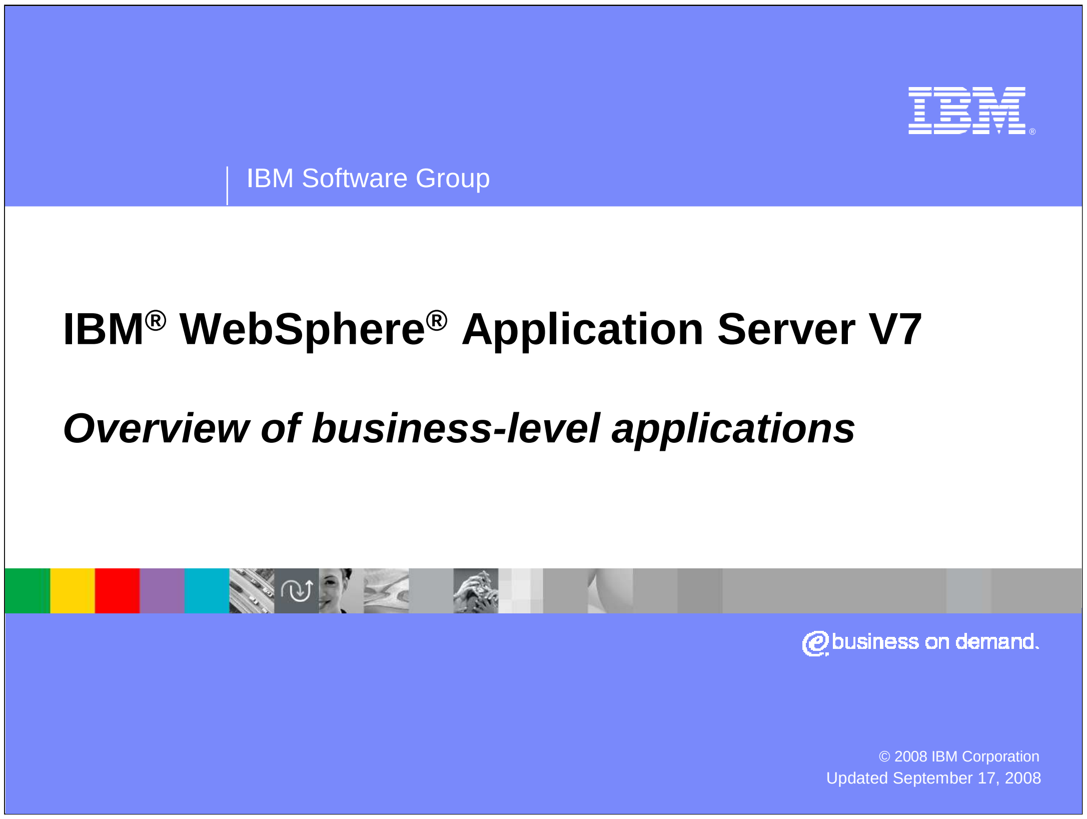


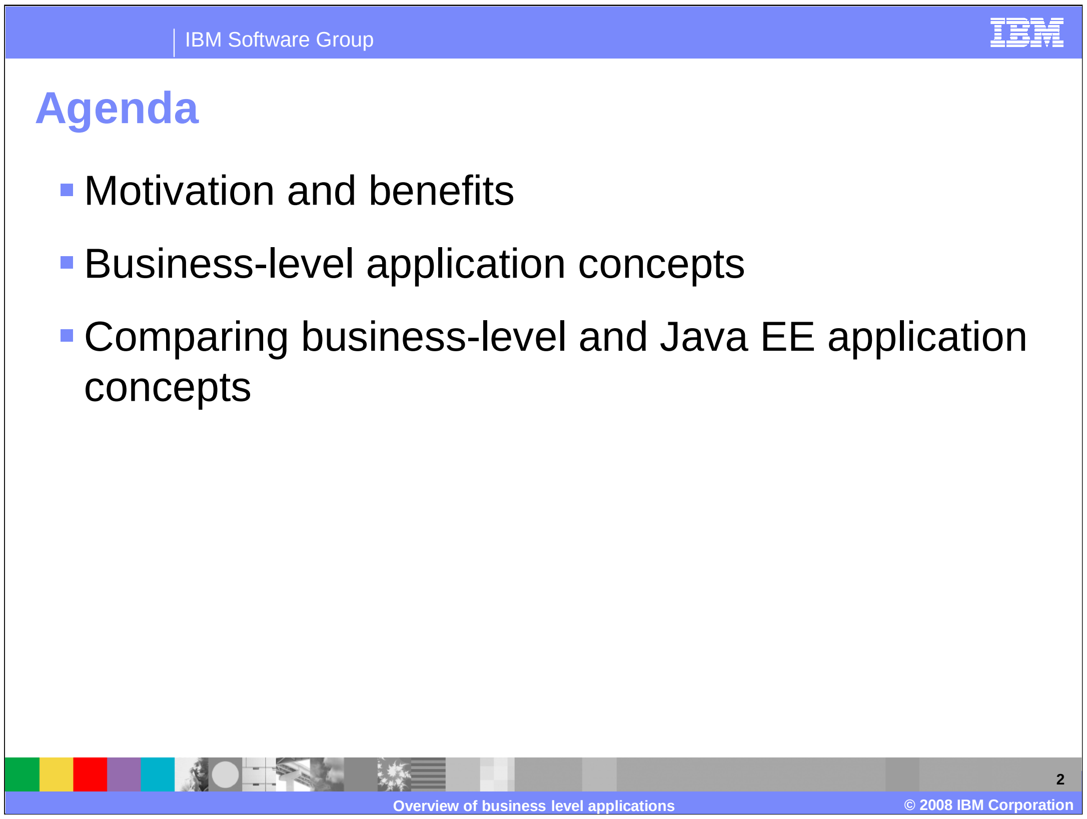
IBM® WebSphere® Application Server V7

Overview of business-level applications

business on demand.

Updated September 17, 2008

This presentation will introduce the concept of business-level applications in WebSphere Application Server V7.


IBM Software Group

Agenda

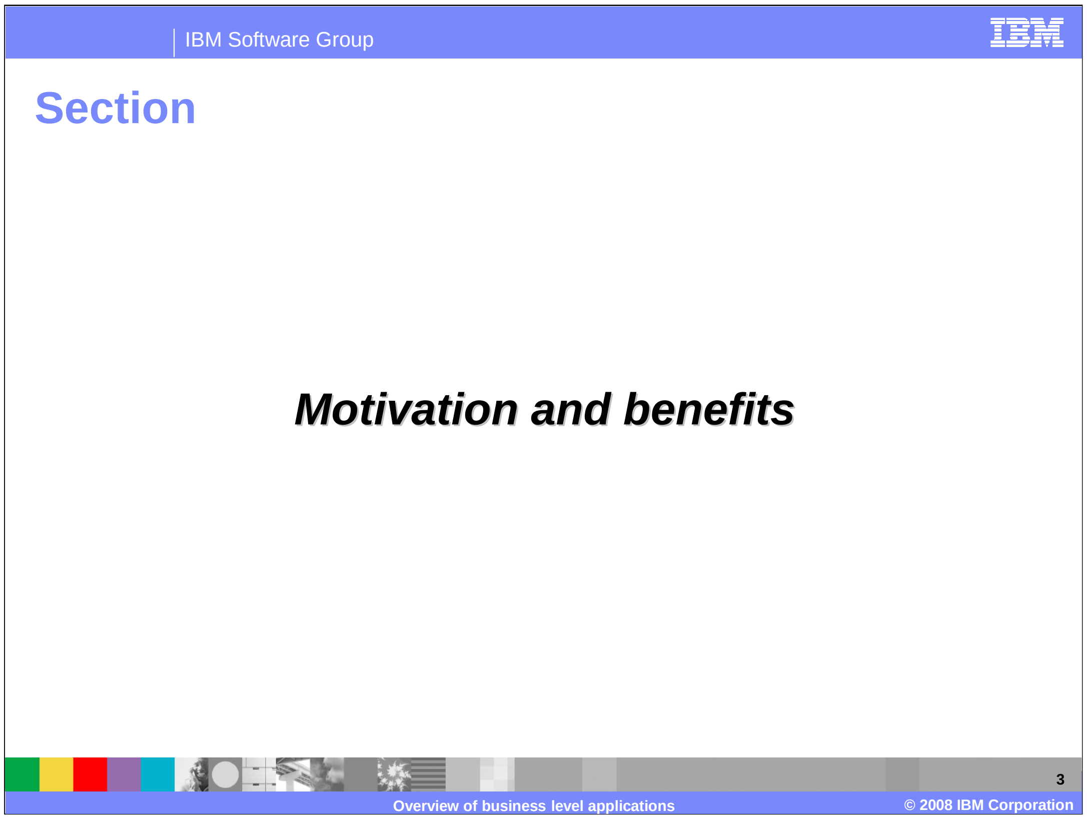
- Motivation and benefits
- Business-level application concepts
- Comparing business-level and Java EE application concepts

2

Overview of business level applications

© 2008 IBM Corporation

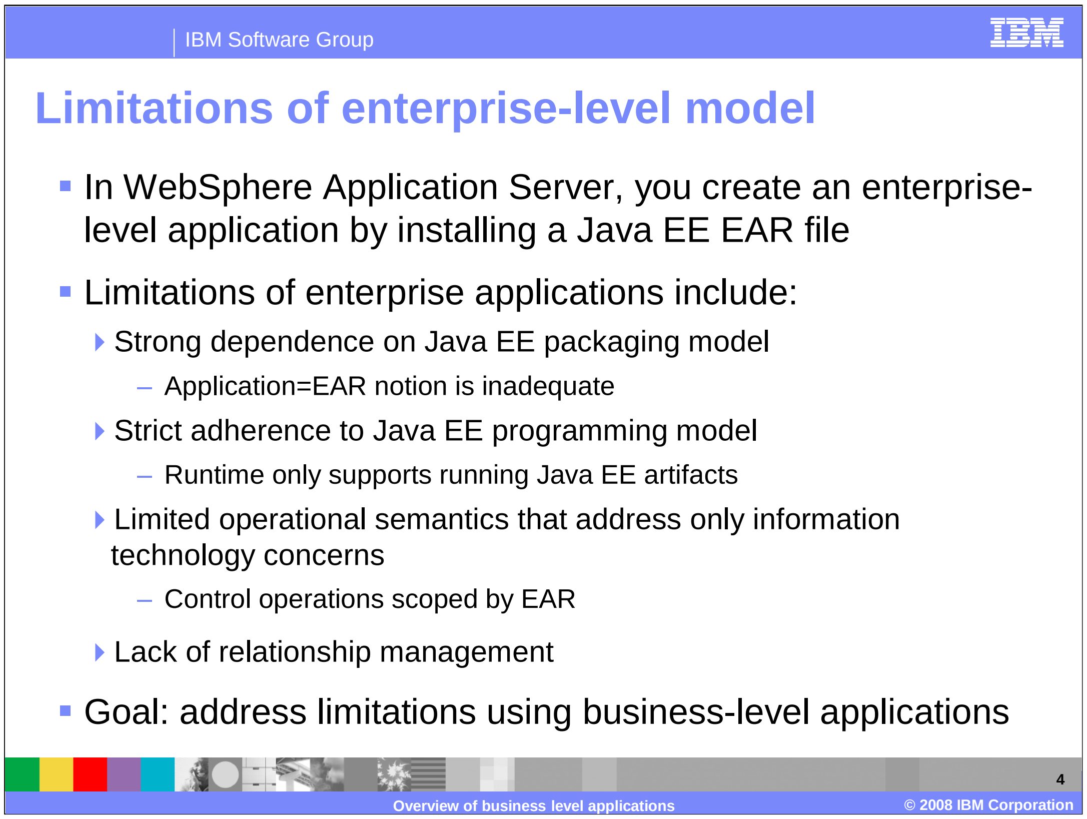
This presentation will first cover the motivation for and benefits of business-level applications, then introduce the key concepts, and compare those concepts to traditional Java EE application concepts.



Section

Motivation and benefits

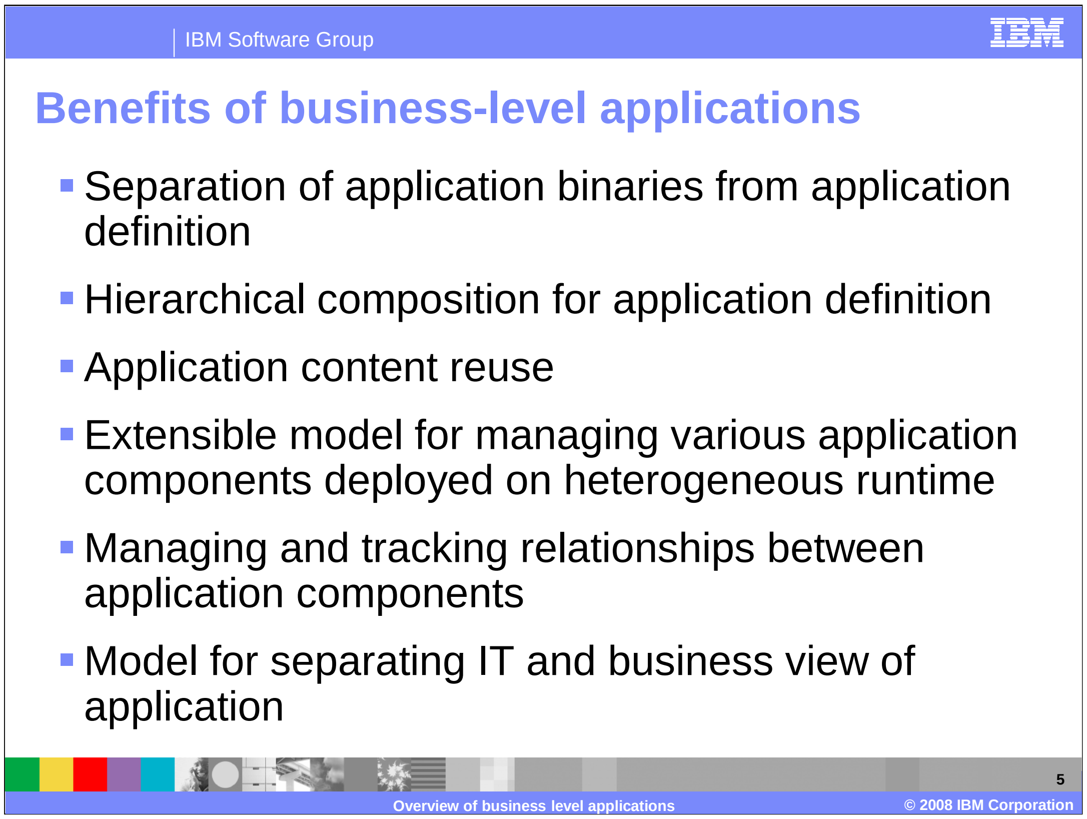
This section will discuss the motivation for and benefits of business-level applications.



Limitations of enterprise-level model

- In WebSphere Application Server, you create an enterprise-level application by installing a Java EE EAR file
- Limitations of enterprise applications include:
  - Strong dependence on Java EE packaging model
    - Application=EAR notion is inadequate
  - Strict adherence to Java EE programming model
    - Runtime only supports running Java EE artifacts
  - Limited operational semantics that address only information technology concerns
    - Control operations scoped by EAR
  - Lack of relationship management
- Goal: address limitations using business-level applications

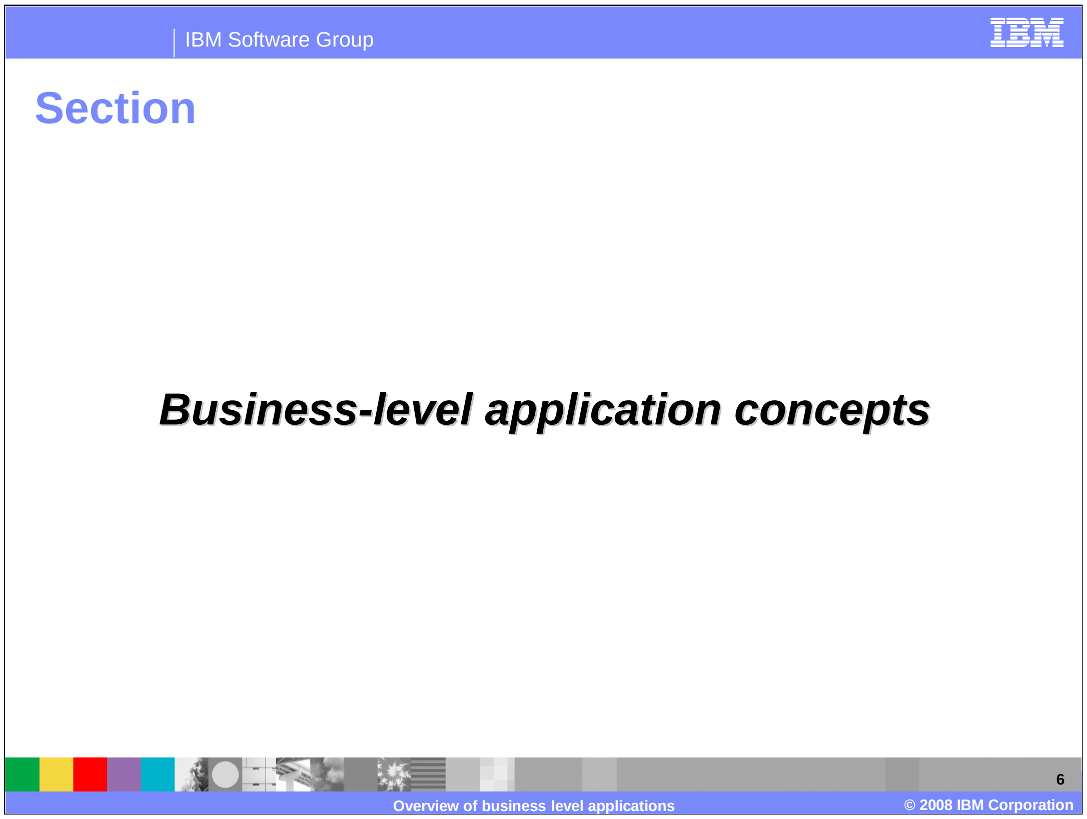
Traditionally, Java EE EAR files have been the only means of defining an enterprise application in WebSphere Application Server. This approach has some drawbacks, since the concept of an “application” might have a broader meaning to a line of business. WebSphere Application Server V7 addresses these concerns with “business-level applications”. A business-level application is an administrative construct that lets you group multiple artifacts, including EAR files, shared libraries, and other components as a single logical construct. This grouping denotes that the artifacts are related, and enables centralized operational control of the grouped resources.



Benefits of business-level applications

- Separation of application binaries from application definition
- Hierarchical composition for application definition
- Application content reuse
- Extensible model for managing various application components deployed on heterogeneous runtime
- Managing and tracking relationships between application components
- Model for separating IT and business view of application

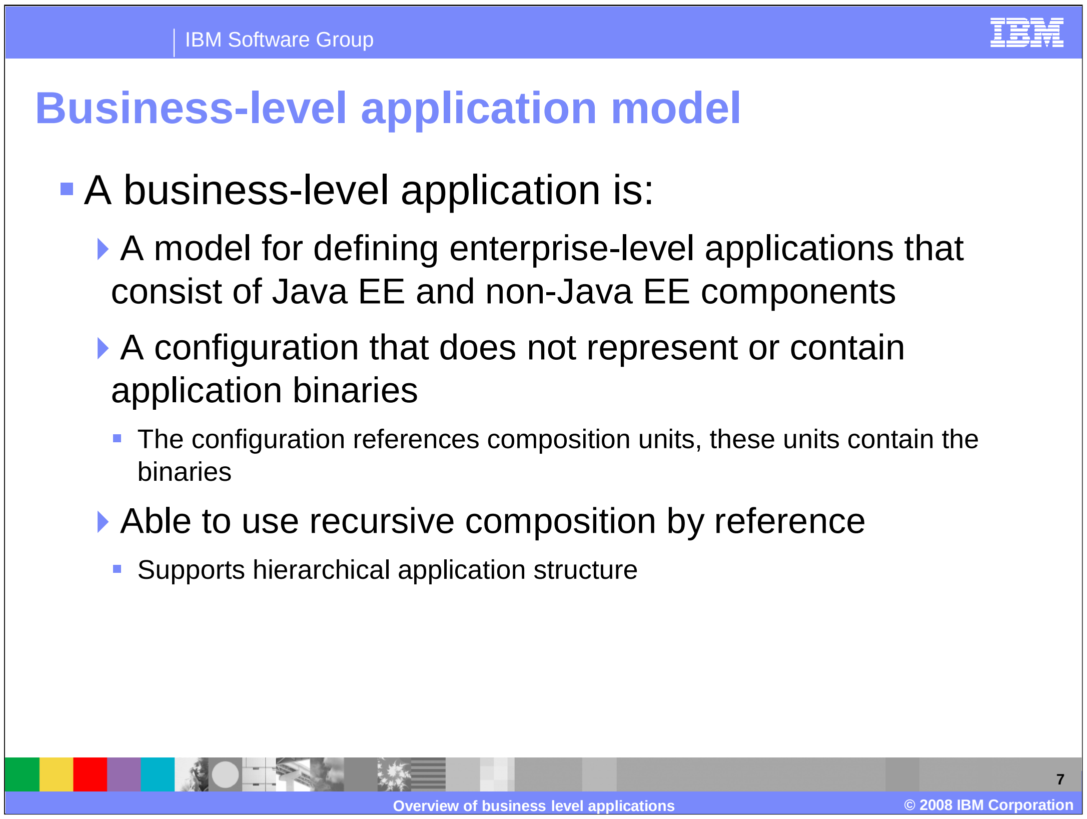
Business-level applications can be a useful way to define relationships between various application components. In addition to containing enterprise applications and shared resources, a business-level application can also contain other business-level applications, allowing you to define hierarchical relationships between applications.



Section

Business-level application concepts

This section will introduce new concepts related to business-level applications.


IBM Software Group

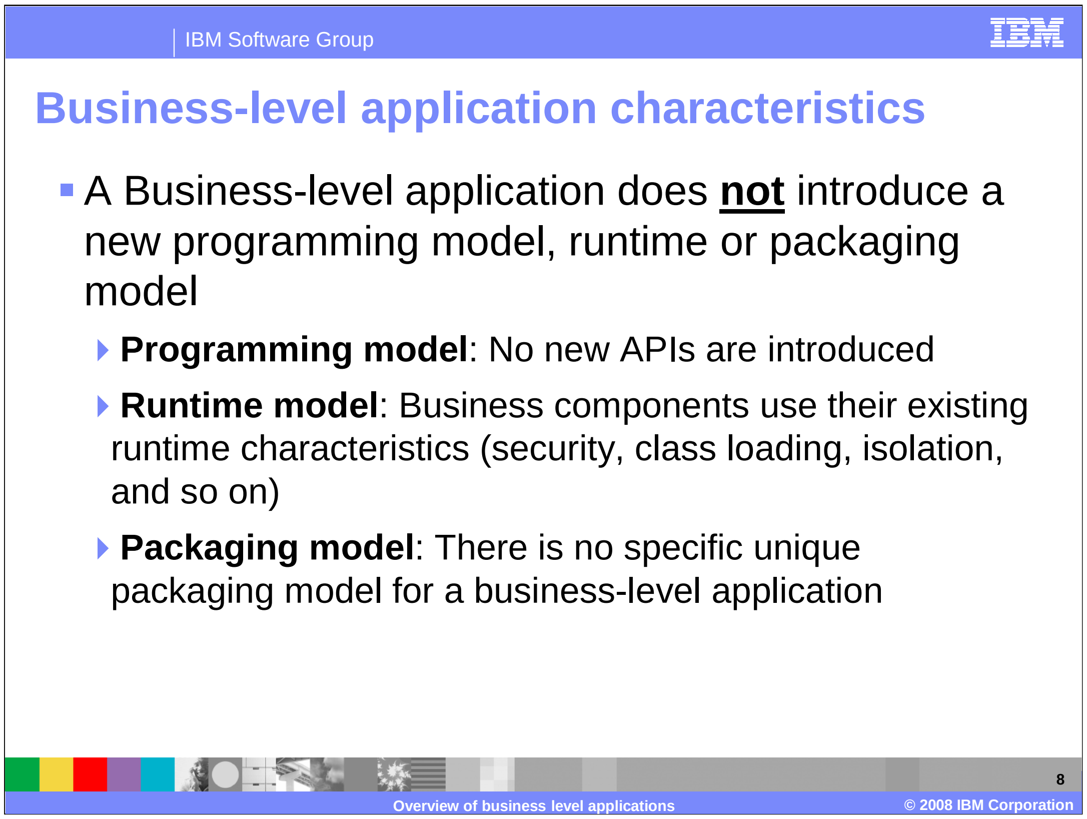
Business-level application model

- A business-level application is:
  - A model for defining enterprise-level applications that consist of Java EE and non-Java EE components
  - A configuration that does not represent or contain application binaries
    - The configuration references composition units, these units contain the binaries
  - Able to use recursive composition by reference
    - Supports hierarchical application structure

7

Overview of business level applications © 2008 IBM Corporation

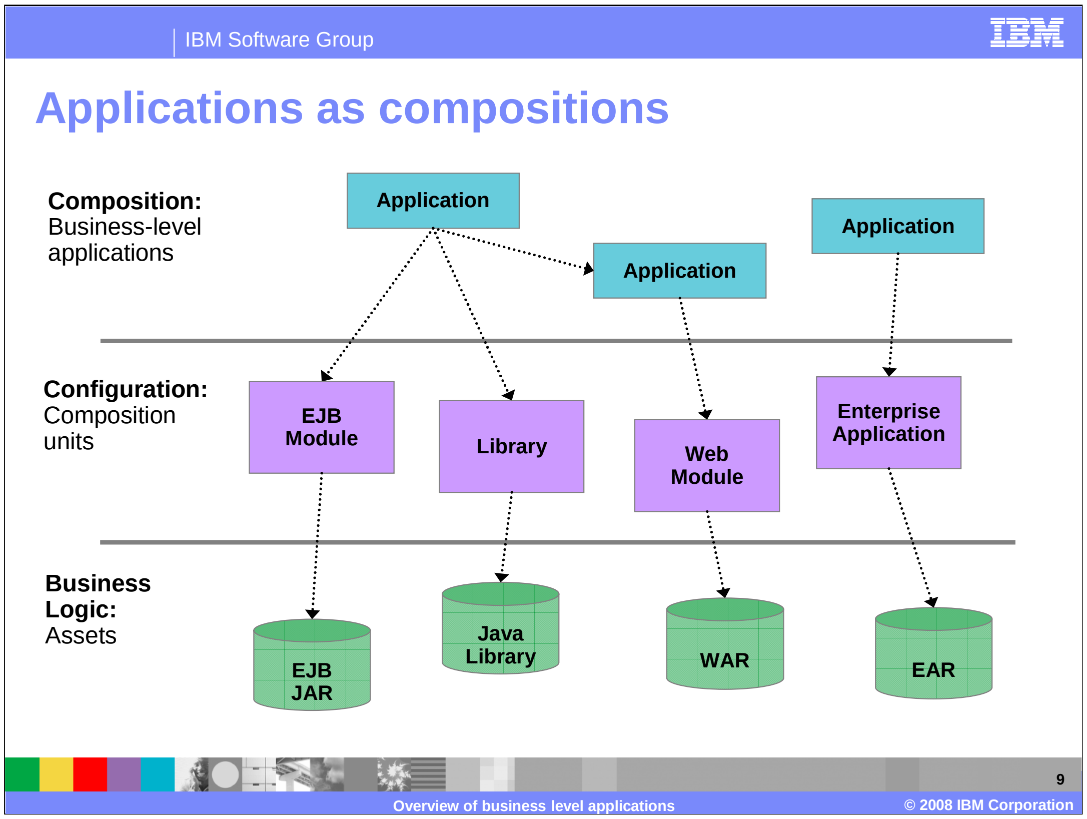
First, it is important to understand that a business-level application is a configuration model that defines relationships between artifacts, and is not a structure for packaging application binaries, like an EAR file is used for. A business-level application refers to one or more composition units, which can contain binary code, like EAR or JAR files.



Business-level application characteristics

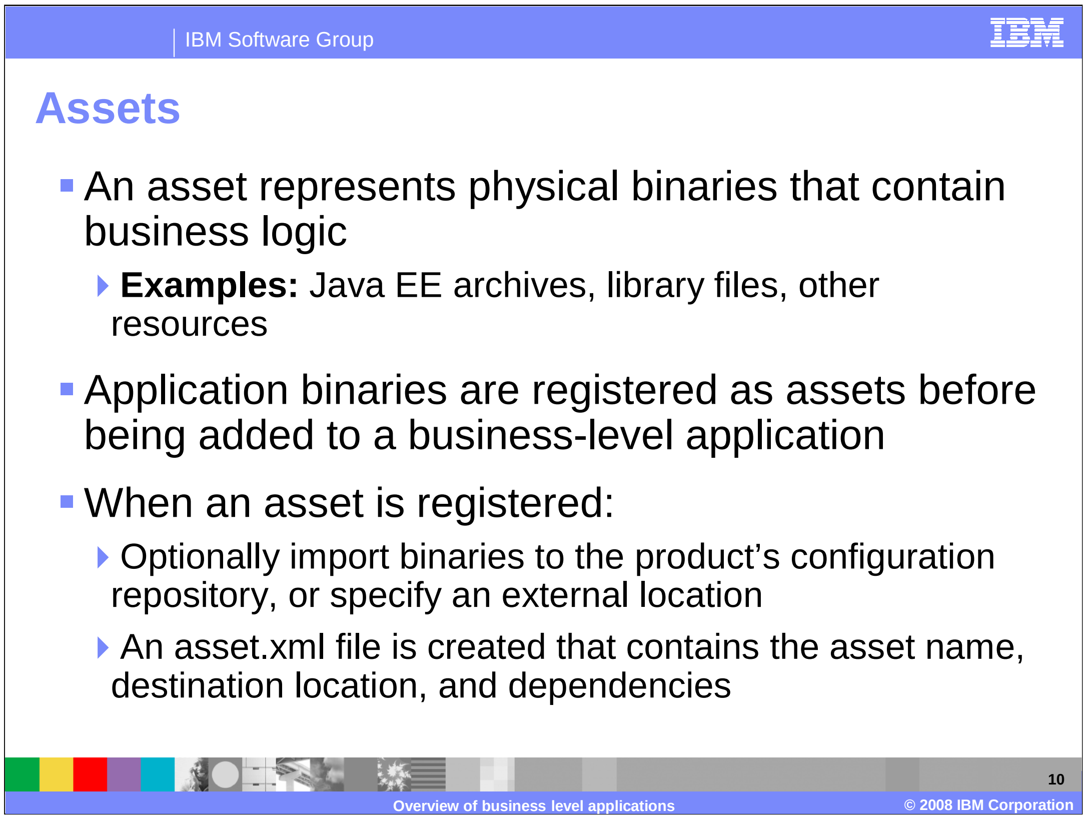
- A Business-level application does **not** introduce a new programming model, runtime or packaging model
  - **Programming model**: No new APIs are introduced
  - **Runtime model**: Business components use their existing runtime characteristics (security, class loading, isolation, and so on)
  - **Packaging model**: There is no specific unique packaging model for a business-level application

Being a model for administratively grouping, viewing, and operating related components, support for business-level applications does not introduce a new programming model, runtime framework, or packaging model. Applications are developed, deployed, and run as they would be otherwise. After enterprise applications and resources have been deployed, they can be grouped as business-level applications, so they can be viewed and controlled as a group. This grouping does not affect the runtime characteristics of the enterprise applications.



Applications as compositions

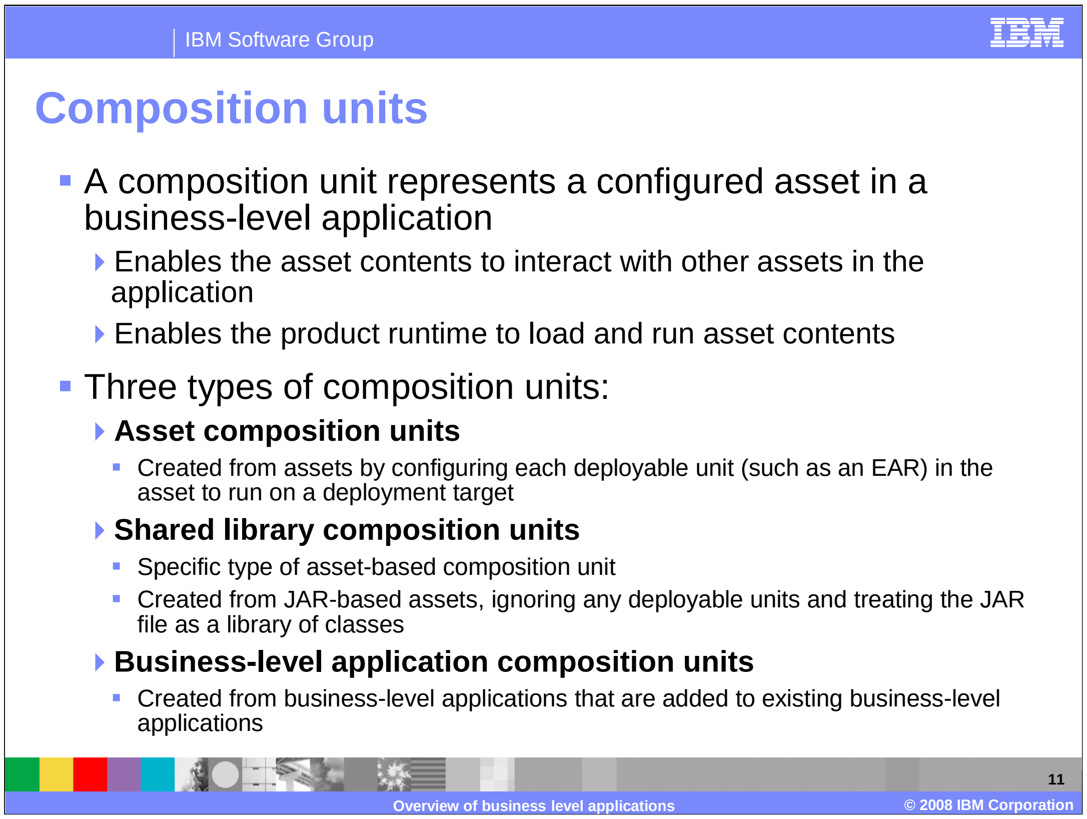
A business-level application is a composition made up of one or more “composition units”. These composition units contain configuration information about “assets”. Assets are the actual binary files that contain business logic, such as EAR or JAR files.



Assets

- An asset represents physical binaries that contain business logic
  - **Examples:** Java EE archives, library files, other resources
- Application binaries are registered as assets before being added to a business-level application
- When an asset is registered:
  - Optionally import binaries to the product's configuration repository, or specify an external location
  - An asset.xml file is created that contains the asset name, destination location, and dependencies

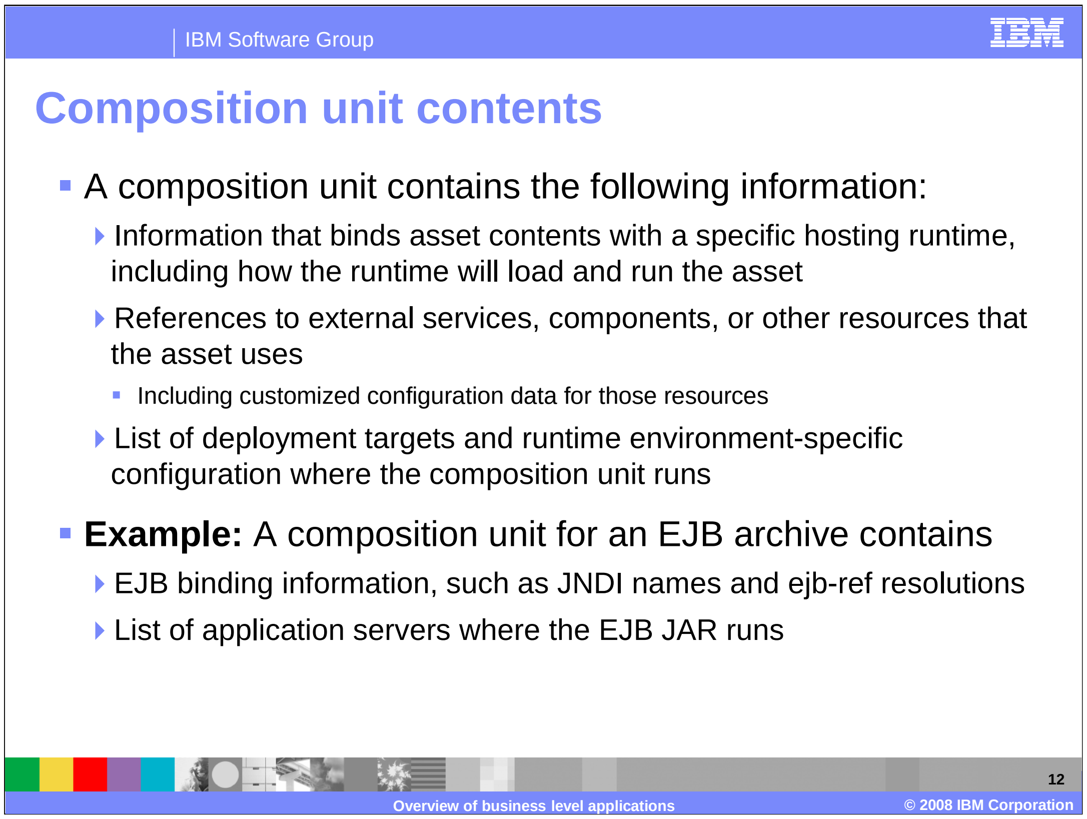
An asset represents the binary files that contain business logic, such as EAR, WAR, or JAR files, including shared libraries. Once a file has been deployed to WebSphere Application Server, through the traditional process, it can be registered as an asset, which will create a configuration file that points to the binary and lists any dependencies. This is a required step for any file before it can be added to a business-level application.



Composition units

- A composition unit represents a configured asset in a business-level application
  - Enables the asset contents to interact with other assets in the application
  - Enables the product runtime to load and run asset contents
- Three types of composition units:
  - **Asset composition units**
    - Created from assets by configuring each deployable unit (such as an EAR) in the asset to run on a deployment target
  - **Shared library composition units**
    - Specific type of asset-based composition unit
    - Created from JAR-based assets, ignoring any deployable units and treating the JAR file as a library of classes
  - **Business-level application composition units**
    - Created from business-level applications that are added to existing business-level applications

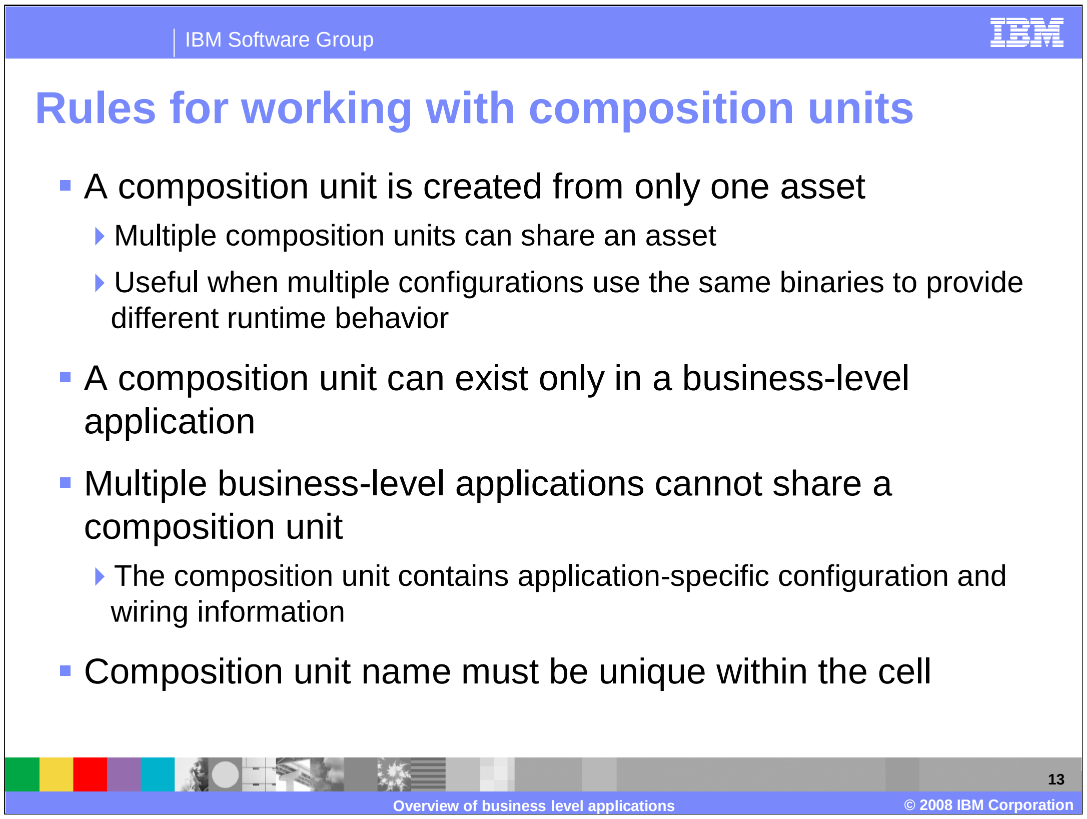
Business-level applications are represented by “composition units”, which represent assets and their associated configuration. You can create three types of composition units. An asset composition unit associates each deployable unit in the asset, such as a module, with a deployment target, which is a server or a cluster. A shared library composition unit defines a JAR file as a library. A business-level application composition unit defines a business-level application, so that you can add it to a different business-level application.



Composition unit contents

- A composition unit contains the following information:
  - Information that binds asset contents with a specific hosting runtime, including how the runtime will load and run the asset
  - References to external services, components, or other resources that the asset uses
    - Including customized configuration data for those resources
  - List of deployment targets and runtime environment-specific configuration where the composition unit runs
- **Example:** A composition unit for an EJB archive contains
  - EJB binding information, such as JNDI names and ejb-ref resolutions
  - List of application servers where the EJB JAR runs

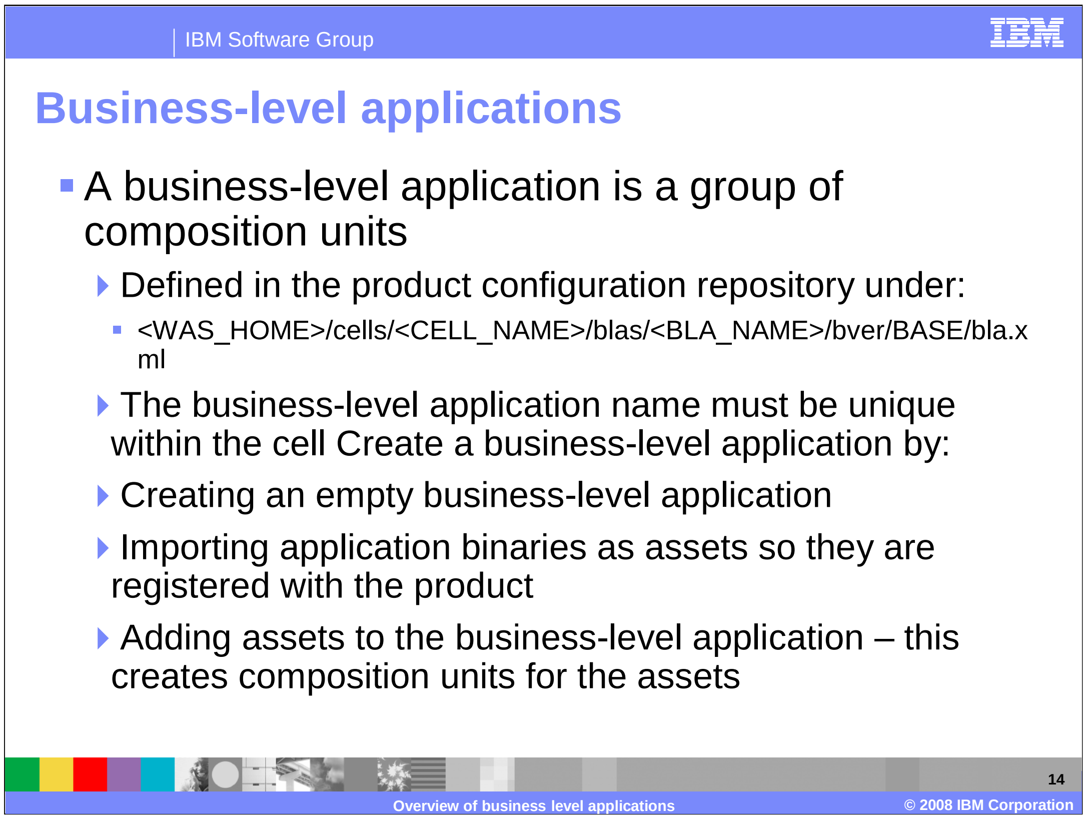
As an example of what a composition unit might contain, a composition unit for an EJB-JAR would contain a list of servers or clusters where the EJB module will run, along with references to any external dependencies, like JNDI names for EJB bindings.



Rules for working with composition units

- A composition unit is created from only one asset
  - Multiple composition units can share an asset
  - Useful when multiple configurations use the same binaries to provide different runtime behavior
- A composition unit can exist only in a business-level application
- Multiple business-level applications cannot share a composition unit
  - The composition unit contains application-specific configuration and wiring information
- Composition unit name must be unique within the cell

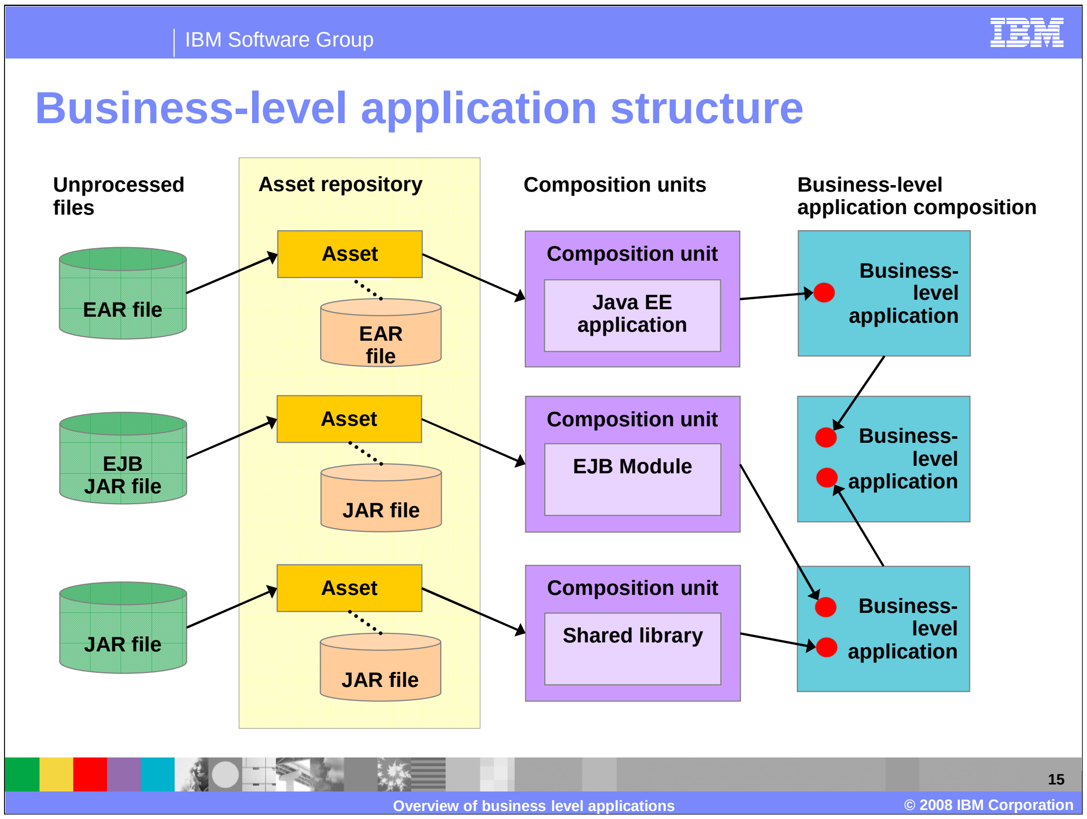
A composition unit always includes one and only one asset, though multiple composition units can share an asset, enabling you to create two separate runtime configurations based on the same binary files. Because composition units contain application-specific information, like bindings, they cannot be shared by multiple business-level applications.



Business-level applications

- A business-level application is a group of composition units
  - Defined in the product configuration repository under:
    - <WAS_HOME>/cells/<CELL_NAME>/blas/<BLA_NAME>/bver/BASE/bla.xml
  - The business-level application name must be unique within the cell Create a business-level application by:
  - Creating an empty business-level application
  - Importing application binaries as assets so they are registered with the product
  - Adding assets to the business-level application – this creates composition units for the assets

To create a business level application, use the administrative tools (such as the administrative console or wsadmin) to create a new business-level application. This will create configuration data for the business-level application in your configuration repository. You can then add assets to the business-level application, which will create composition units for each asset, and associate them with your business-level application.

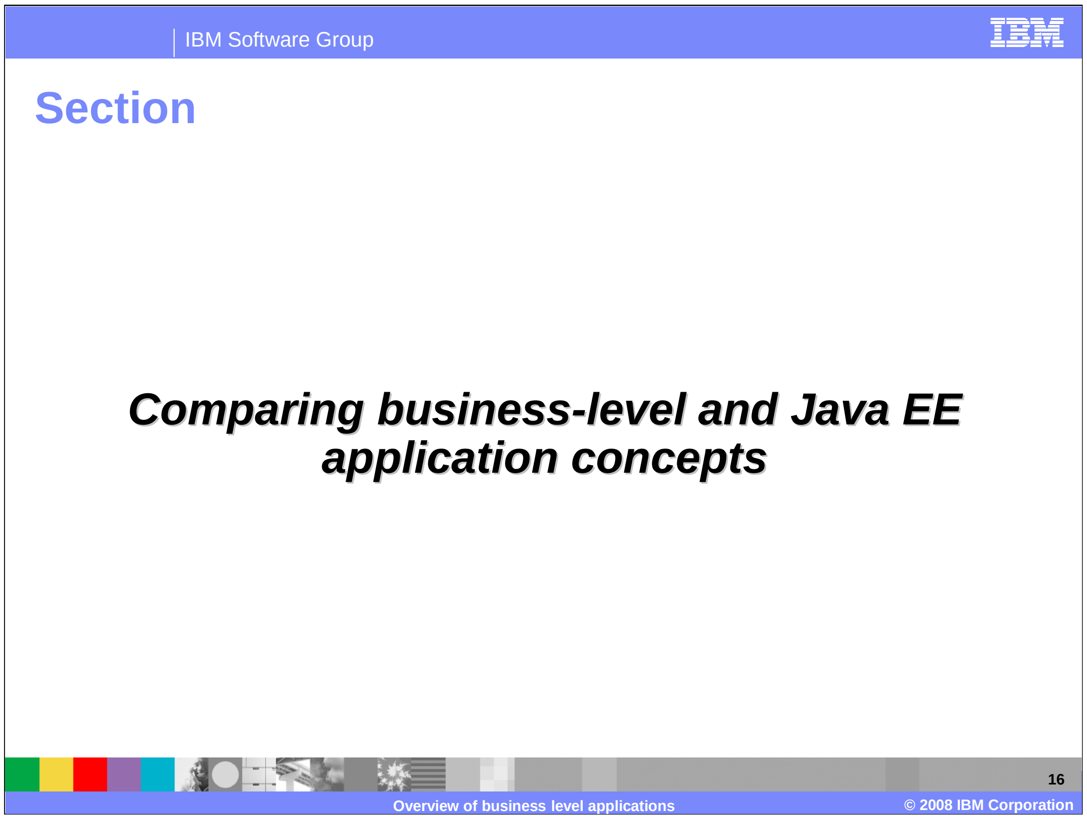

IBM Software Group

Business-level application structure

15

Overview of business level applications © 2008 IBM Corporation

This diagram illustrates how the components of business-level applications fit together. On the left, you start with unprocessed application resources, like an EAR or a JAR file, and import them as assets. When you import the assets, you have the option of copying them into the application server's asset repository. When an asset is added to a business-level application, it gets wrapped up as a composition unit, which contains specific configuration information for the asset. Not only can business-level applications contain composition units based on assets, but they can also contain other business-level applications.



IBM Software Group

Section

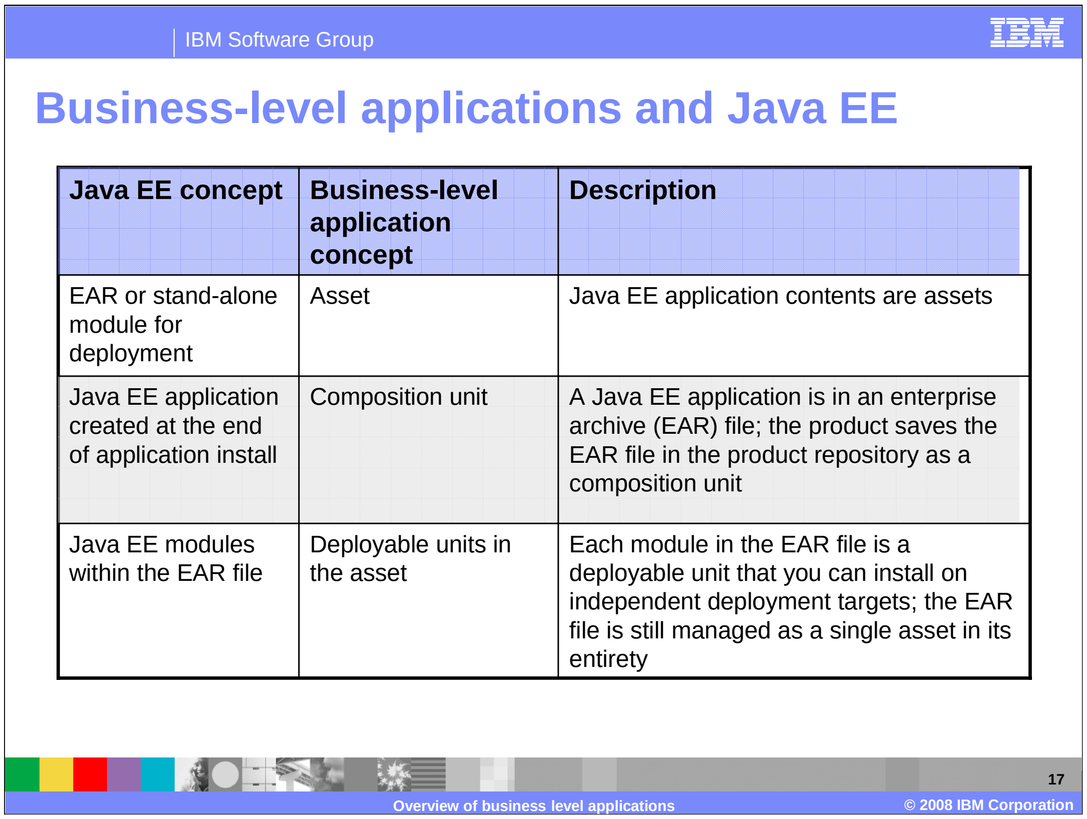
Comparing business-level and Java EE application concepts

16

Overview of business level applications

© 2008 IBM Corporation

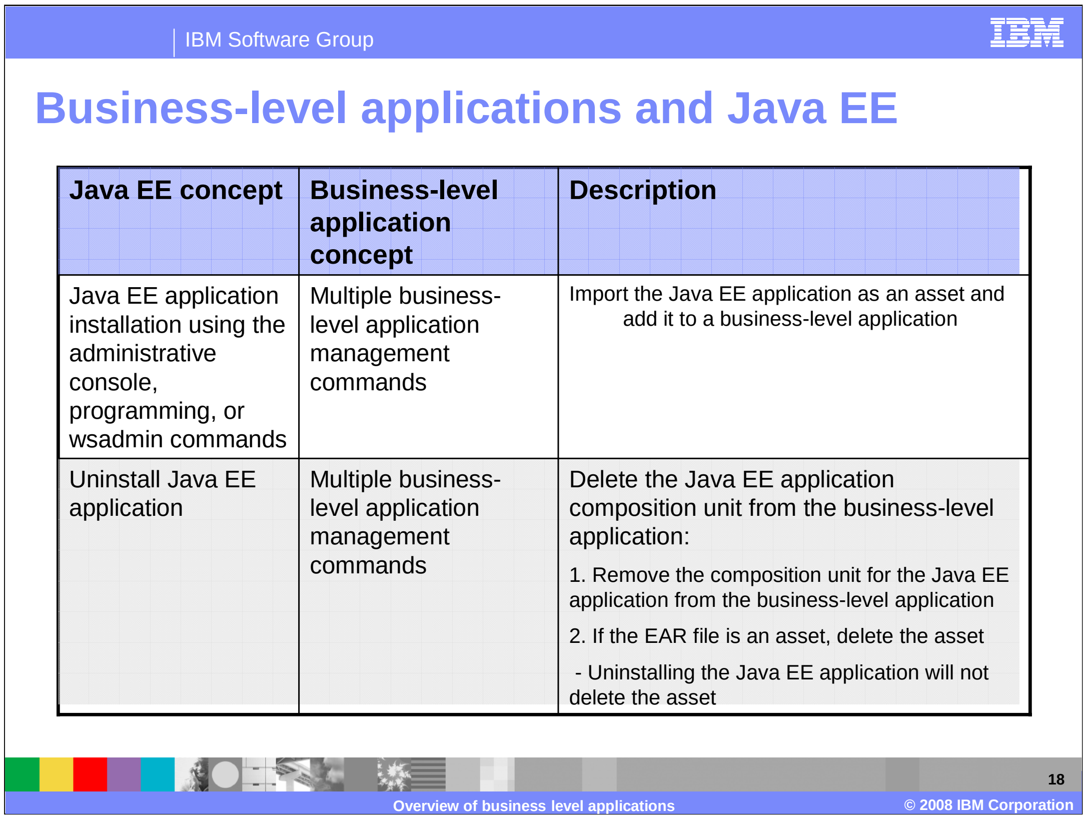
This section will compare business-level application concepts with the traditional Java EE application concepts.



Business-level applications and Java EE

Java EE concept	Business-level application concept	Description
EAR or stand-alone module for deployment	Asset	Java EE application contents are assets
Java EE application created at the end of application install	Composition unit	A Java EE application is in an enterprise archive (EAR) file; the product saves the EAR file in the product repository as a composition unit
Java EE modules within the EAR file	Deployable units in the asset	Each module in the EAR file is a deployable unit that you can install on independent deployment targets; the EAR file is still managed as a single asset in its entirety

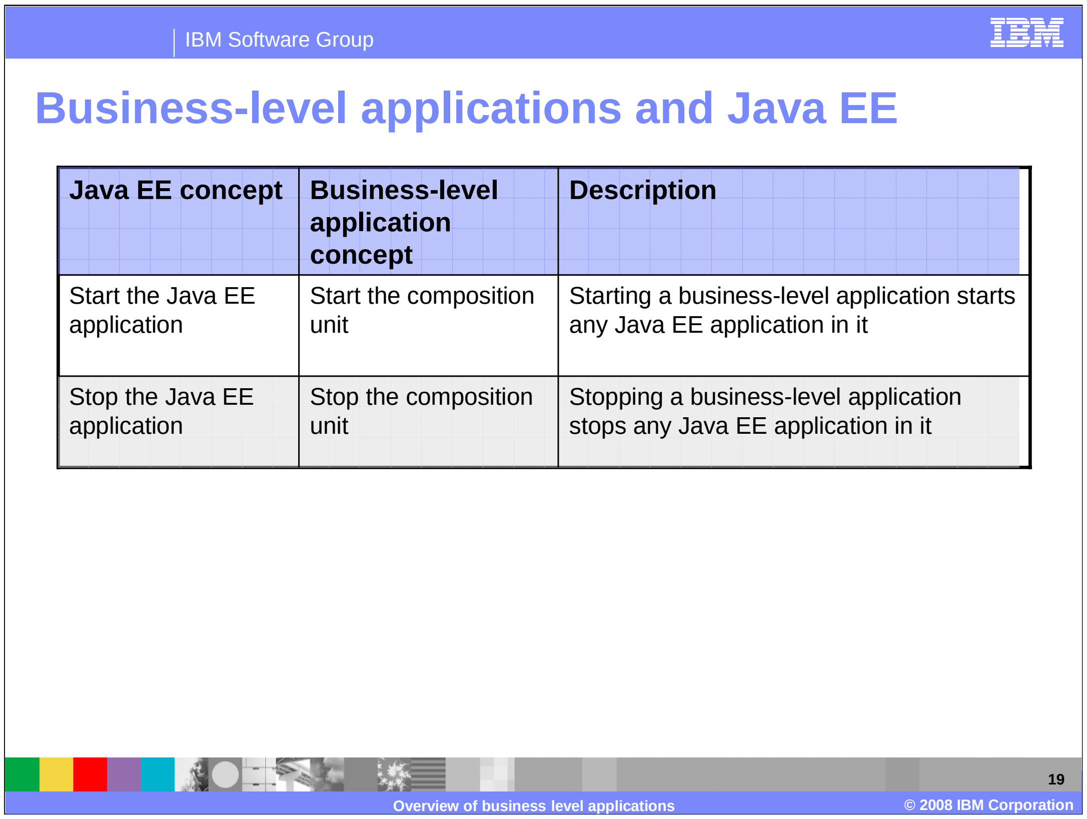
A Java EE archive on the file system, such as an EAR or JAR, is the equivalent of an asset. When you deploy an enterprise application to WebSphere Application Server, configuration items, such as bindings information, get created in the configuration repository. This is the rough equivalent of adding an asset to a business-level application, which creates a composition unit, containing similar information. Individual Java EE modules, which are contained within the deployed application, are represented as “deployable units” in a business-level application.



Business-level applications and Java EE

Java EE concept	Business-level application concept	Description
Java EE application installation using the administrative console, programming, or wsadmin commands	Multiple business-level application management commands	Import the Java EE application as an asset and add it to a business-level application
Uninstall Java EE application	Multiple business-level application management commands	Delete the Java EE application composition unit from the business-level application: 1. Remove the composition unit for the Java EE application from the business-level application 2. If the EAR file is an asset, delete the asset - Uninstalling the Java EE application will not delete the asset

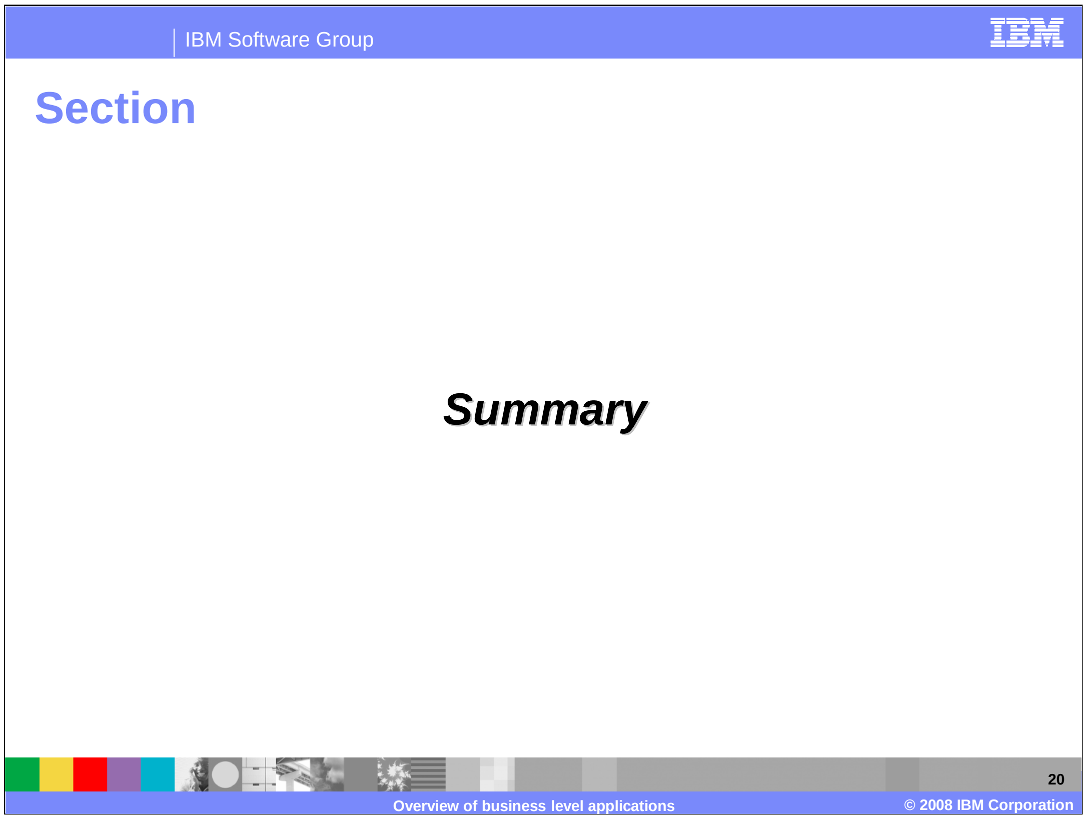
When Java EE applications are installed as assets in a business-level application, you interact with them through different management capabilities of the administrative console or wsadmin. To add a Java EE application to a business-level application, you import it as an asset, then add it to a business-level application. To remove a Java EE application from a business-level application, remove the composition unit from the business-level application. If no other business-level applications are using the EAR file, you can also delete the asset.



Business-level applications and Java EE

Java EE concept	Business-level application concept	Description
Start the Java EE application	Start the composition unit	Starting a business-level application starts any Java EE application in it
Stop the Java EE application	Stop the composition unit	Stopping a business-level application stops any Java EE application in it

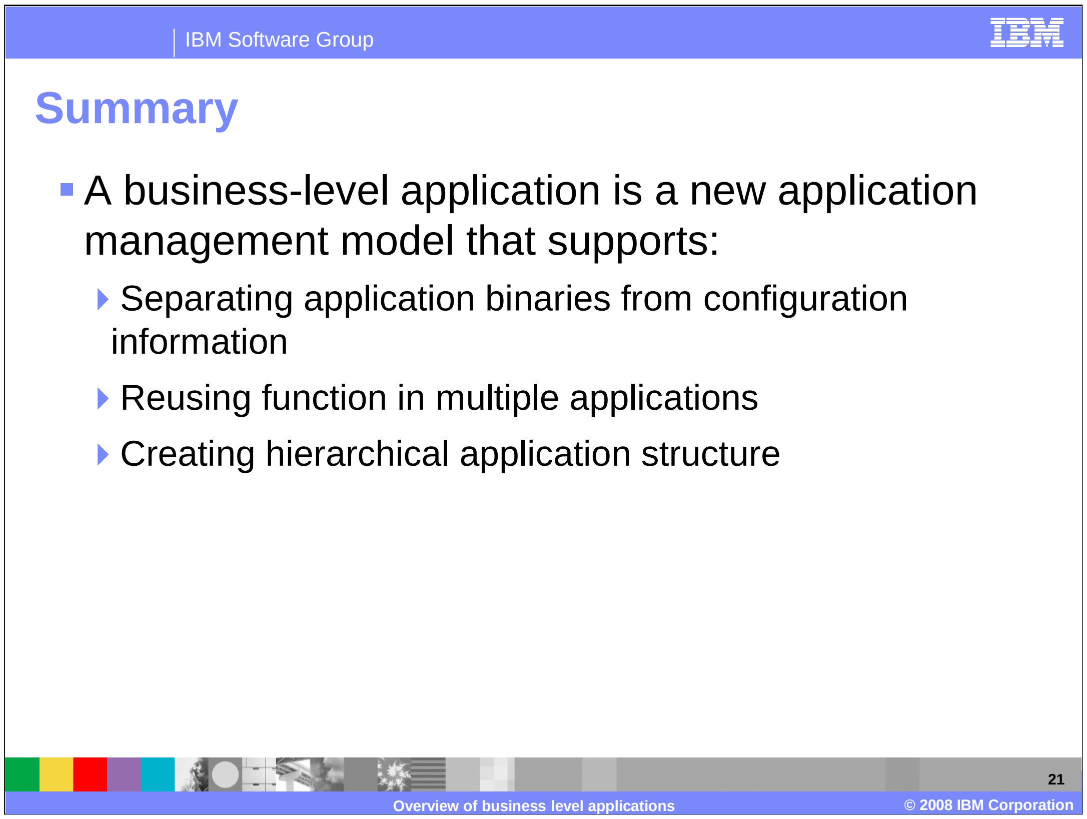
Starting and stopping a business-level application, starts or stops any Java EE application that is defined as part of the business-level application.



Section

Summary

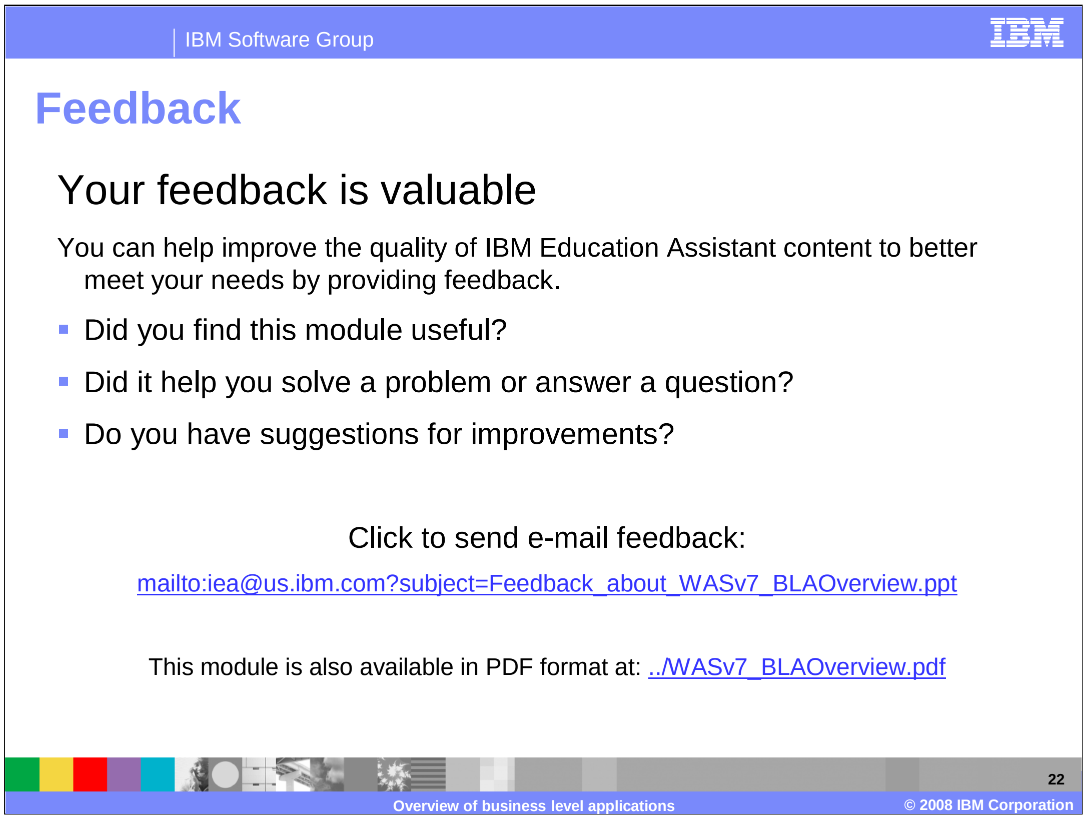
This section will summarize the key points of the presentation.



Summary

- A business-level application is a new application management model that supports:
  - Separating application binaries from configuration information
  - Reusing function in multiple applications
  - Creating hierarchical application structure

Business-level applications are part of a new application management model introduced in WebSphere Application Server V7. This model introduces a layer of abstraction that separates application binaries from configuration information, and allows you to easily reuse functionality in multiple applications. A few key terms were introduced: assets represent binary resources, like EAR files or shared libraries. A composition unit contains the application-specific configuration information for an asset. A business-level application is the configuration construct that contains multiple composition units, giving you a single point of control for all of the contained assets.



Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

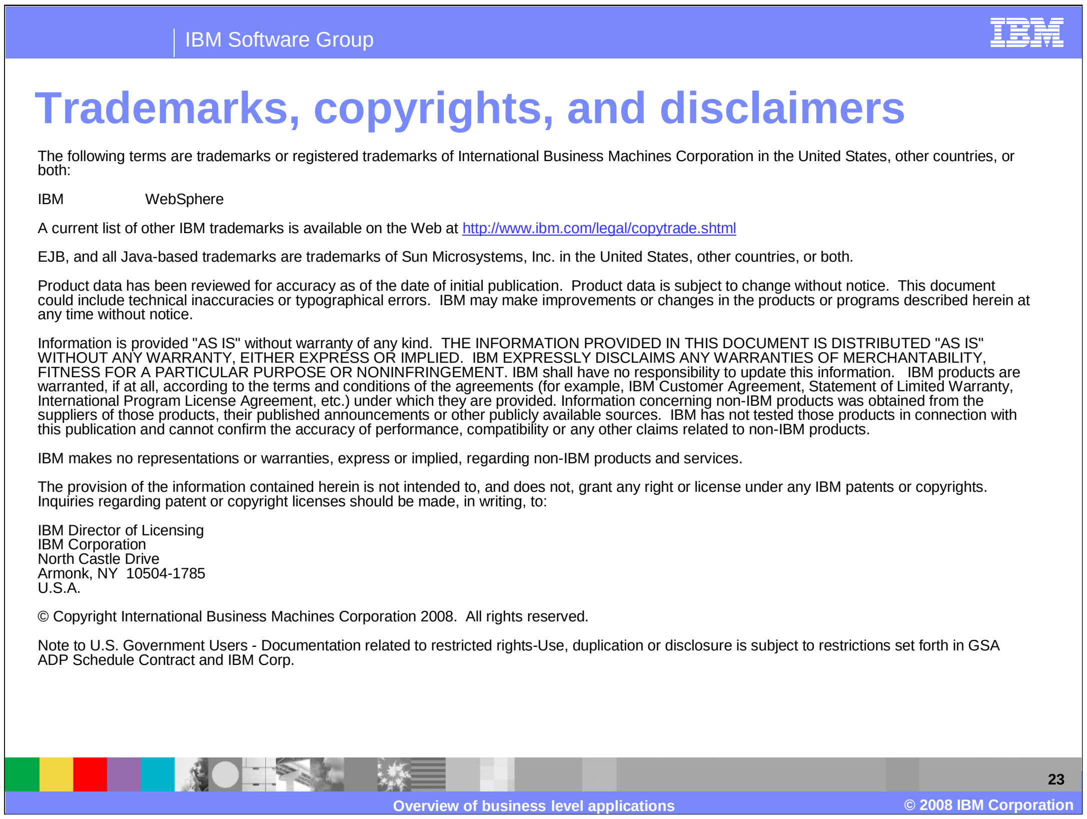
- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WASv7_BLAOverview.ppt

This module is also available in PDF format at: ../WASv7_BLAOverview.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.



Trademarks, copyrights, and disclaimers