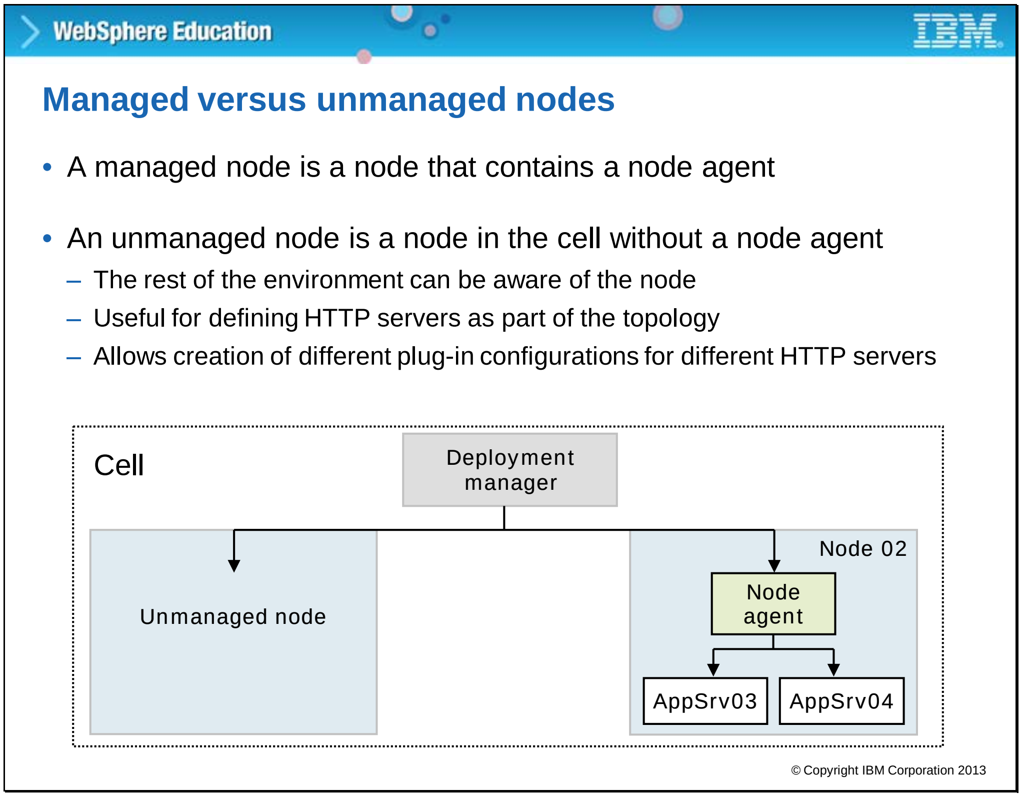
- Each web server plug-in is customizable
- Requires a web server definition
  – Defining a web server does not mean that it is managed
- Plug-in properties are defined on a per web server basis
- Each plug-in has a unique `plugin-cfg.xml` generated for it
- A cell level `plugin-cfg.xml` can also be generated

Web servers can optionally be managed

- Web servers can be unmanaged
  – No management is available
- You can manage web servers by:
  – A node agent
  – The IBM HTTP Server administrative process

Web servers within a cell are customized. Each web server plug-in is customizable and requires a web server definition. Defining a web server does not mean that it is managed; plug-in properties are defined on a web server-by-web server basis. Configuration information for each plug-in is stored in its own plug-in configuration XML file. Each plug-in has a unique plugin-cfg.xml file that is generated for it; a cell level plugin-cfg.xml can also be generated. This file can be generated, and in certain cases the file can be propagated out to the web server as needed.
Web servers can be managed or they can be unmanaged. A node agent can manage the web servers, or if the web server is an IBM HTTP Server, by the IHS admin process. Otherwise, a web server is unmanaged.

Slide 13

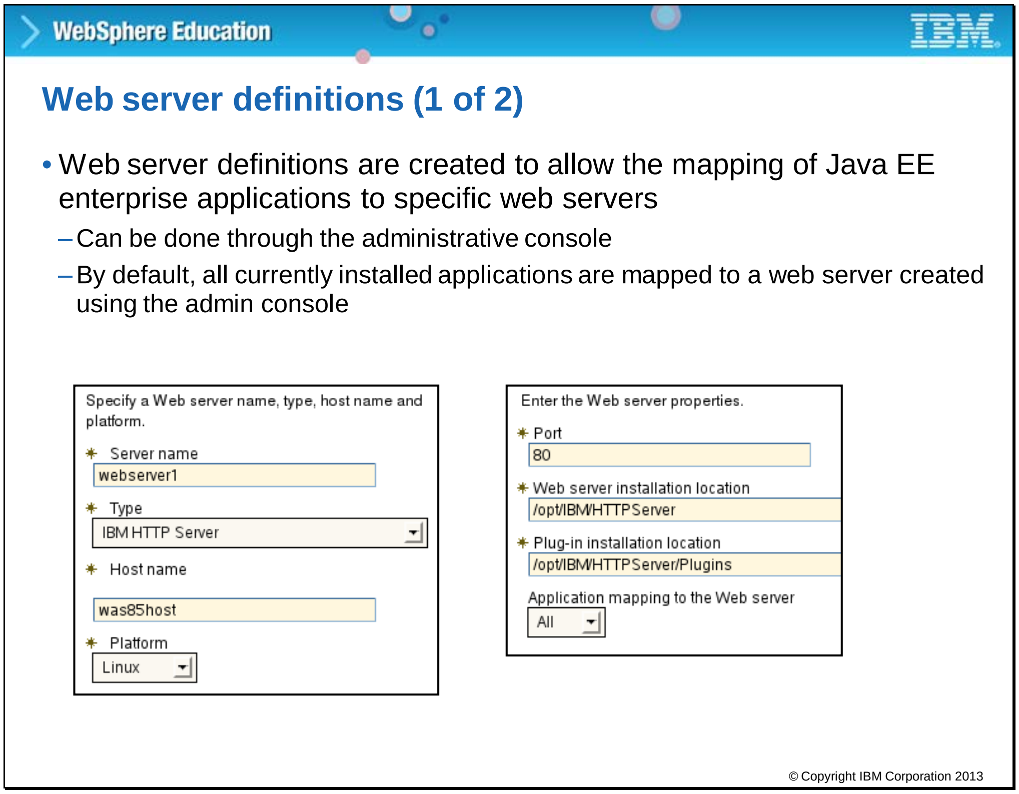


Managed versus unmanaged nodes

- A managed node is a node that contains a node agent
- An unmanaged node is a node in the cell without a node agent
  - The rest of the environment can be aware of the node
  - Useful for defining HTTP servers as part of the topology
  - Allows creation of different plug-in configurations for different HTTP servers

A node with a node agent is called a managed node. It is also possible to have a node in a cell that does not have a node agent, which is called an unmanaged node. This configuration is useful in cases where you want the rest of the network deployment environment to be aware of the node, but a node agent is not required. For example, you can run a web server on an unmanaged node.

Slide 14



WebSphere Education

Web server definitions (1 of 2)

- Web server definitions are created to allow the mapping of Java EE enterprise applications to specific web servers
  - Can be done through the administrative console
  - By default, all currently installed applications are mapped to a web server created using the admin console

Specify a Web server name, type, host name and platform.

* Server name: webserver1

* Type: IBM HTTP Server

* Host name: was85host

* Platform: Linux

Enter the Web server properties.

* Port: 80

* Web server installation location: /opt/IBM/HTTPServer

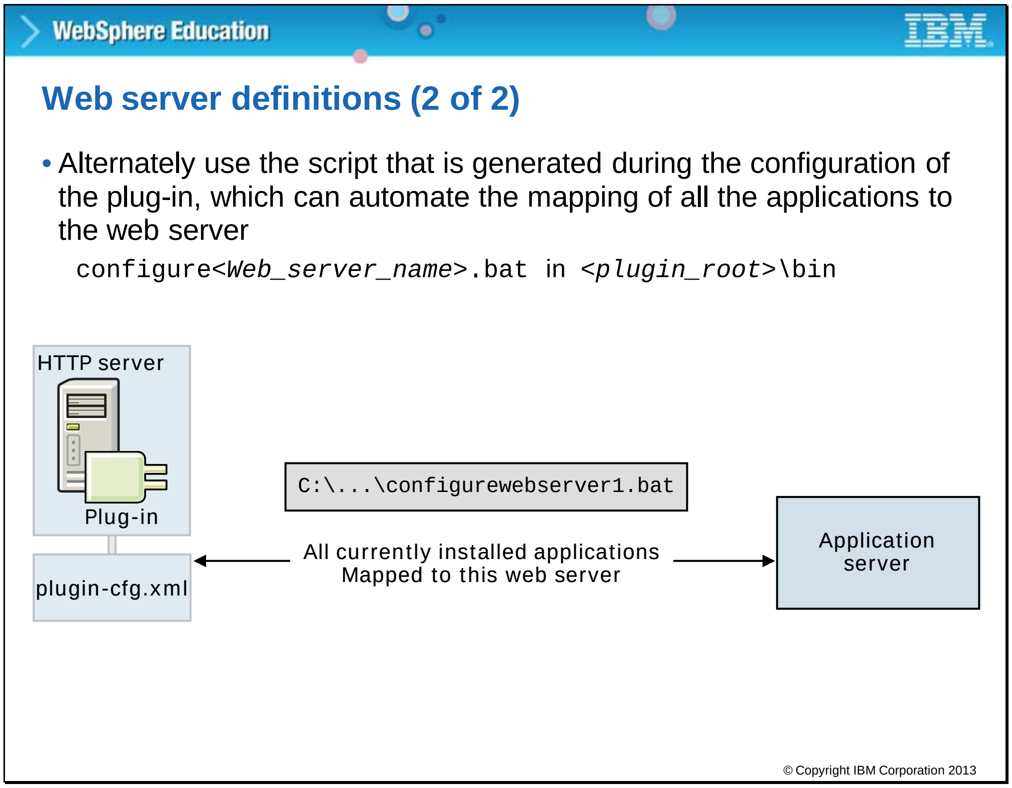
* Plug-in installation location: /opt/IBM/HTTPServer/Plugins

Application mapping to the Web server: All

© Copyright IBM Corporation 2013

Just as modules for an enterprise application are mapped to one or more application servers, they also are mapped to one or more web servers. Web server definitions are created to allow the mapping of Java EE enterprise applications to specific web servers. When a web server is defined, by default, all currently installed applications are mapped to the web server.

Slide 15



WebSphere Education

Web server definitions (2 of 2)

- Alternately use the script that is generated during the configuration of the plug-in, which can automate the mapping of all the applications to the web server

  `configure<Web_server_name>.bat in <plugin_root>\bin`

HTTP server

Plug-in

plugin-cfg.xml

```
C:\...\configurewebserver1.bat
```

All currently installed applications
Mapped to this web server

Application server

© Copyright IBM Corporation 2013

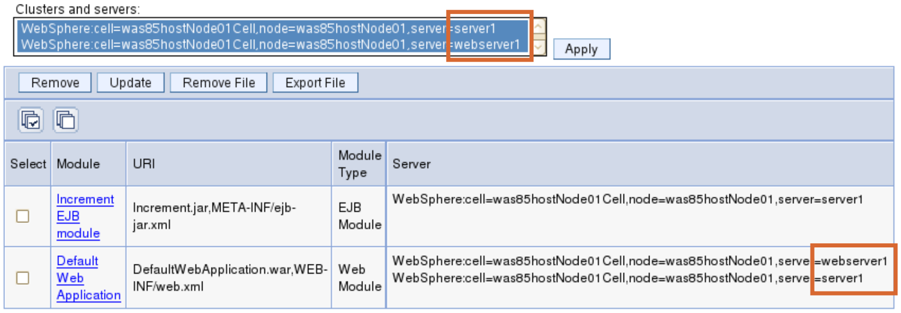
When a web server plug-in is configured, a script is generated which defines the web server. This script can be used instead of using the administrative console to define the web server. The script also maps all currently installed applications to the defined web server.

Slide 16

WebSphere Education

Web server custom plugin-cfg.xml

- Mapping the applications to specific web servers causes the `plugin-cfg.xml` files for only those web servers to include the information for those applications
  - Target-specific web server applications that run in a cell
  - Deployment manager automatically generates them



Clusters and servers:

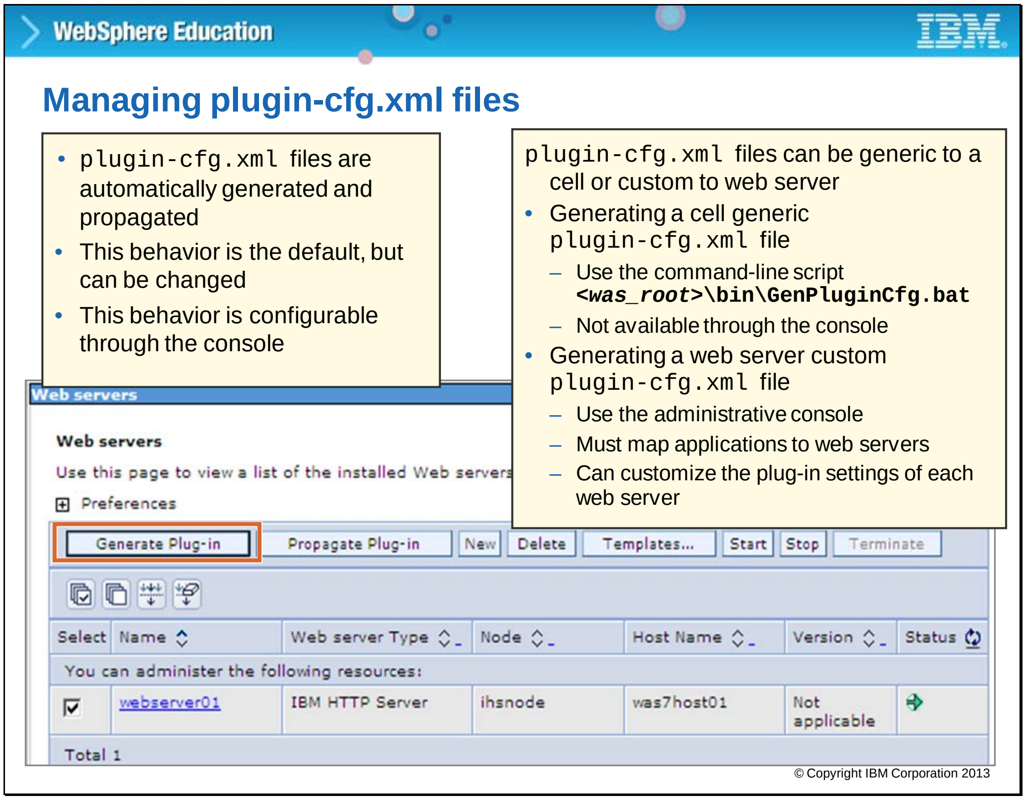
```
WebSphere:cell=was85hostNode01Cell,node=was85hostNode01,server=server1
WebSphere:cell=was85hostNode01Cell,node=was85hostNode01,server=webserver1
```

Select	Module	URI	Module Type	Server
□	Increment EJB module	Increment.jar,META-INF/ejb-jar.xml	EJB Module	WebSphere:cell=was85hostNode01Cell,node=was85hostNode01,server=server1
□	Default Web Application	DefaultWebApplication.war,WEB-INF/web.xml	Web Module	WebSphere:cell=was85hostNode01Cell,node=was85hostNode01,server=webserver1 WebSphere:cell=was85hostNode01Cell,node=was85hostNode01,server=server1

Application module explicitly mapped to a web server

Specific applications can be mapped to some web server definitions and not to others. This mapping causes the custom plug-in configuration files for only those web servers to include the information for those applications.

Slide 17



Managing plugin-cfg.xml files

- `plugin-cfg.xml` files are automatically generated and propagated
- This behavior is the default, but can be changed
- This behavior is configurable through the console

`plugin-cfg.xml` files can be generic to a cell or custom to web server

- Generating a cell generic `plugin-cfg.xml` file
  - Use the command-line script **`<was_root>\bin\GenPluginCfg.bat`**
  - Not available through the console
- Generating a web server custom `plugin-cfg.xml` file
  - Use the administrative console
  - Must map applications to web servers
  - Can customize the plug-in settings of each web server

Web servers

Web servers

Use this page to view a list of the installed Web servers

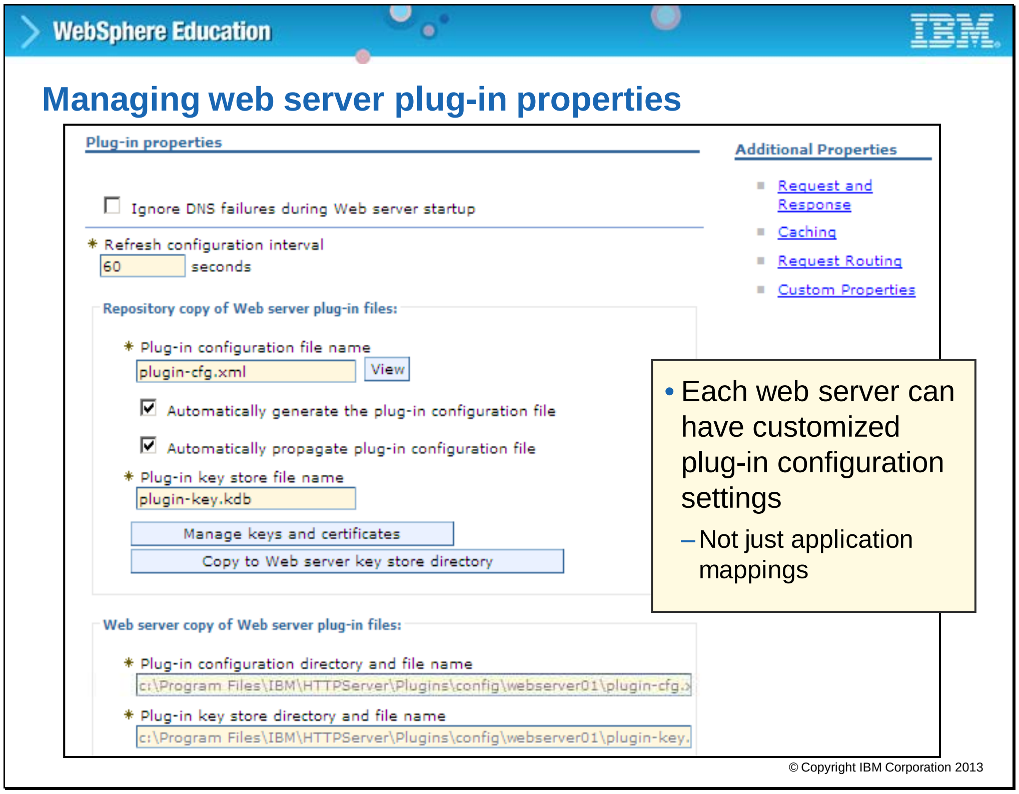
Select	Name	Web server Type	Node	Host Name	Version	Status
You can administer the following resources:						
☑	webserver01	IBM HTTP Server	ihsnode	was7host01	Not applicable	➔
Total 1						

© Copyright IBM Corporation 2013

The default behavior of WebSphere Application Server is to automatically generate and propagate the plug-in XML file, but this behavior can be configured by using the administrative console. The plug-in configuration file can be generic to a cell or customized for a web server. You can use the command-line scripts to generate a generic plug-in configuration file. To generate a cell generic plugin-cfg.xml file, use the command-line script: <*was_root*>\bin\GenPluginCfg.bat. This action is not available through the console.

You can use the administrative console to generate a plug-in configuration file that is customized for a web server. To customize, you can map applications to various web servers, and you can customize the plug-in settings for each web server.

Slide 18



Managing web server plug-in properties

Plug-in properties

☐ Ignore DNS failures during Web server startup

* Refresh configuration interval
60 seconds

Repository copy of Web server plug-in files:

* Plug-in configuration file name
plugin-cfg.xml [View]

☑ Automatically generate the plug-in configuration file

☑ Automatically propagate plug-in configuration file

* Plug-in key store file name
plugin-key.kdb

[Manage keys and certificates]

[Copy to Web server key store directory]

Web server copy of Web server plug-in files:

* Plug-in configuration directory and file name
c:\Program Files\IBM\HTTPServer\Plugins\config\webserver01\plugin-cfg.x

* Plug-in key store directory and file name
c:\Program Files\IBM\HTTPServer\Plugins\config\webserver01\plugin-key.

Additional Properties

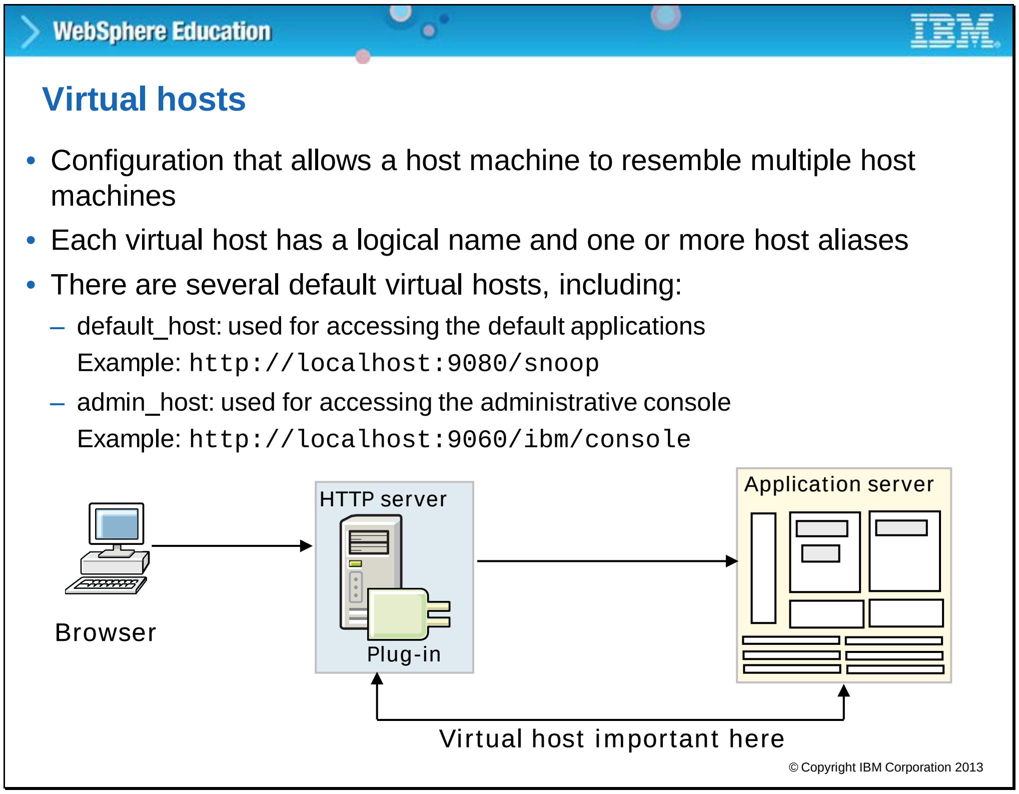
- Request and Response
- Caching
- Request Routing
- Custom Properties

- Each web server can have customized plug-in configuration settings
  - Not just application mappings

© Copyright IBM Corporation 2013

This screen capture of the administrative console shows the plug-in properties for a web server. You can examine the various settings for the plug-in from this page.
Each web server can have plug-in configuration settings customized, not just application mappings. The links on the right side of the page under **Additional Properties**, provide access to pages for changing the plug-in properties.

Slide 19



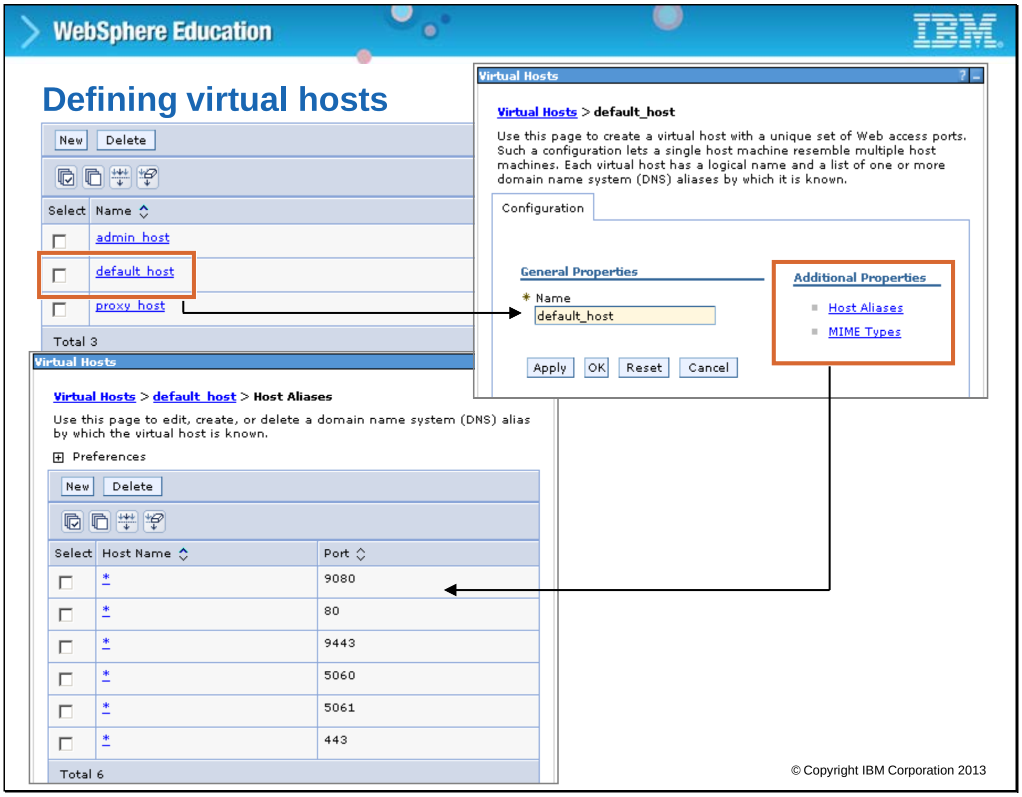
WebSphere Education

Virtual hosts

- Configuration that allows a host machine to resemble multiple host machines
- Each virtual host has a logical name and one or more host aliases
- There are several default virtual hosts, including:
  - default_host: used for accessing the default applications
    Example: `http://localhost:9080/snoop`
  - admin_host: used for accessing the administrative console
    Example: `http://localhost:9060/ibm/console`

Virtual hosts are supported in WebSphere to allow one host to resemble multiple hosts and applications. The virtual hosts definition, for the application server, determines ***not*** which ports are listened to, but what ports are acceptable for the incoming URL. Since most requests come in from the external web server, the ports that are specified either explicitly or implicitly are 80 and 443. If an administrator wants to allow other ports to be used, the ports must be specified on the virtual host to which the application is mapped.
The virtual host is associated with the cell, not a single server.
Each virtual host has a logical name and one or more host aliases. Each alias includes a host name and port number. Wildcards can also be used in an alias, for example: *.80.
The default host is used for accessing the default applications. The admin host is used for accessing the administrative console.

Slide 20



Defining virtual hosts

Select	Name
□	admin_host
□	default_host
□	proxy_host
Total 3	

Virtual Hosts > default_host

Use this page to create a virtual host with a unique set of Web access ports. Such a configuration lets a single host machine resemble multiple host machines. Each virtual host has a logical name and a list of one or more domain name system (DNS) aliases by which it is known.

* Name: default_host

Virtual Hosts > default_host > Host Aliases

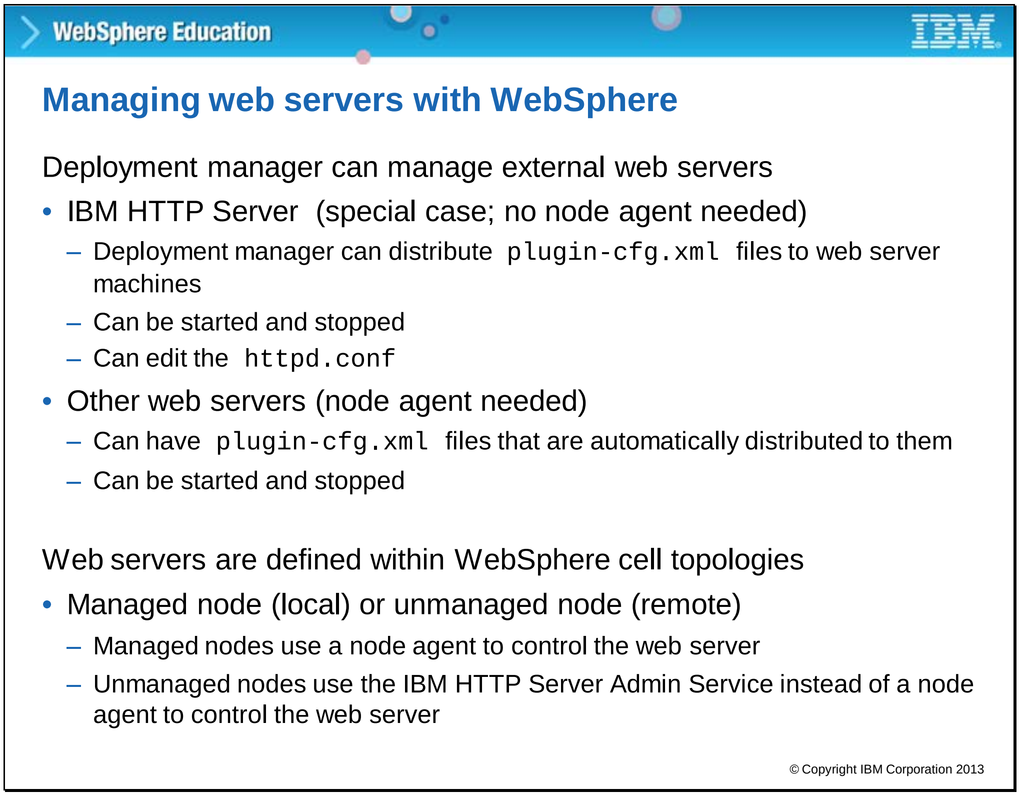
Use this page to edit, create, or delete a domain name system (DNS) alias by which the virtual host is known.

Select	Host Name	Port
□	*	9080
□	*	80
□	*	9443
□	*	5060
□	*	5061
□	*	443
Total 6		

© Copyright IBM Corporation 2013

This screen capture shows how to define a virtual host by using the administrative console. On the first screen, you can click **New** to create a new virtual host or click one of the virtual hosts that are listed to edit it. On the next page, click the **Host aliases** link to create or edit host aliases for that virtual host. This slide shows the definition for "default_host." Notice that the host aliases specified are "*****." The wildcard character means that a request for any host name on ports 9080, 80, and 9443 is forwarded to this host. When applications are installed, web modules within those applications must be mapped to a virtual host.

Slide 21



WebSphere Education

Managing web servers with WebSphere

Deployment manager can manage external web servers

- IBM HTTP Server (special case; no node agent needed)
  - Deployment manager can distribute `plugin-cfg.xml` files to web server machines
  - Can be started and stopped
  - Can edit the `httpd.conf`
- Other web servers (node agent needed)
  - Can have `plugin-cfg.xml` files that are automatically distributed to them
  - Can be started and stopped

Web servers are defined within WebSphere cell topologies

- Managed node (local) or unmanaged node (remote)
  - Managed nodes use a node agent to control the web server
  - Unmanaged nodes use the IBM HTTP Server Admin Service instead of a node agent to control the web server

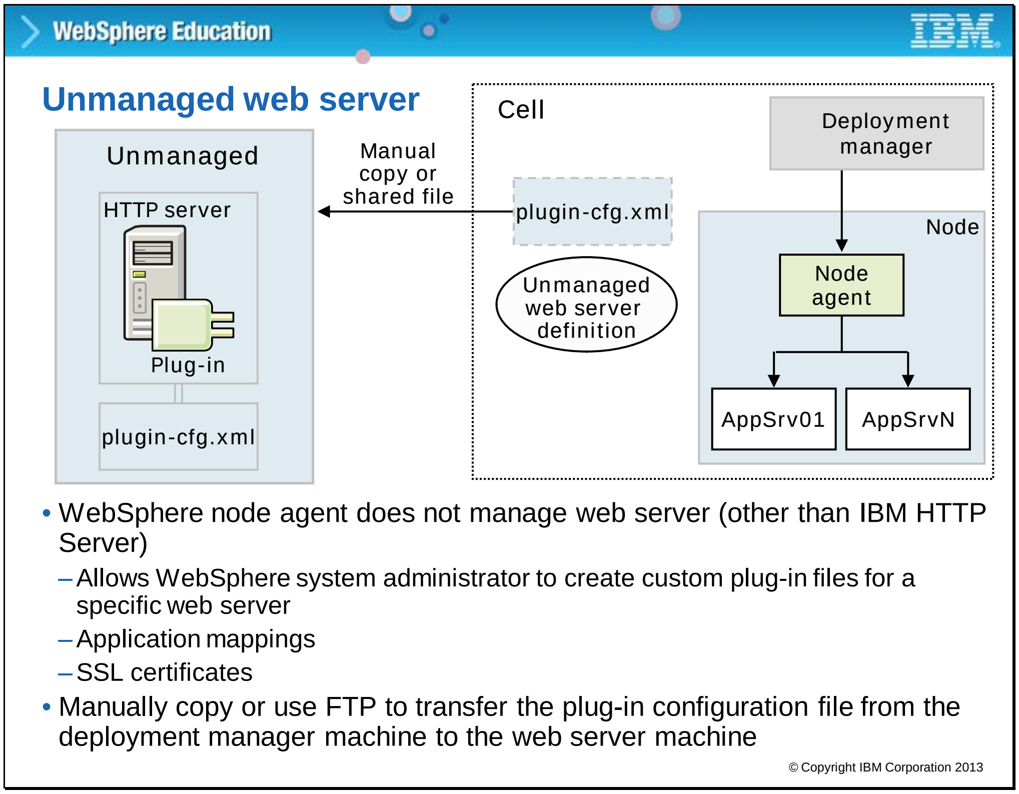
External web servers can be managed by using the deployment manager. With the IBM HTTP Server, no node agent is needed. The deployment manager can communicate directly with the IBM HTTP Server admin service. Other web servers can be managed with the deployment manager by installing a node agent on the web server.

Managed web servers can be started and stopped from the console and can have plug-in configuration files that are automatically distributed to them.

There are three types of nodes upon which you can create and manage a web server. Over the next several pages, three common web server management scenarios are presented:

- Using a web server as an unmanaged node
- Using IBM HTTP Server as an unmanaged node
- Using a web server as a managed node

Slide 22



WebSphere Education

Unmanaged web server

Unmanaged

HTTP server

Plug-in

plugin-cfg.xml

Manual copy or shared file

Cell

plugin-cfg.xml

Unmanaged web server definition

Deployment manager

Node

Node agent

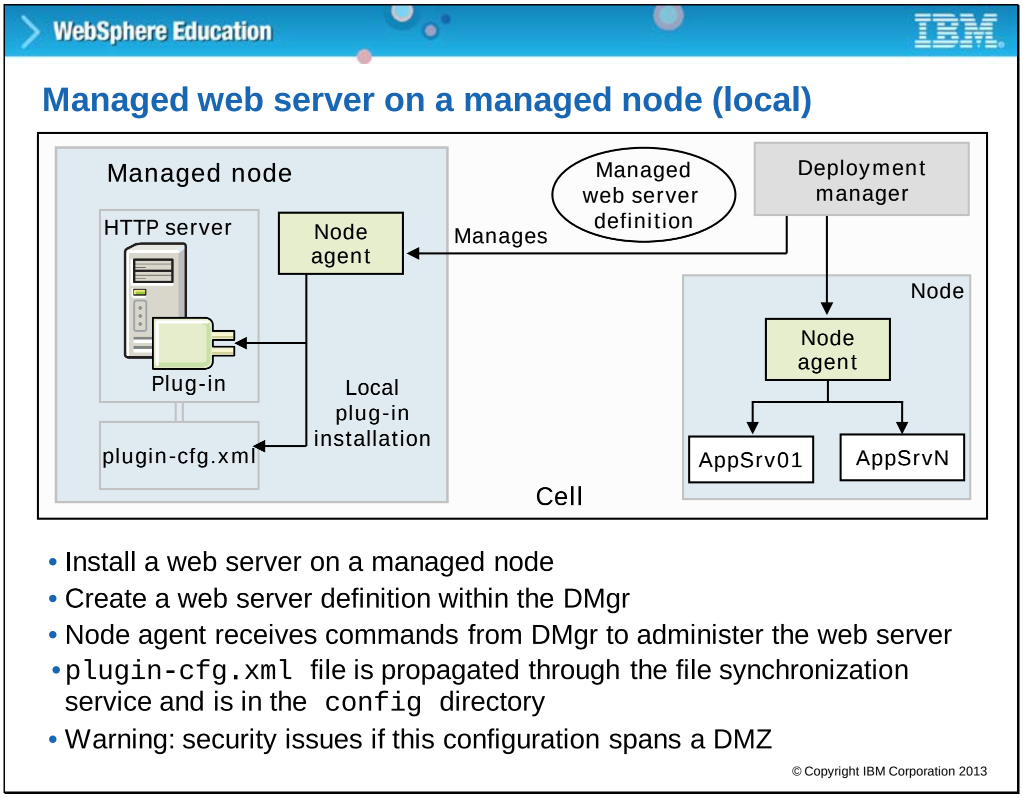
AppSrv01

AppSrvN

- WebSphere node agent does not manage web server (other than IBM HTTP Server)
  - Allows WebSphere system administrator to create custom plug-in files for a specific web server
  - Application mappings
  - SSL certificates
- Manually copy or use FTP to transfer the plug-in configuration file from the deployment manager machine to the web server machine

This diagram illustrates the relationship between an unmanaged web server and a WebSphere cell. The web server is not an IBM HTTP Server, and there is no node agent on the web server. The web server is registered as an unmanaged node in this WebSphere configuration. This scenario is common for web servers that are installed outside the firewall or in a DMZ where no WebSphere Application Server exists. The implication with this scenario is that all management of the web server occurs manually, which is outside the control of WebSphere Application Server. There is no automatic propagation of the plug-in configuration file. The plug-in configuration file must be manually generated and copied over to the web server. There is no way to start and stop the web server from the WebSphere administration tools. Command-line tools for the web server must be used.

Slide 23



Managed web server on a managed node (local)

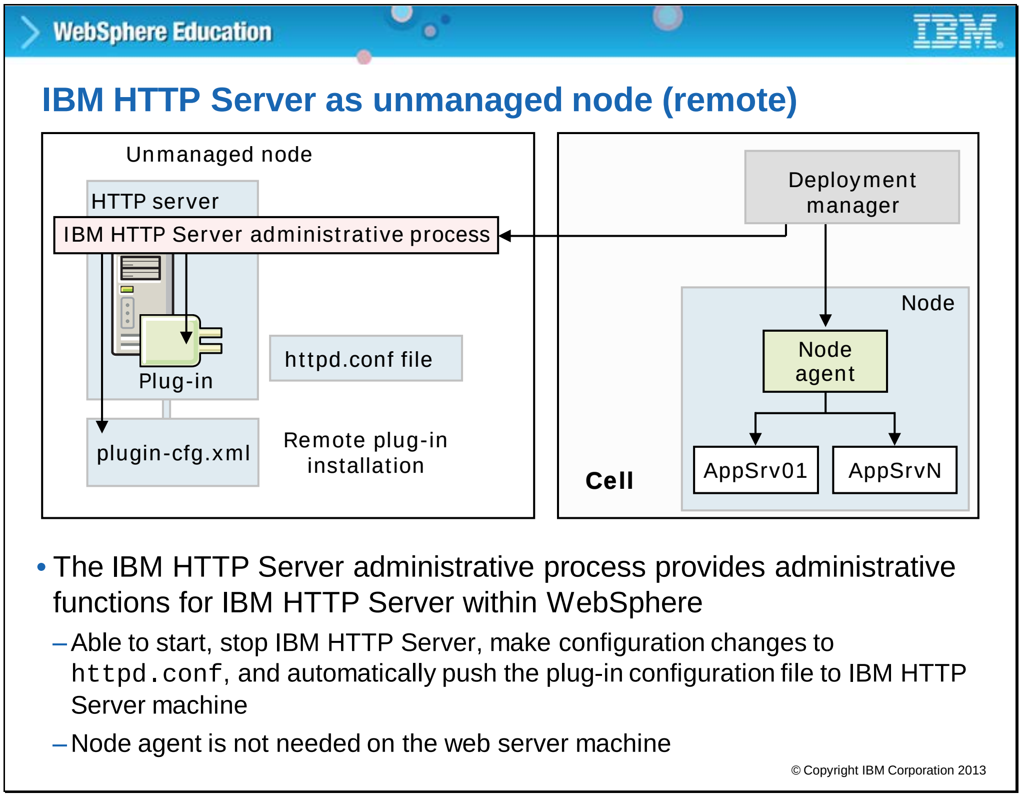
- Install a web server on a managed node
- Create a web server definition within the DMgr
- Node agent receives commands from DMgr to administer the web server
- `plugin-cfg.xml` file is propagated through the file synchronization service and is in the `config` directory
- Warning: security issues if this configuration spans a DMZ

This diagram illustrates the relationship between a managed web server and a WebSphere cell. In this case, a node agent is installed on the web server. A node agent communicates with the web server from the administrative tools of WebSphere. The deployment manager manages the web server through the node agent.

In this configuration, you can start or stop the web server from the administrative console of the deployment manager. The plug-in configuration file can be automatically generated and propagated to the web server through the managed web server definition in the cell.

This configuration can be used when the web server is installed on the same system as the WebSphere Application Server installation. It is a common scenario for behind a firewall where a WebSphere node can be installed. It might be undesirable to use this configuration, since access to the node agent in a DMZ can compromise security.

Slide 24



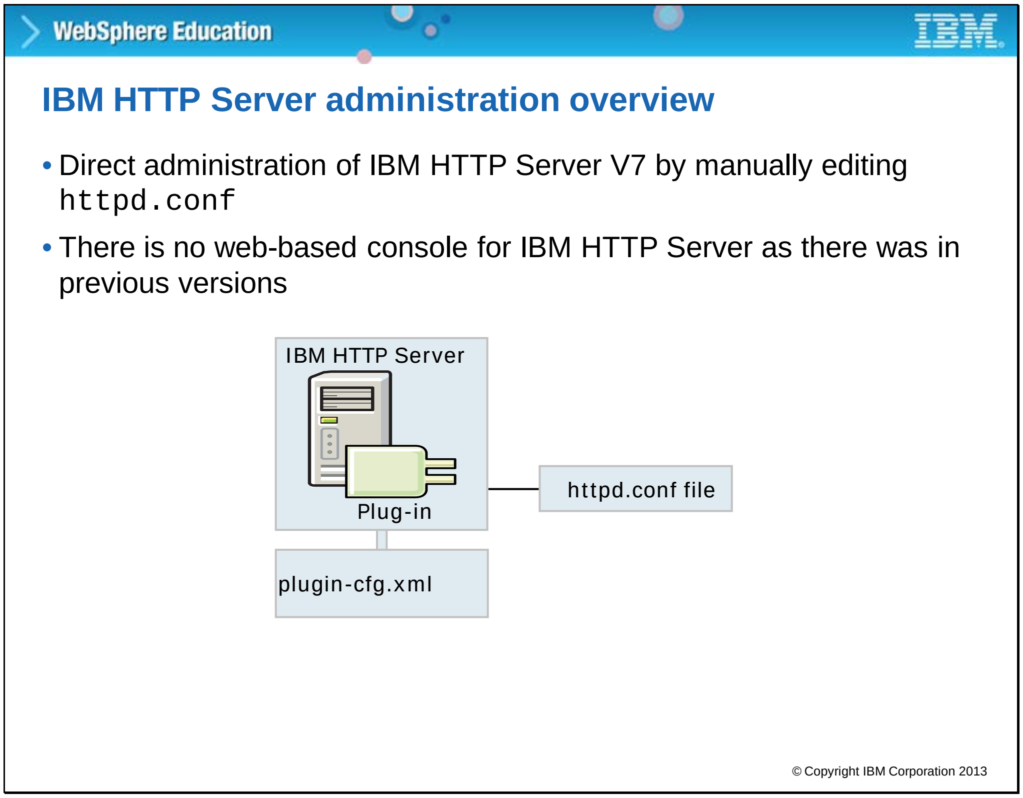
WebSphere Education

IBM HTTP Server as unmanaged node (remote)

- The IBM HTTP Server administrative process provides administrative functions for IBM HTTP Server within WebSphere
  - Able to start, stop IBM HTTP Server, make configuration changes to `httpd.conf`, and automatically push the plug-in configuration file to IBM HTTP Server machine
  - Node agent is not needed on the web server machine

In this case, the web server is an IBM HTTP Server and no node agent is installed, so it is considered to be an unmanaged web server.
The IBM HTTP Server can be managed completely from the deployment manager. The deployment manager communicates with the IBM HTTP Server administrative process that runs on the node with IBM HTTP Server. There are actually two Apache processes on the IBM HTTP Server system: one running the administrative services, and one running the actual web server. The deployment manager communicates directly with the IBM HTTP Server administrative process, and automatically propagates the plug-in configuration file to the web server. You can also start and stop the web server process from the administrative console. However, you still must manually start the administrative process.

Slide 25



WebSphere Education

IBM HTTP Server administration overview

- Direct administration of IBM HTTP Server V7 by manually editing `httpd.conf`
- There is no web-based console for IBM HTTP Server as there was in previous versions

IBM HTTP Server

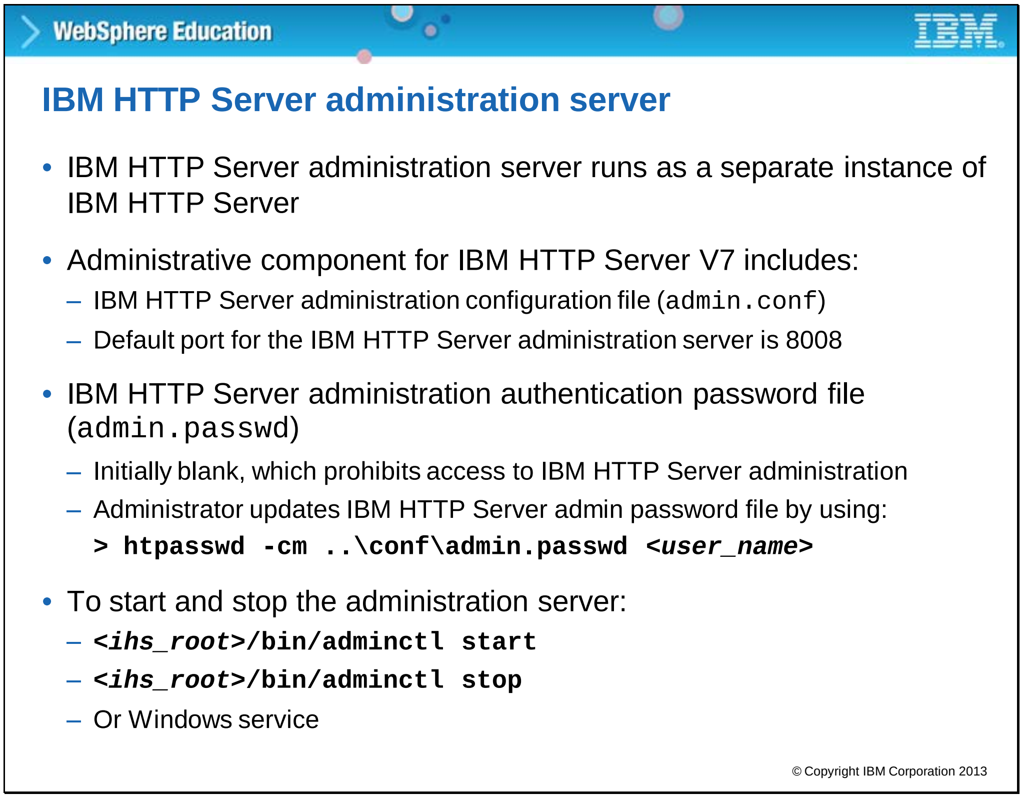
Plug-in

httpd.conf file

plugin-cfg.xml

You can still administer the IBM HTTP Server directly by editing the httpd.conf configuration file. There is no longer a GUI administrative client as in previous versions of the IBM HTTP Server.

Slide 26



WebSphere Education
IBM

IBM HTTP Server administration server

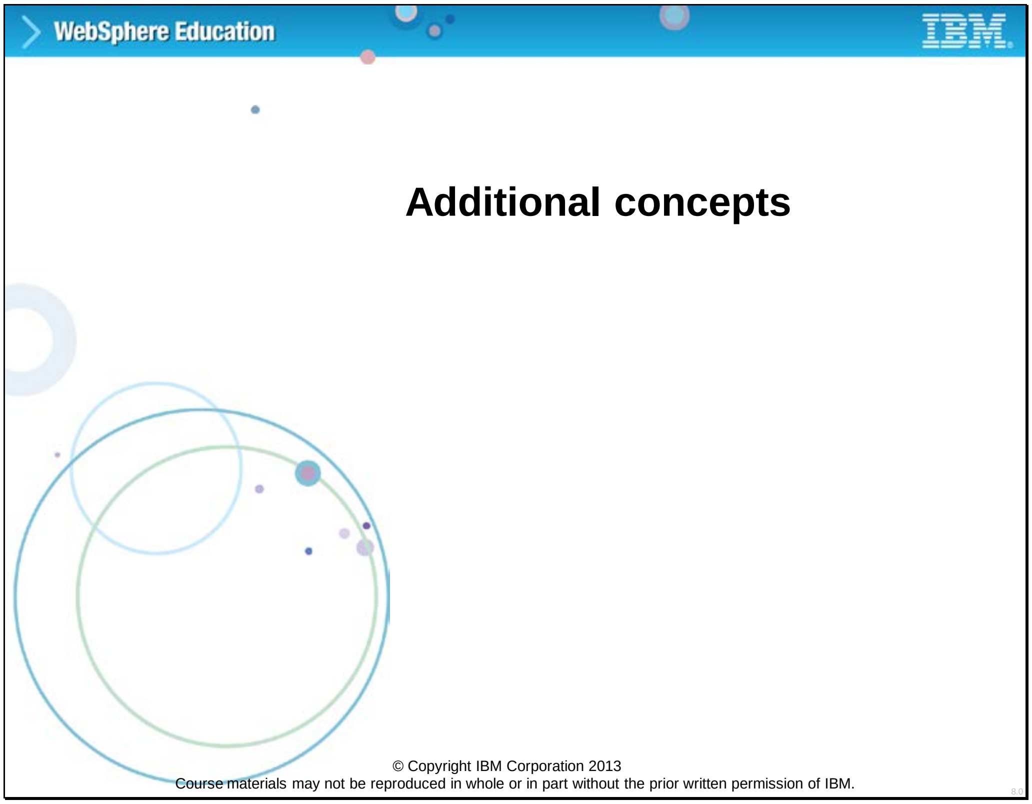
- IBM HTTP Server administration server runs as a separate instance of IBM HTTP Server
- Administrative component for IBM HTTP Server V7 includes:
  - IBM HTTP Server administration configuration file (`admin.conf`)
  - Default port for the IBM HTTP Server administration server is 8008
- IBM HTTP Server administration authentication password file (`admin.passwd`)
  - Initially blank, which prohibits access to IBM HTTP Server administration
  - Administrator updates IBM HTTP Server admin password file by using:

```
> htpasswd -cm ..\conf\admin.passwd <user_name>
```

- To start and stop the administration server:
  - **<*ihs_root*>/bin/adminctl start**
  - **<*ihs_root*>/bin/adminctl stop**
  - Or Windows service

The IBM HTTP Server administrative service runs as a separate instance of the IBM HTTP Server. Some other IBM HTTP Server artifacts to be aware of include: the administrative configuration file (admin.conf), the administrative server port (8008), and the authentication password file (admin.passwd). The authentication password file is initially blank and can be updated by using the command that is displayed on the slide. The commands for starting and stopping the administrative server are also shown here.

Slide 27



WebSphere Education

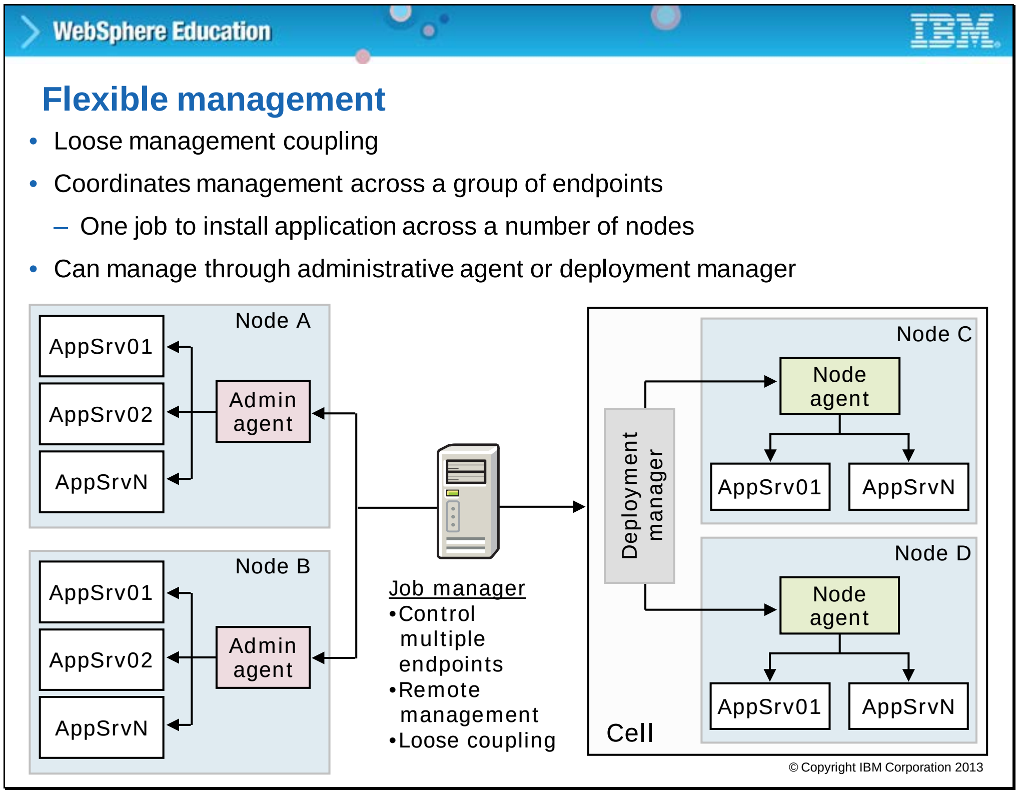
IBM

Additional concepts

8.0

Topic: Additional concepts. In this topic, you learn about various network deployment concepts.

Slide 28



WebSphere Education

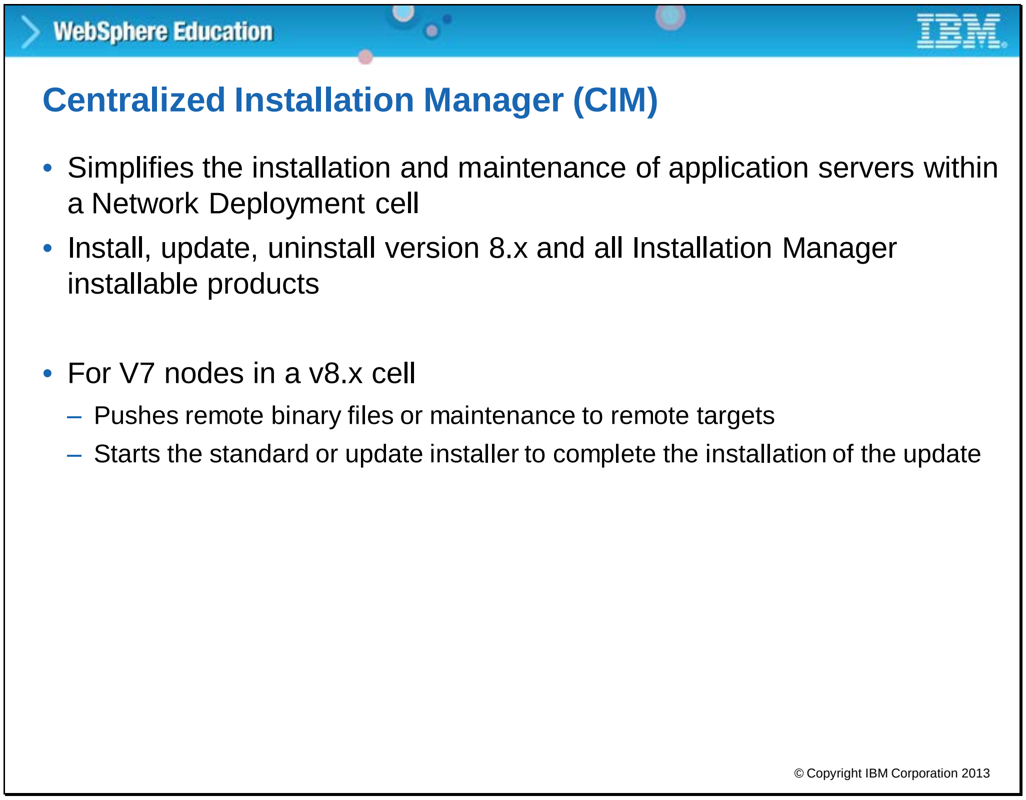
Flexible management

- Loose management coupling
- Coordinates management across a group of endpoints
  - One job to install application across a number of nodes
- Can manage through administrative agent or deployment manager

Job manager
- Control multiple endpoints
- Remote management
- Loose coupling

Flexible management provides administrators with alternative ways of managing a WebSphere Application Server environment. It is based on a loose coupling model that is better suited for certain types of environments where a network deployment cell might not be practical or wanted. A job manager can be used to coordinate management across a group of endpoints. They can be a combination of stand-alone application servers, federated nodes in a cell, or a deployment manager. An administrative agent must be installed on a server to manage it with the job manager. Flexible management is covered later in this course.

Slide 29



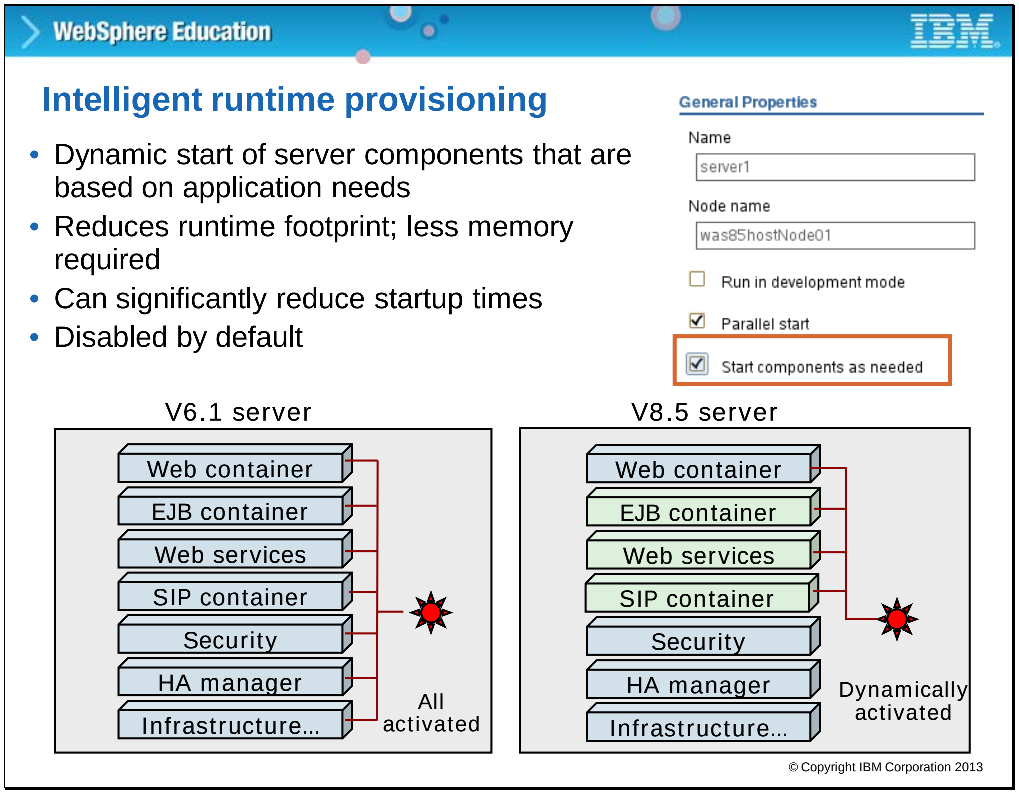
WebSphere Education
IBM

Centralized Installation Manager (CIM)

- Simplifies the installation and maintenance of application servers within a Network Deployment cell
- Install, update, uninstall version 8.x and all Installation Manager installable products
- For V7 nodes in a v8.x cell
  - Pushes remote binary files or maintenance to remote targets
  - Starts the standard or update installer to complete the installation of the update

The centralized installation manager, or CIM, simplifies the installation and maintenance of application servers within a Network Deployment cell.
The centralized installation manager can install and control IBM Installation Manager on other hosts. The Job manager submits CIM jobs to manager offerings.
For version 7 nodes in a version 8 cell, the centralized installation manager copies installation of fix files to a remote host and starts the appropriate installation program on the remote host.

Slide 30



WebSphere Education

Intelligent runtime provisioning

- Dynamic start of server components that are based on application needs
- Reduces runtime footprint; less memory required
- Can significantly reduce startup times
- Disabled by default

General Properties

Name: server1

Node name: was85hostNode01

□ Run in development mode

☑ Parallel start

☑ Start components as needed

V6.1 server

Web container
EJB container
Web services
SIP container
Security
HA manager
Infrastructure...

All activated

V8.5 server

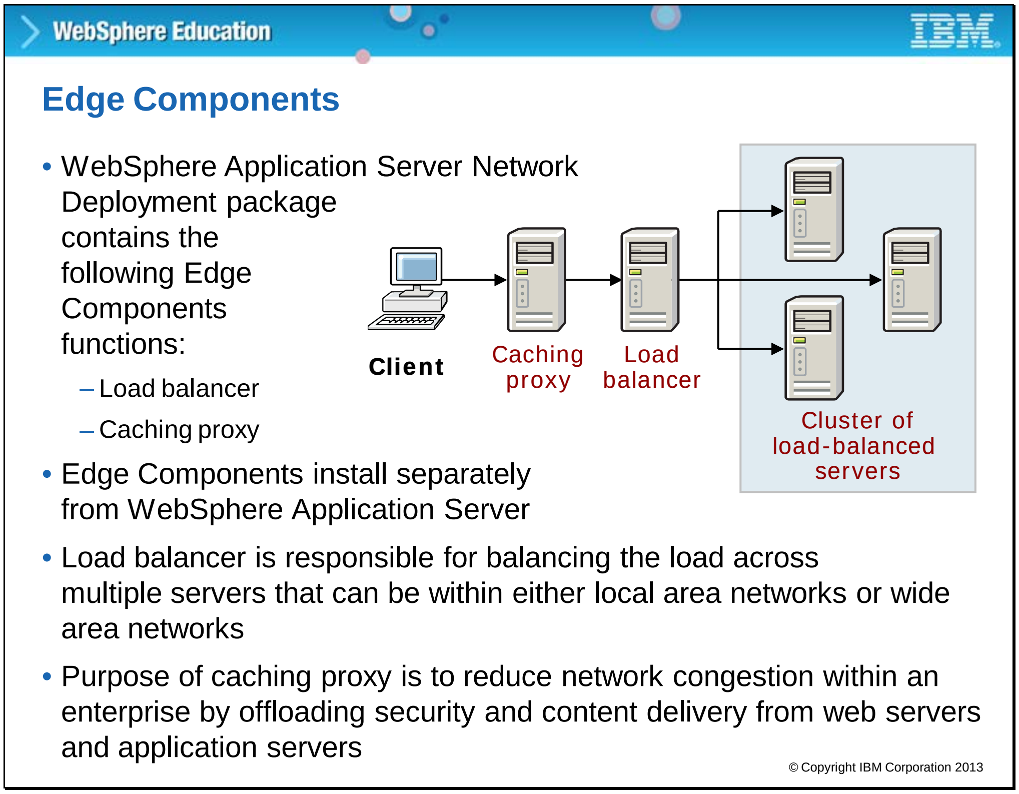
Web container
EJB container
Web services
SIP container
Security
HA manager
Infrastructure...

Dynamically activated

© Copyright IBM Corporation 2013

Intelligent runtime provisioning is a feature in WebSphere Application Server that dynamically starts application server components that are based on application needs, rather than starting all components. This feature reduces the runtime footprint, requires less memory, and can significantly reduce startup time. To enable this feature, check "Start components as needed" in the server details panel.

Slide 31



WebSphere Education

IBM

Edge Components

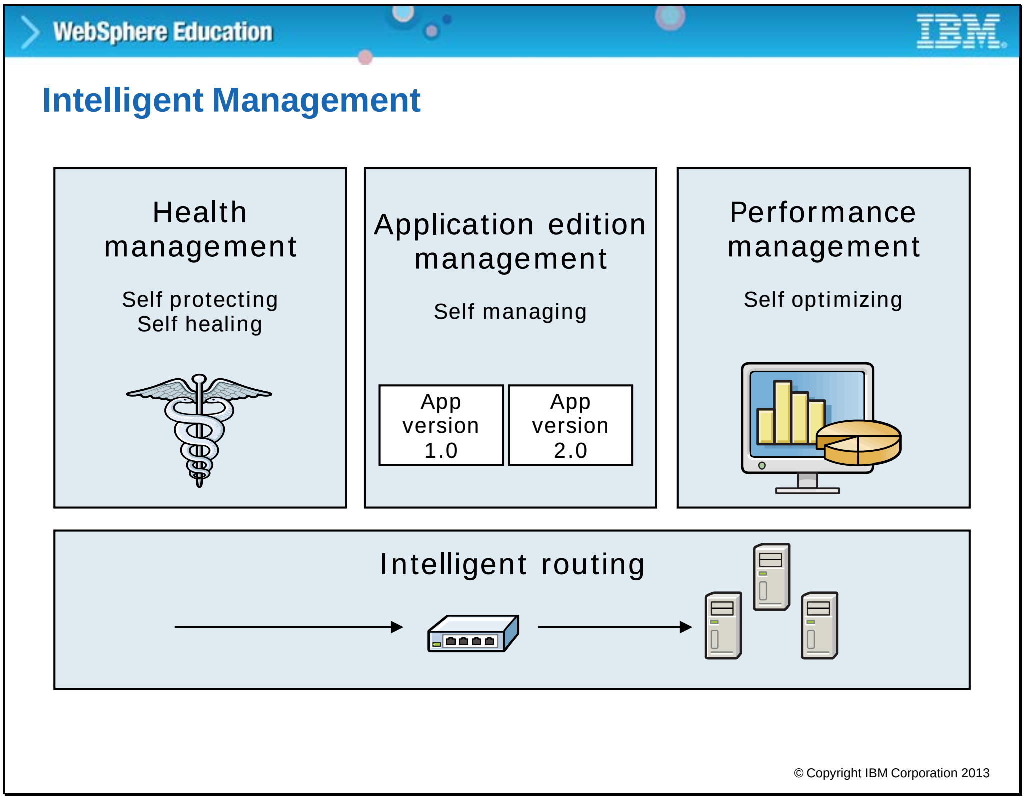
- WebSphere Application Server Network Deployment package contains the following Edge Components functions:
  - Load balancer
  - Caching proxy
- Edge Components install separately from WebSphere Application Server
- Load balancer is responsible for balancing the load across multiple servers that can be within either local area networks or wide area networks
- Purpose of caching proxy is to reduce network congestion within an enterprise by offloading security and content delivery from web servers and application servers

© Copyright IBM Corporation 2013

WebSphere Application Server Network Deployment includes the Edge components. The Edge components include a load balancer and a caching proxy. These components are optional and can be installed separately from WebSphere Application Server. The load balancer distributes incoming client requests across servers, balancing workload and providing high availability by routing around unavailable servers. The load balancer is typically used in front of your web servers to balance load across multiple web servers in a local or wide area network.

The caching proxy can be used to reduce network congestion by offloading security and content delivery from web servers and application servers. The caching proxy can satisfy subsequent requests for the same content by delivering it directly from the local cache, which is much quicker than retrieving it again from the content host, such as the application server. Cacheable content includes static web pages and JSP files with dynamically generated but infrequently changed fragments.

Slide 32

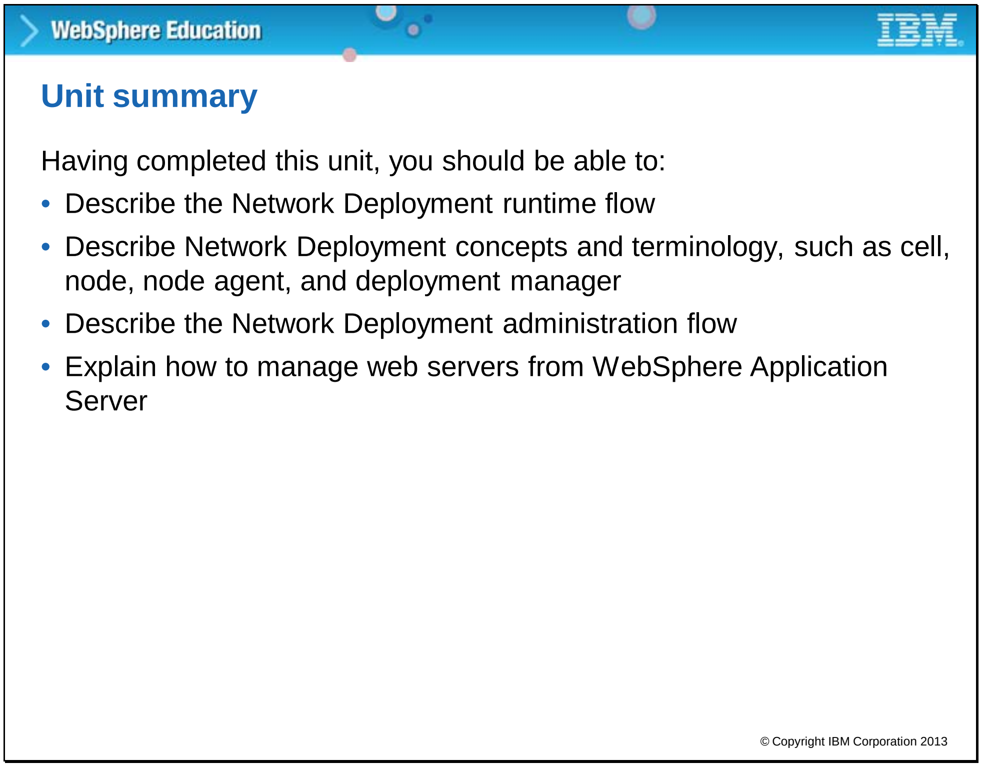


Intelligent management is a new group of capabilities that are integrated into WebSphere Application Server V8.5. These capabilities allow you to build virtualized application infrastructures to support the operation of your applications. Intelligent management provides a set of autonomic components that responds dynamically to the real-time conditions of the environment, adapting the infrastructure to respond to business needs. Intelligent management allows requests to be prioritized, and intelligent routing to respond to critical applications and users.

The Intelligent Management functions include the following key features:

- **Intelligent routing** can improve the quality of service by ensuring that business critical applications and users get priority. Requests to applications are prioritized and routed based on administrator-defined rules.
- **Health management** provides the ability for you to specify conditions that can automatically be detected and take corrective actions when these conditions are observed.
- **Application edition management** allows you to roll out new versions of applications without experiencing downtime for a maintenance window. Using this feature, you can validate a new edition of an application in your production environment without affecting users, and upgrade your applications without incurring outages to your users. You can also run multiple editions of a single application concurrently, directing different users to different editions.
- **Performance Management** provides a self-optimizing middleware infrastructure. By using dynamic clusters, you can automatically scale up or down the number of running instances of a cluster to meet the defined service policies. You can take advantage of an overload protection to limit the workload of a server instance, and prevent heap exhaustion, processor exhaustion, or both from occurring.

Slide 33



WebSphere Education
IBM

Unit summary

Having completed this unit, you should be able to:

- Describe the Network Deployment runtime flow
- Describe Network Deployment concepts and terminology, such as cell, node, node agent, and deployment manager
- Describe the Network Deployment administration flow
- Explain how to manage web servers from WebSphere Application Server

You completed this unit.
Having completed this unit, you should be able to:

- Describe the network deployment runtime flow
- Describe network deployment concepts and terminology such as cell, node, node agent, and deployment manager
- Describe the network deployment administration flow
- Explain how to manage web servers from WebSphere Application Server