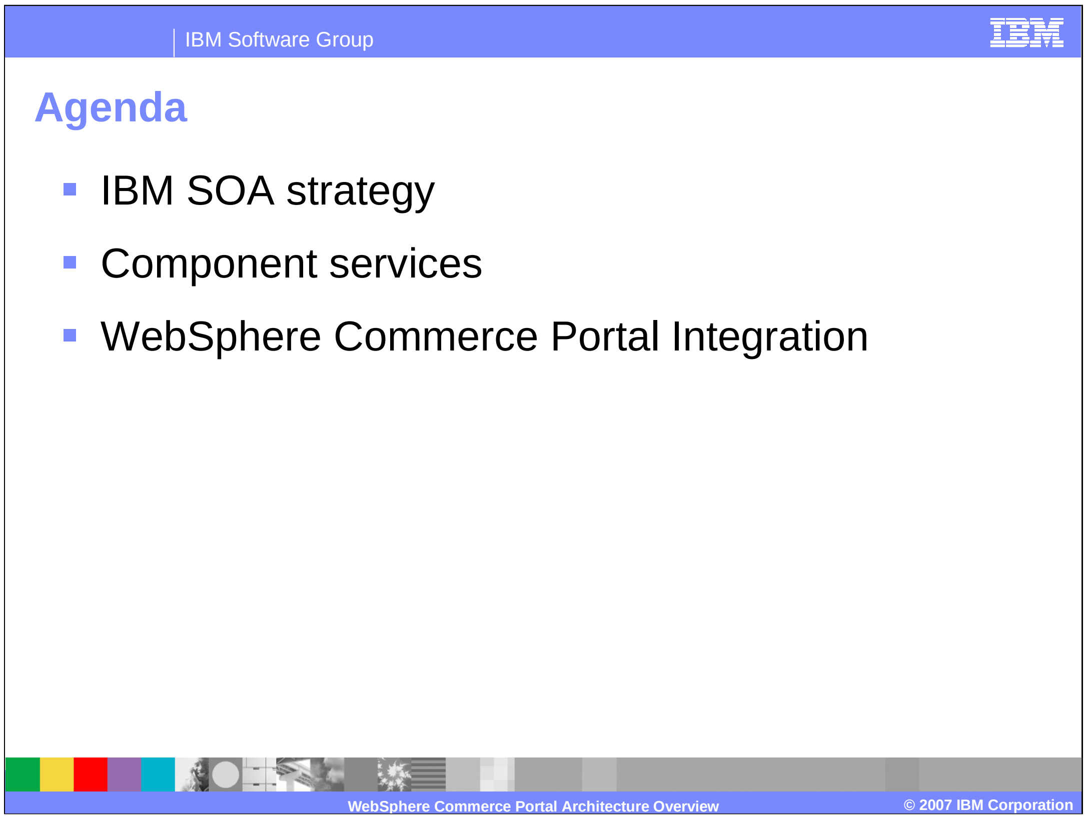
IBM Software Group

WebSphere® Commerce Feature Pack 2

WebSphere Commerce Portal Architecture Overview

business on demand.

Welcome to the WebSphere Commerce Feature Pack 2, WebSphere Commerce Portal Architecture overview presentation.



Agenda

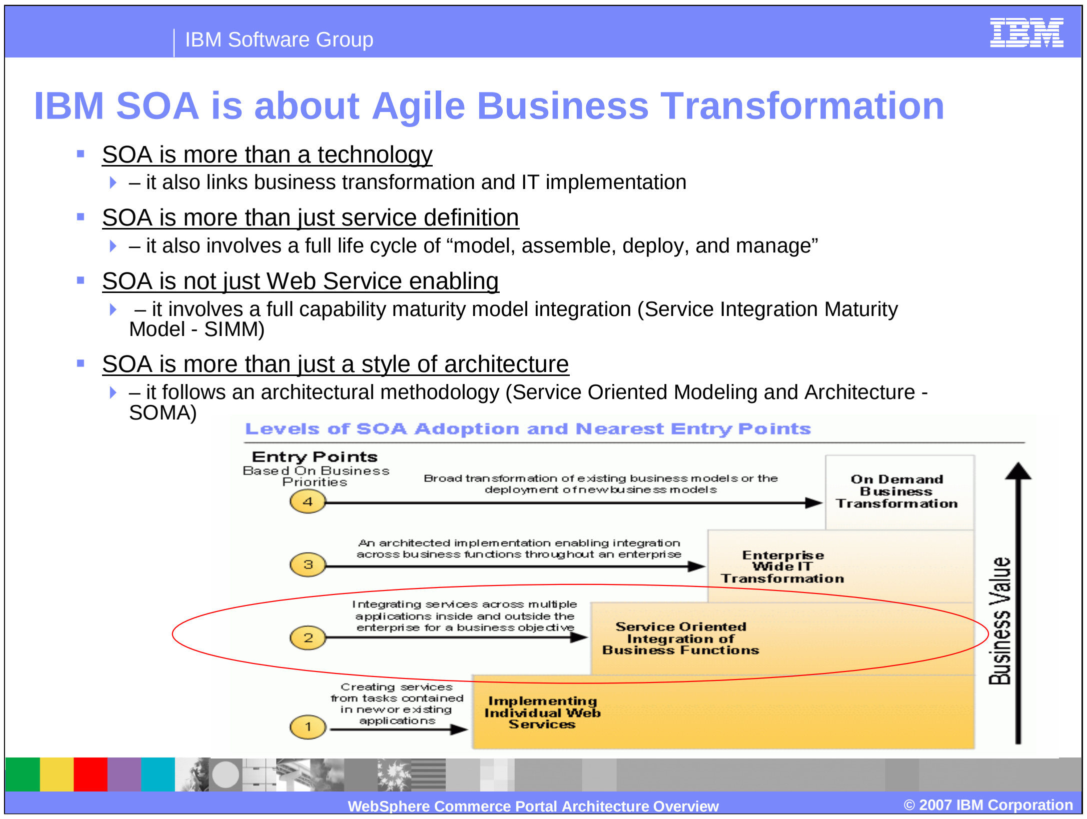
- IBM SOA strategy
- Component services
- WebSphere Commerce Portal Integration

This presentation covers the following three topics;

- the IBM SOA Strategy and how WebSphere Commerce fits within this strategy,

- Business Component services

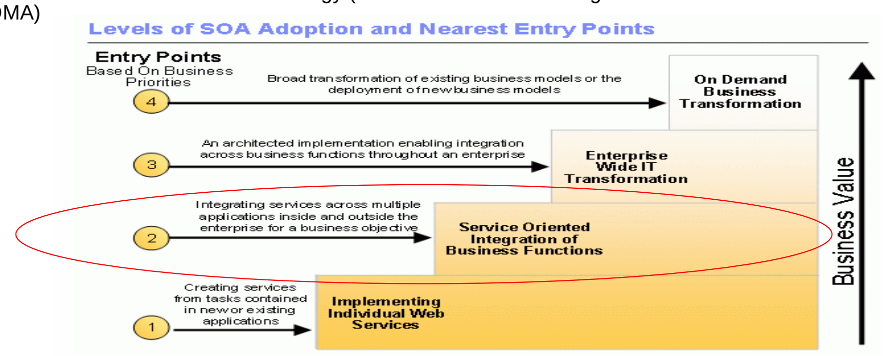
- and the integration efforts between WebSphere Commerce and WebSphere Portal.



IBM Software Group

IBM SOA is about Agile Business Transformation

- SOA is more than a technology
  - – it also links business transformation and IT implementation
- SOA is more than just service definition
  - – it also involves a full life cycle of "model, assemble, deploy, and manage"
- SOA is not just Web Service enabling
  - – it involves a full capability maturity model integration (Service Integration Maturity Model - SIMM)
- SOA is more than just a style of architecture
  - – it follows an architectural methodology (Service Oriented Modeling and Architecture - SOMA)


Levels of SOA Adoption and Nearest Entry Points
Entry Points
Based On Business Priorities
4
Broad transformation of existing business models or the deployment of new business models
On Demand Business Transformation
3
An architected implementation enabling integration across business functions throughout an enterprise
Enterprise Wide IT Transformation
2
Integrating services across multiple applications inside and outside the enterprise for a business objective
Service Oriented Integration of Business Functions
1
Creating services from tasks contained in new or existing applications
Implementing Individual Web Services
Business Value

WebSphere Commerce Portal Architecture Overview © 2007 IBM Corporation

Let's start with the question What is SOA?

IBM describes Service-Oriented Architecture as a process, life cycle and a set of tools to enable agile business transformation.

It is this particular practice of deriving the IT implementation of the system from the business requirements that is known as IBM SOA.

In industry, there are many instances where SOA is equated with Web services.

The IBM perspective on SOA is not solely about technology; so, there is no SOA platform that one deploys their solution on.

Instead, IBM SOA defines an architecture which allows clients to build a solution that has a tight coupling between the business requirements and the IT implementation.

Changes in one, can be quickly reflected in the other, allowing for agile business transformation.

In IBM SOA there is a life cycle defined which flows from gathering the business requirements to maintaining the system.

This life cycle guides one in building an SOA architecture. This life cycle begins with the business analyst producing a business design that captures the business processes of the organization.

The business analyst then works with the IT architect, to model the appropriate artifacts.

These models are then transformed in to IT assets that are then assembled.

The assets are subsequently deployed, and then the day-to-day operations are monitored and managed.

This simple flow is what is known as the IBM SOA life cycle of *Model*, *Assemble*, *Deploy*, and *Manage*.

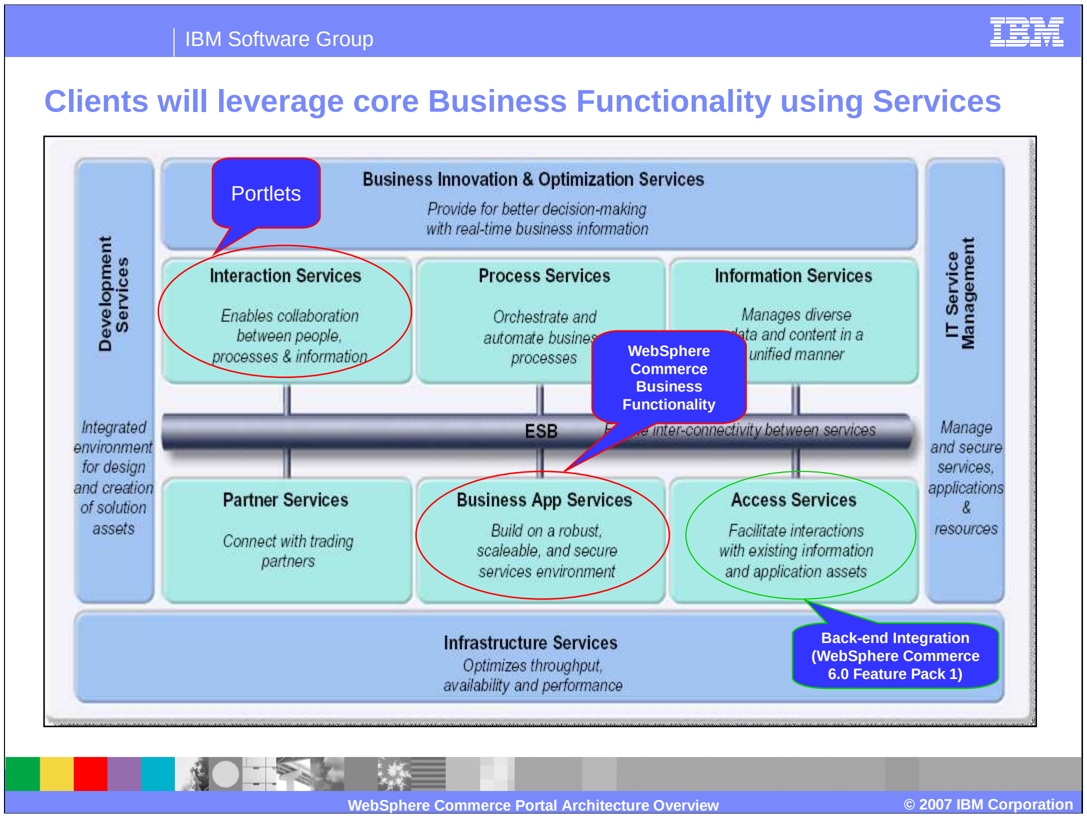
Tools are provided to aid in activities performed at each stage of the life cycle; these tools are known as the IBM SOA Foundation tools.

A key concept in SOA is that of *Services*. *Services* are repeatable tasks found within business processes.

Service-orientation is a way of integrating your business as a set of linked services.

A *Service-Oriented Architecture*, then, exploits the principles of service-orientation to achieve a tighter relationship between the business and the information systems that support the business.

Implementing IBM SOA can help your company realize the following benefits: greater alignment of business and IT, loosely coupled components and systems, a network-based infrastructure, enabling geographically and technologically diverse resources to work together, greater code reuse and better process standardization throughout the enterprise.



IBM Software Group

Clients will leverage core Business Functionality using Services

Portlets
Business Innovation & Optimization Services
Provide for better decision-making with real-time business information
Development Services
Integrated environment for design and creation of solution assets
Interaction Services
Enables collaboration between people, processes & information
Process Services
Orchestrate and automate business processes
Information Services
Manages diverse data and content in a unified manner
WebSphere Commerce Business Functionality
IT Service Management
Manage and secure services, applications & resources
ESB
Facilitate inter-connectivity between services
Partner Services
Connect with trading partners
Business App Services
Build on a robust, scaleable, and secure services environment
Access Services
Facilitate interactions with existing information and application assets
Infrastructure Services
Optimizes throughput, availability and performance
Back-end Integration (WebSphere Commerce 6.0 Feature Pack 1)

WebSphere Commerce Portal Architecture Overview © 2007 IBM Corporation

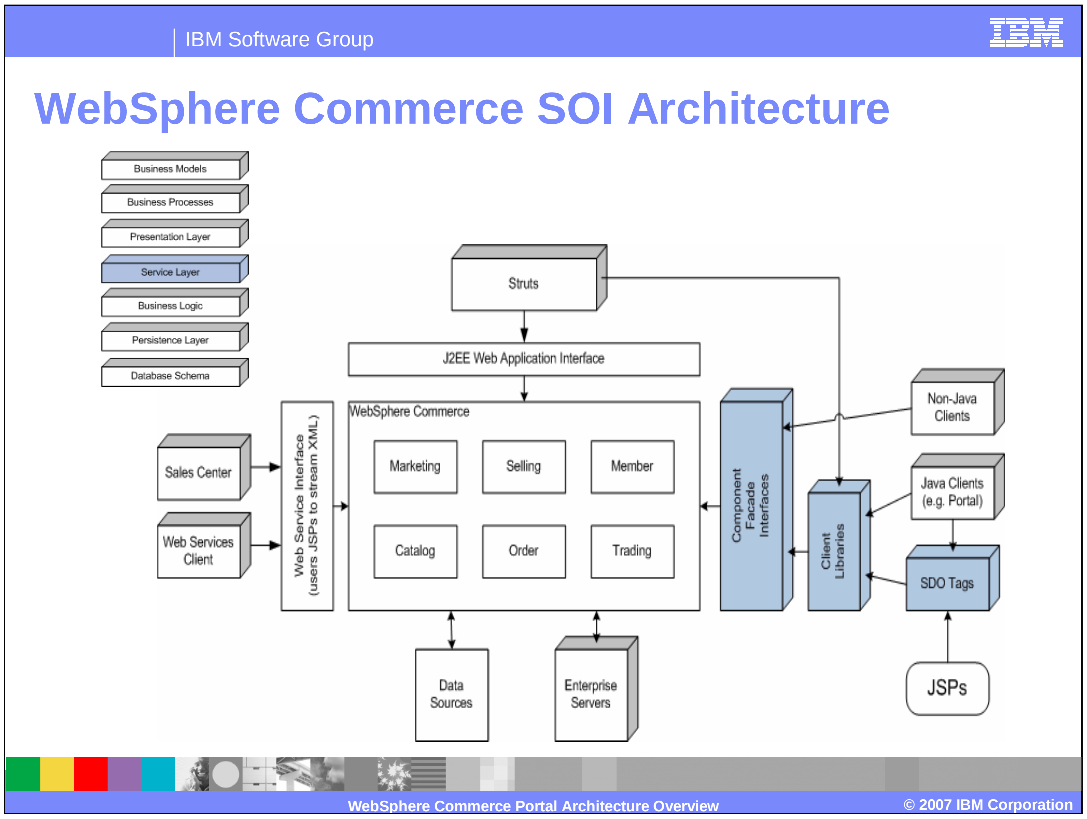
Now let's look at SOA Adoption and Service Oriented Integration in WebSphere Commerce. IBM describes four phases for SOA adoption. Phase 1, ***Implementing individual Web services***, focuses on creating services from tasks contained in new or existing applications. Phase 2, ***Service-oriented integration of business functions,*** focuses on integrating services across multiple applications inside and outside the enterprise for a business objective. Phase 3**,** ***Enterprise-wide IT transformation***, is an architected implementation enabling integration across business functions throughout an enterprise. Phase 4, ***On Demand Business Transformation***, is a broad transformation of existing business models or the deployment of new business models.

WebSphere Commerce Enhancements for version 6 addresses *phase 2* adoption, which is known as *Service Oriented Integration* (SOI). Unlike *phase 1* where a set of services are provided in a more ad-hoc basis to enable applications to communicate, *phase 2* is more focused on enabling scenarios. So while an adoption at *phase 1* may enable an external system to update the inventory in your application using the exposure of an UpdateInventory service or get the price by using a PriceCheck service, *phase 2* would expose a related set of inventory services on your application. This enables your application to delegate inventory responsibilities to an external system.

As part of the service-oriented integration of business functions, WebSphere Commerce has enabled *Service Oriented Integration* (SOI) scenarios specifically focused on back-office integration. WebSphere Commerce is defining connections from its *Business Application Services* to *Access Services*. The two scenarios of focus are with external order management systems (OMS) and with enterprise resource planning systems (ERP).

This stage of SOA adoption allows you to: incorporate WebSphere Commerce into an overall SOA Enterprise Architecture as both a producer and consumer of business services, integrate with legacy systems and vendors using a Service Oriented Integration model and dynamically adapt WebSphere Commerce processes and services to fit their business.

WebSphere Portal integration provides the *interaction services* in this diagram. These interaction services provide the presentation layer to support the business services communicating over the ESB. Portals allow for aggregation and interaction of different services provided by different systems, and enables sharing contextual information across different services, for example, authentication information. For example, by using WebSphere Commerce Portal integration, you can provide a presentation for *business application services* provided by WebSphere Commerce and other service providers, and enable interactions between these service providers.

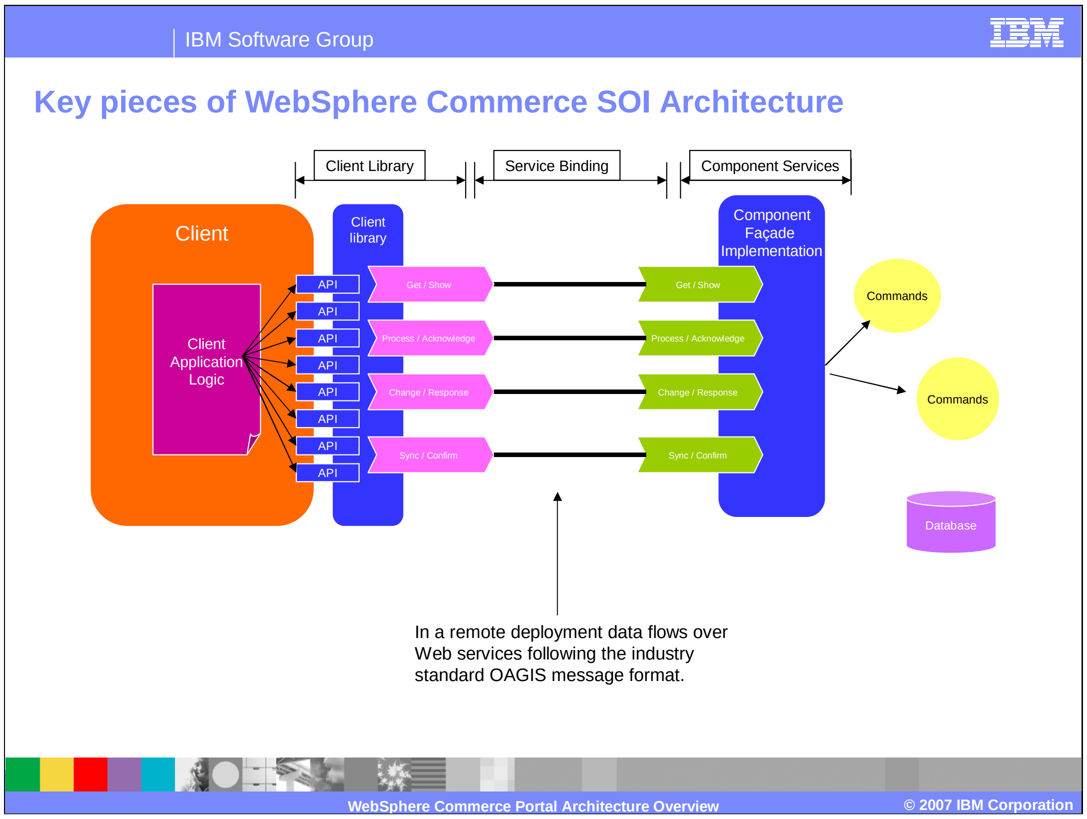


In previous releases of WebSphere Commerce, the framework was designed for Web-based transactions. Now, WebSphere Commerce is multi-channel-enabled, meaning that WebSphere Commerce can support transactions across various sales channels. The framework enhancements in this release support multiple presentation layers, responsible for displaying results, which decouple control logic from business logic.

Notice the service layer on the left side of the diagram. The service layer, implemented using OAGIS messages, is a channel-independent mechanism that can access WebSphere Commerce business logic. The service layer segregates the implementation of business logic such as order and catalog. This segregation permits the underlying implementation to change without requiring that the caller change. All clients, including Web clients and back-end services, go through the service layer to run business logic. The service layer supports two transport mechanisms: local Java binding and Web services.

A Web service is a series of open protocols and standards used in communication between two systems. In WebSphere Commerce, Web services perform functions ranging from simple requests to complicated business processes. After a Web service is registered, other applications can discover it. In this release, WebSphere Commerce has a new Web services framework. This new framework has the following capabilities: it leverages the WebSphere Web Service Runtime Infrastructure and Rational tools, promotes the use of industry-standard service definitions, allows top-down and bottom-up creation of Web services and leverages the JSP page composition service to generate the Web service response, allowing dynamic caching to be optimized (for either full-page or fragment caching).

The new Web services framework uses the existing command pattern to represent the business logic, and allows for the existing URL-based controller commands to also be used by the Web services channel. WebSphere Commerce can be the service provider by enabling its business functions as Web services that can be accessed by external systems. Also, WebSphere Commerce can be the service requester by enabling it to invoke Web services that are hosted by external systems.


IBM Software Group

Key pieces of WebSphere Commerce SOI Architecture

In a remote deployment data flows over Web services following the industry standard OAGIS message format.

WebSphere Commerce Portal Architecture Overview © 2007 IBM Corporation

Starting with the WebSphere Commerce 6.0 release, the runtime environment has started to decouple the presentation layer from the business logic layer. With the release of WebSphere Commerce 6.0, a generic facade was placed between the Struts framework and the command layer. This generic facade handles the name-value pair parameters to the WebSphere Commerce commands. In this release, instead of just having a single generic facade, well-defined facades for various business logic subsystems have been introduced. These facades take structured objects instead of name-value pair input and follow a model known as the OAGIS model.

With this new model, there is a clean separation between the presentation layer and the business logic. Regardless of whether the request is to get data or change data, the request is independent of the presentation layer and the subsystem will not have different logic depending on whether the request was from a browser or Web services.

The starting point of a WebSphere Commerce service is the definition of the high level business object called the noun. Based on the noun, services are defined based on supported verbs that can act upon that noun. These combinations of verb and noun form the OAGIS-style messages that represent a WebSphere Commerce service request. The functional architecture is structured around the transmission of these OAGIS messages from the client (for example, a portlet in a WebSphere Commerce Portal server) to the WebSphere Commerce server, and back again.

This diagram shows how a client such as a portlet can use WebSphere Commerce business logic:

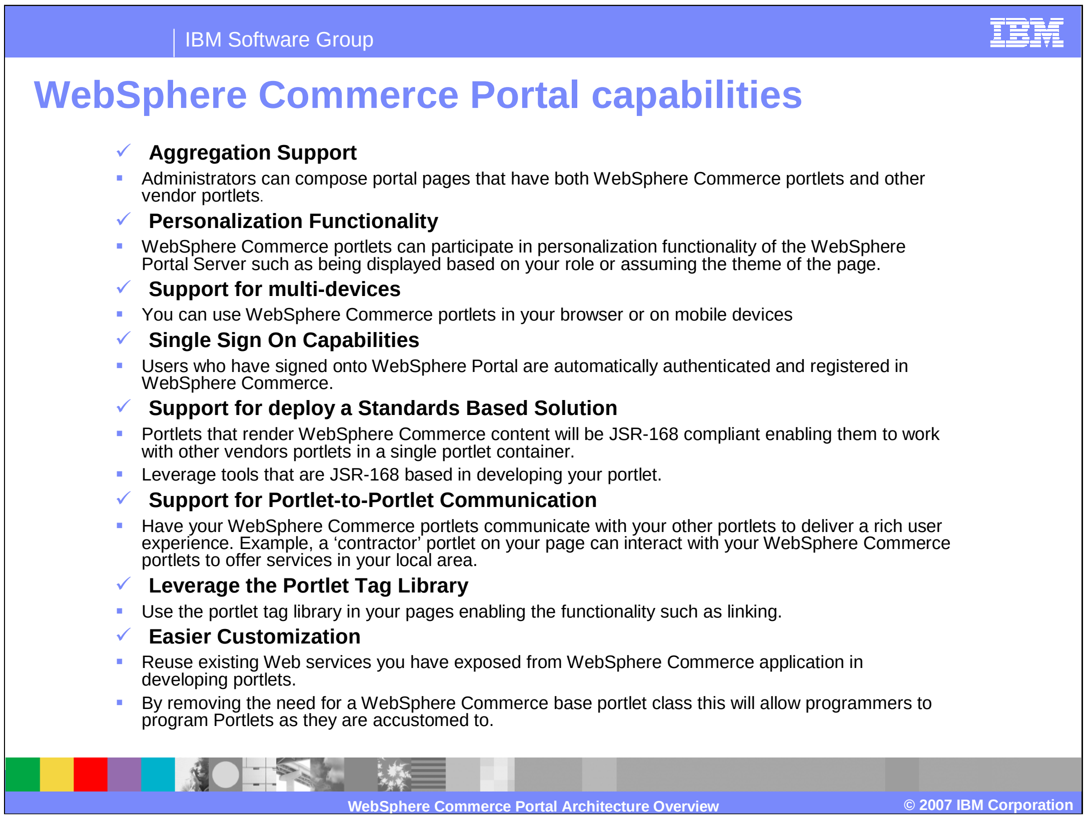
+ The client (for example, a portlet in a WebSphere Portal Server) uses the client library to create an OAGIS message.

+ The communication layer then routes the request and handles the serialization and deserialization of the OAGIS message and uses Web services to send the request to the component facade.

+ The component facade calls the appropriate WebSphere Commerce commands and forms the response.

+ The communication layer then routes the response back to the client library.

+ Finally, the client library returns the response message to the client.

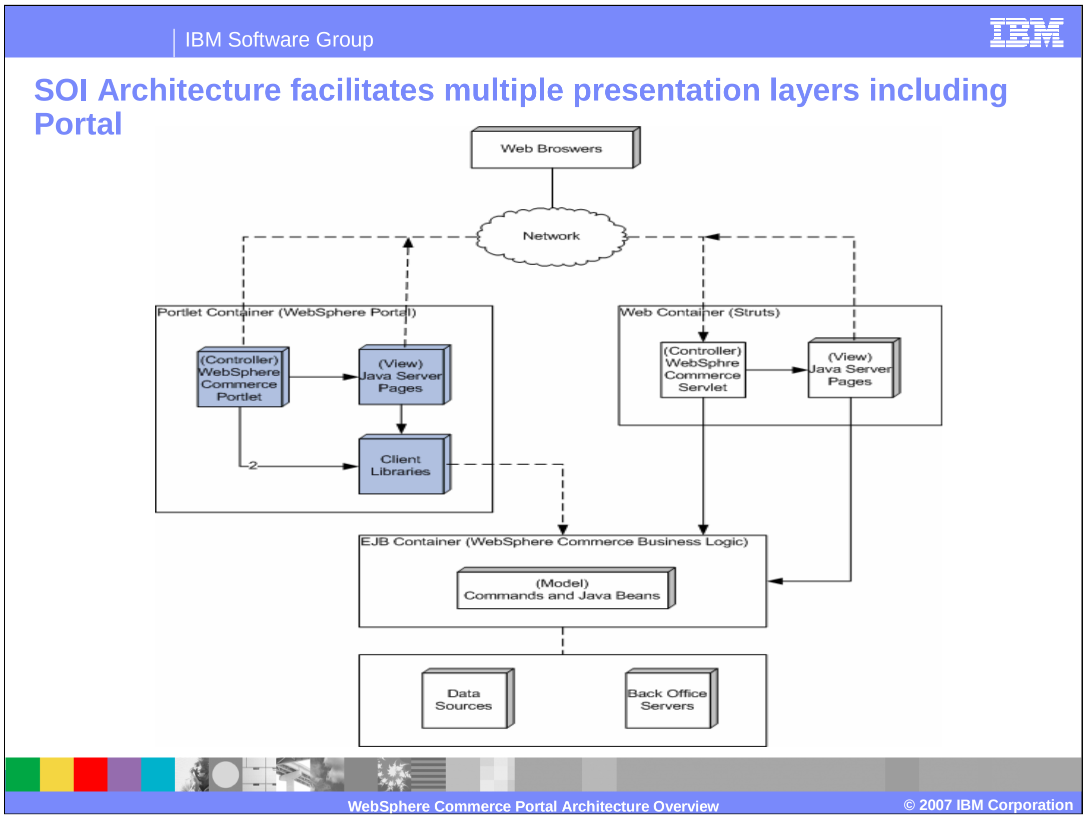

IBM Software Group

WebSphere Commerce Portal capabilities

- ✓ **Aggregation Support**
  - Administrators can compose portal pages that have both WebSphere Commerce portlets and other vendor portlets.
- ✓ **Personalization Functionality**
  - WebSphere Commerce portlets can participate in personalization functionality of the WebSphere Portal Server such as being displayed based on your role or assuming the theme of the page.
- ✓ **Support for multi-devices**
  - You can use WebSphere Commerce portlets in your browser or on mobile devices
- ✓ **Single Sign On Capabilities**
  - Users who have signed onto WebSphere Portal are automatically authenticated and registered in WebSphere Commerce.
- ✓ **Support for deploy a Standards Based Solution**
  - Portlets that render WebSphere Commerce content will be JSR-168 compliant enabling them to work with other vendors portlets in a single portlet container.
  - Leverage tools that are JSR-168 based in developing your portlet.
- ✓ **Support for Portlet-to-Portlet Communication**
  - Have your WebSphere Commerce portlets communicate with your other portlets to deliver a rich user experience. Example, a 'contractor' portlet on your page can interact with your WebSphere Commerce portlets to offer services in your local area.
- ✓ **Leverage the Portlet Tag Library**
  - Use the portlet tag library in your pages enabling the functionality such as linking.
- ✓ **Easier Customization**
  - Reuse existing Web services you have exposed from WebSphere Commerce application in developing portlets.
  - By removing the need for a WebSphere Commerce base portlet class this will allow programmers to program Portlets as they are accustomed to.

WebSphere Commerce Portal Architecture Overview © 2007 IBM Corporation

The following capabilities exist in the new WebSphere Commerce WebSphere Portal integration: aggregation support, personalization functionality, support for multi-devices, single sign on capabilities, support for deploying a standards based solution, support for portlet-to-portlet communication, the ability to leverage the portlet tag library and easier customization.



IBM Software Group

SOI Architecture facilitates multiple presentation layers including Portal

Web Broswers
Network
Portlet Container (WebSphere Portal)
(Controller) WebSphere Commerce Portlet
(View) Java Server Pages
Client Libraries
2
Web Container (Struts)
(Controller) WebSphre Commerce Servlet
(View) Java Server Pages
EJB Container (WebSphere Commerce Business Logic)
(Model) Commands and Java Beans
Data Sources
Back Office Servers

WebSphere Commerce Portal Architecture Overview

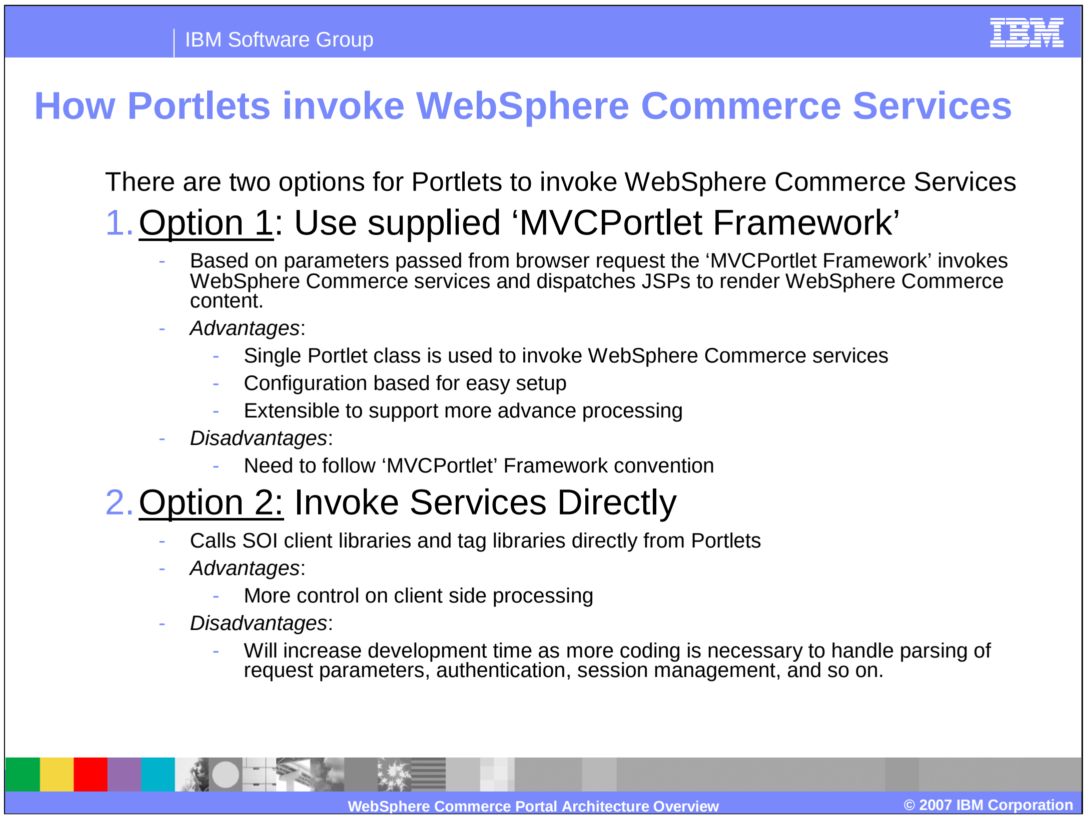
© 2007 IBM Corporation

WebSphere Commerce supports multiple presentations layers; you can use an appropriate presentation layer based on your business requirements. For example, if your business processes are represented by WebSphere Commerce business logic and data then Struts is an appropriate choice for your presentation layer. However, if you want to aggregate WebSphere Commerce business processes with non-Commerce business processes from other applications then it is appropriate to use WebSphere Portal for the presentation layer.

Both the Struts and Portal presentation layers follow the Model-View-Controller (MVC) design pattern; the MVC design pattern separates control between the business logic and presentation logic. This separation enables a Web designer to develop the presentation layer at the same time that an application developer implements business logic.

In the Struts framework, a browser request is routed to a servlet that acts as a controller. The controller, using local Java calls, calls the model for processing. The controller then dispatches the appropriate view to render data. The model encapsulates all business logic (implemented by following the command pattern) and data (implemented using JSP pages). The JSP pages retrieve data from the database using data beans then format the output.

In the Portal framework, the browser request is routed to a portlet that acts as a controller. The portlet calls client libraries (Java classes). The client library sends a service request to WebSphere Commerce business logic for processing. When the portlet renders it dispatches to a JSP in the portlet container. The JSP pages use tags that delegate to the client libraries to retrieve data from the WebSphere Commerce system.


IBM Software Group

How Portlets invoke WebSphere Commerce Services

There are two options for Portlets to invoke WebSphere Commerce Services

1. Option 1: Use supplied 'MVCPortlet Framework'
   - Based on parameters passed from browser request the 'MVCPortlet Framework' invokes WebSphere Commerce services and dispatches JSPs to render WebSphere Commerce content.
   - *Advantages*:
     - Single Portlet class is used to invoke WebSphere Commerce services
     - Configuration based for easy setup
     - Extensible to support more advance processing
   - *Disadvantages*:
     - Need to follow 'MVCPortlet' Framework convention
2. Option 2: Invoke Services Directly
   - Calls SOI client libraries and tag libraries directly from Portlets
   - *Advantages*:
     - More control on client side processing
   - *Disadvantages*:
     - Will increase development time as more coding is necessary to handle parsing of request parameters, authentication, session management, and so on.

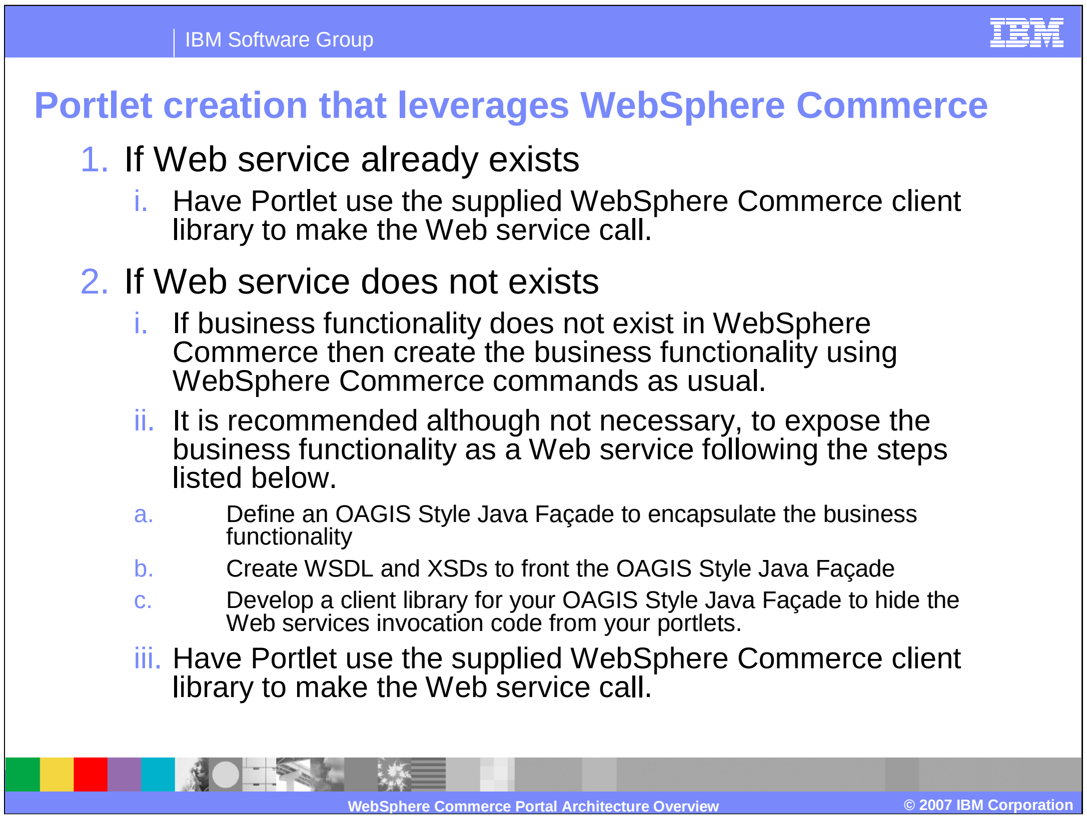
WebSphere Commerce Portal Architecture Overview © 2007 IBM Corporation

There are two main approaches to Customizing WebSphere Commerce Portal integration: Using the provided WebSphere Commerce MVCPortlet class (MVC style) and Using WebSphere Portal general programming techniques (Web service style)

The first approach, is using the provided WebSphere Commerce MVCPortlet class (MVC style)

The WebSphere Commerce Portlet, MVCPortlet, is a generic implementation of the MVC pattern. It allows you to set up one or more portlets, each with its own configuration, to call various WebSphere Commerce services. This programming pattern greatly reduces code redundancy and maintains a consistent behavior across all WebSphere Commerce portlet actions.

In the second approach, using WebSphere Portal general programming techniques (Web service style), a portlet can invoke the backend Web services provided by WebSphere Commerce or a vendor directly without the need for a client library. Although this model is less restrictive than the previous one, you have to write more code and configuration for any customization logic.


IBM Software Group

Portlet creation that leverages WebSphere Commerce

1. If Web service already exists
   i. Have Portlet use the supplied WebSphere Commerce client library to make the Web service call.
2. If Web service does not exists
   i. If business functionality does not exist in WebSphere Commerce then create the business functionality using WebSphere Commerce commands as usual.
   ii. It is recommended although not necessary, to expose the business functionality as a Web service following the steps listed below.
   a. Define an OAGIS Style Java Façade to encapsulate the business functionality
   b. Create WSDL and XSDs to front the OAGIS Style Java Façade
   c. Develop a client library for your OAGIS Style Java Façade to hide the Web services invocation code from your portlets.
   iii. Have Portlet use the supplied WebSphere Commerce client library to make the Web service call.

WebSphere Commerce Portal Architecture Overview

© 2007 IBM Corporation

Creating a portlet that leverages the WebSphere Commerce business functionality depends on whether or not WebSphere Commerce Web services already exist. If WebSphere Commerce Web services already exists then have the portlet use the supplied WebSphere Commerce client library to make the Web service call.

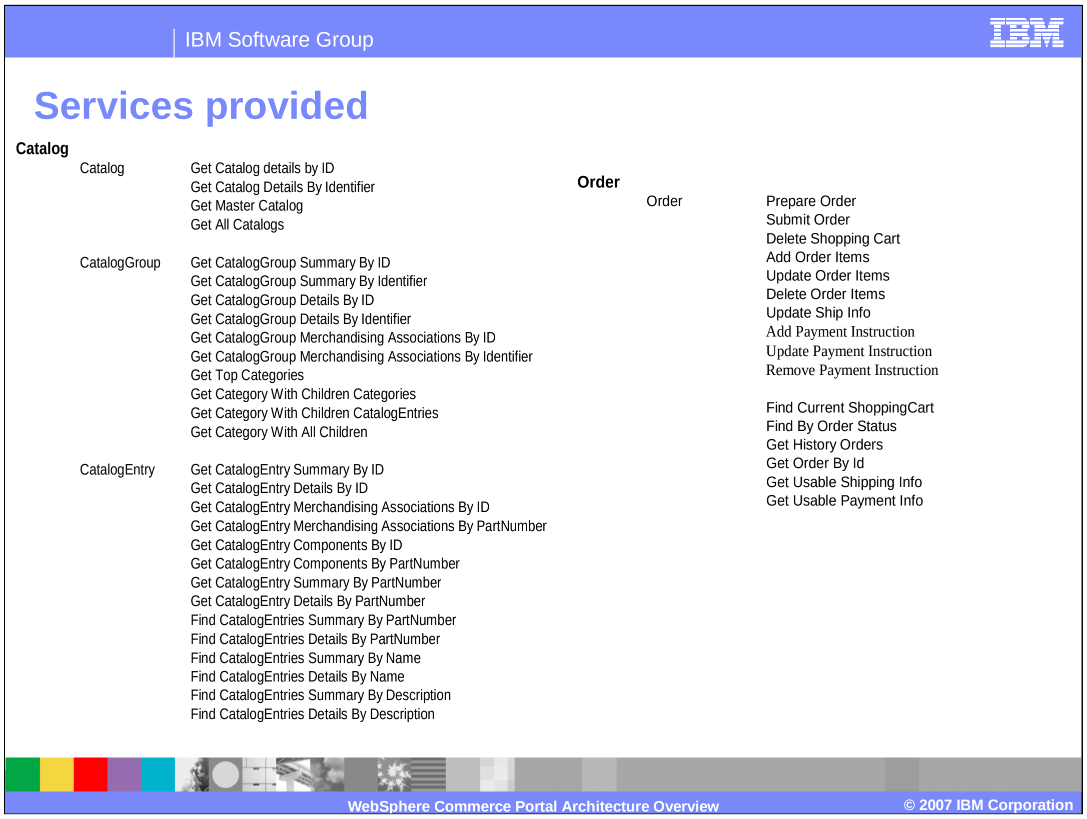
If WebSphere Commerce Web services do not exist, then one must create the business functionality using WebSphere Commerce commands as usual. Although not necessary, it is recommended that one expose the business functionality as a Web service by following these steps:

-Define an OAGIS Style Java Facade to encapsulate the business functionality.

-Then create WSDL and XSD's to front the OAGIS Style Java Façade.

-Then develop a client library for your OAGIS Style Java Façade to hide the Web services invocation code from your portlets.

-Finally, have the portlet use the supplied WebSphere Commerce client library to make the Web service call.


IBM Software Group

Services provided

Catalog

Catalog	Get Catalog details by ID
	Get Catalog Details By Identifier
	Get Master Catalog
	Get All Catalogs
CatalogGroup	Get CatalogGroup Summary By ID
	Get CatalogGroup Summary By Identifier
	Get CatalogGroup Details By ID
	Get CatalogGroup Details By Identifier
	Get CatalogGroup Merchandising Associations By ID
	Get CatalogGroup Merchandising Associations By Identifier
	Get Top Categories
	Get Category With Children Categories
	Get Category With Children CatalogEntries
	Get Category With All Children
CatalogEntry	Get CatalogEntry Summary By ID
	Get CatalogEntry Details By ID
	Get CatalogEntry Merchandising Associations By ID
	Get CatalogEntry Merchandising Associations By PartNumber
	Get CatalogEntry Components By ID
	Get CatalogEntry Components By PartNumber
	Get CatalogEntry Summary By PartNumber
	Get CatalogEntry Details By PartNumber
	Find CatalogEntries Summary By PartNumber
	Find CatalogEntries Details By PartNumber
	Find CatalogEntries Summary By Name
	Find CatalogEntries Details By Name
	Find CatalogEntries Summary By Description
	Find CatalogEntries Details By Description

Order

Order	Prepare Order
	Submit Order
	Delete Shopping Cart
	Add Order Items
	Update Order Items
	Delete Order Items
	Update Ship Info
	Add Payment Instruction
	Update Payment Instruction
	Remove Payment Instruction
	Find Current ShoppingCart
	Find By Order Status
	Get History Orders
	Get Order By Id
	Get Usable Shipping Info
	Get Usable Payment Info

WebSphere Commerce Portal Architecture Overview © 2007 IBM Corporation

There are four services provided in this feature pack: Catalog, Order, Member and Contract.

The **Catalog** services enable an external system, such as WebSphere Portal, to search for Catalog related information in WebSphere Commerce.

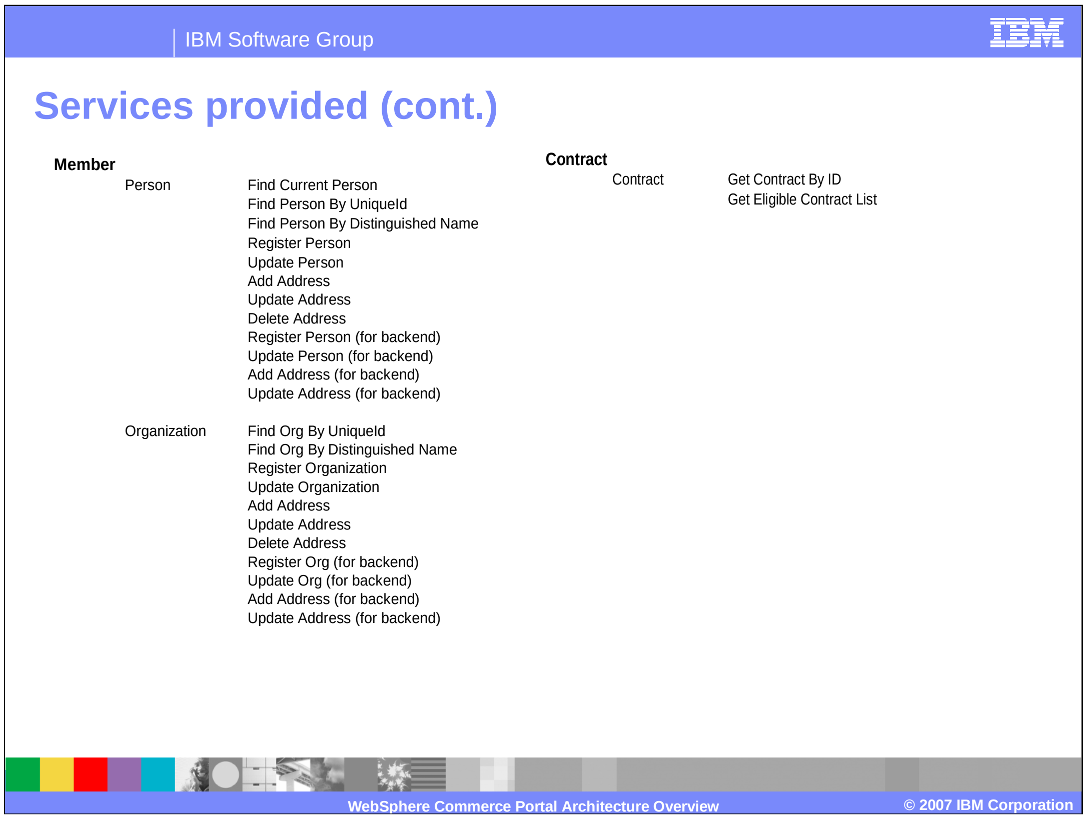
Catalog services use a single verb: Get.

The catalog logical model is defined with customization in mind. All the major elements have a standard extension point to add additional name-value pairs. The catalog entry object in particular has only the major elements defined.

The **Order** Management subsystem is a component of the WebSphere Commerce Server that provides shopping carts, order capture, order fulfillment, inventory, and payment function support.

The Order Management subsystem provides a set of services that can interact with other systems, such as WebSphere Portal. These services represent a subset of the functionality of the Order Management subsystem.

Order services use one noun: Order. Order services use three verbs: Process, Change, and Get.



IBM Software Group

Services provided (cont.)

Member

Person	Find Current Person	
	Find Person By UniqueId	
	Find Person By Distinguished Name	
	Register Person	
	Update Person	
	Add Address	
	Update Address	
	Delete Address	
	Register Person (for backend)	
	Update Person (for backend)	
	Add Address (for backend)	
	Update Address (for backend)	
Organization	Find Org By UniqueId	
	Find Org By Distinguished Name	
	Register Organization	
	Update Organization	
	Add Address	
	Update Address	
	Delete Address	
	Register Org (for backend)	
	Update Org (for backend)	
	Add Address (for backend)	
	Update Address (for backend)	

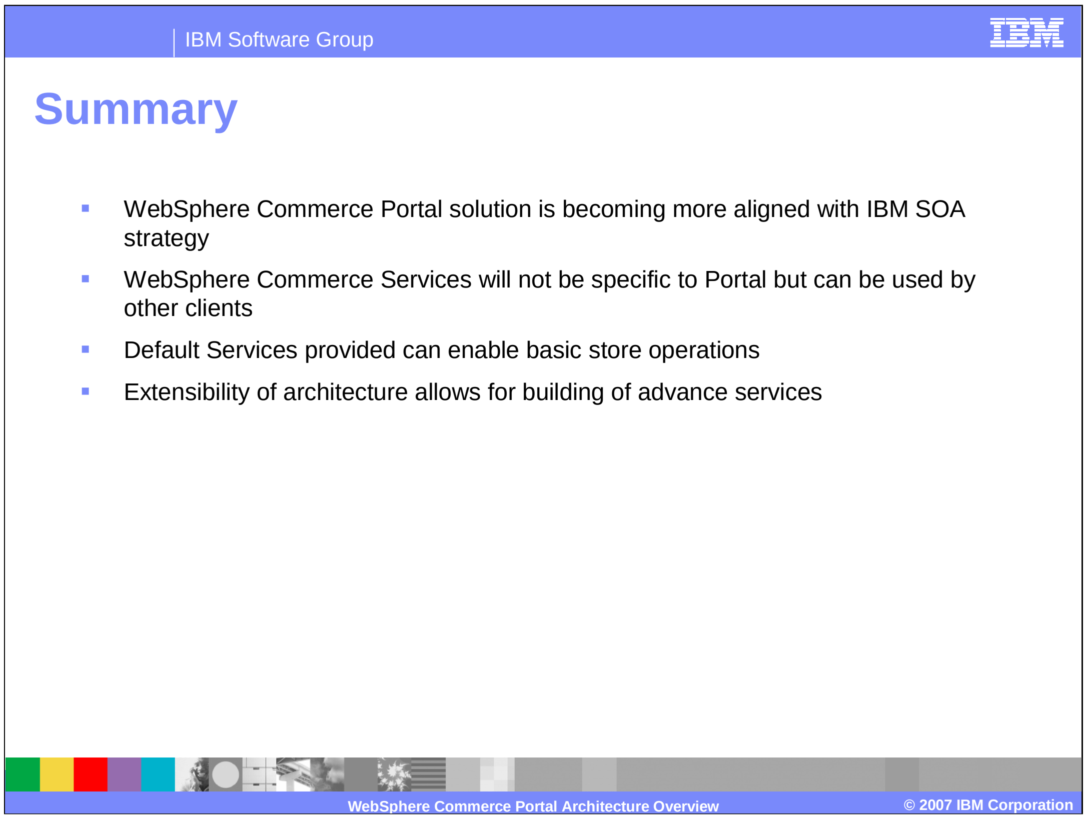
Contract

Contract	Get Contract By ID
	Get Eligible Contract List

WebSphere Commerce Portal Architecture Overview © 2007 IBM Corporation

The **Member** services allow an external system, such as WebSphere Portal, to create, update and search for Members (Organizations, Users, and Member Groups) in WebSphere Commerce. The member component provides a client library and a component facade. Member services use the Person and Organization nouns and the following verbs: Get, Process, Change, and Sync.

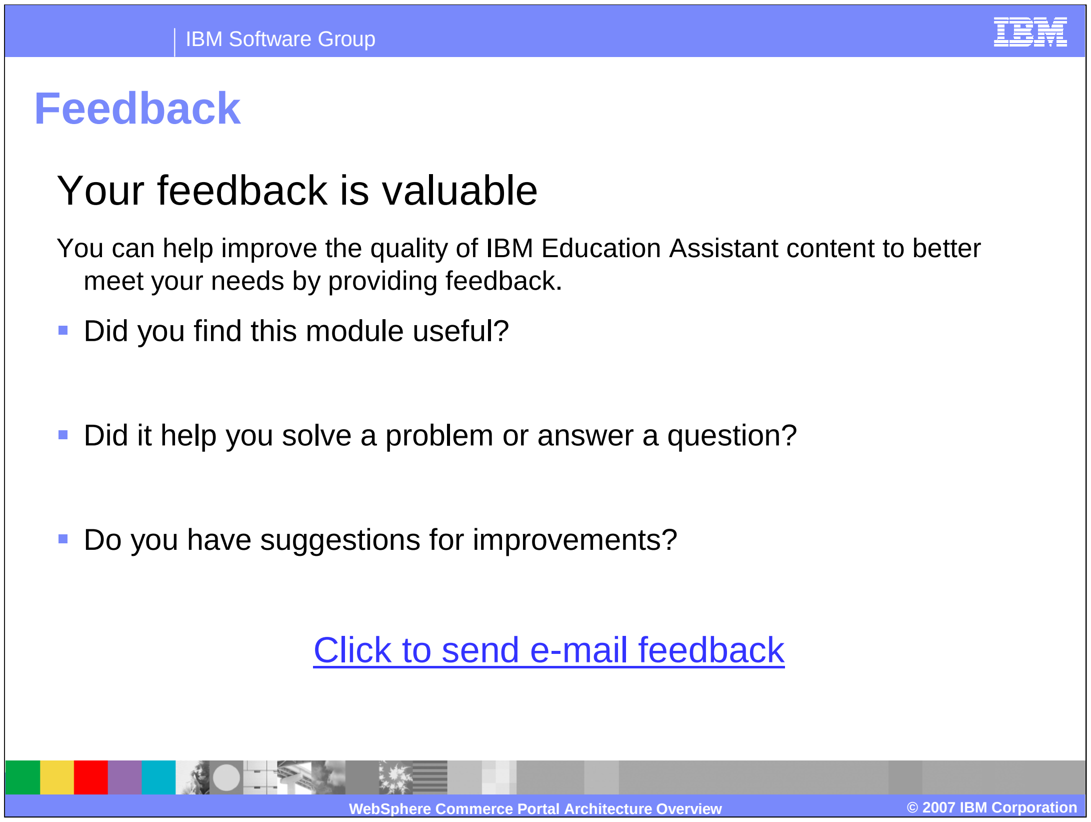
The **Contract** services define a contract noun and Get verb. The contract noun includes the contract Id, contract name and contract description. The contract noun retrieves all entitled contracts for a given user in a store and retrieves the contract content by contract Id.



Summary

- WebSphere Commerce Portal solution is becoming more aligned with IBM SOA strategy
- WebSphere Commerce Services will not be specific to Portal but can be used by other clients
- Default Services provided can enable basic store operations
- Extensibility of architecture allows for building of advance services

In summary, WebSphere Portal WebSphere Commerce solution is becoming better aligned with the IBM SOA strategy. This will be very beneficial for you in the future. The WebSphere Commerce services provided will not be specific to Portal but can be used by other clients if needed. Customers will find that the services, out of the box, will enable basic store operations. By integrating with the IBM SOA strategy, the WebSphere Commerce architecture is extensible which allows for the building of advanced services.



Feedback

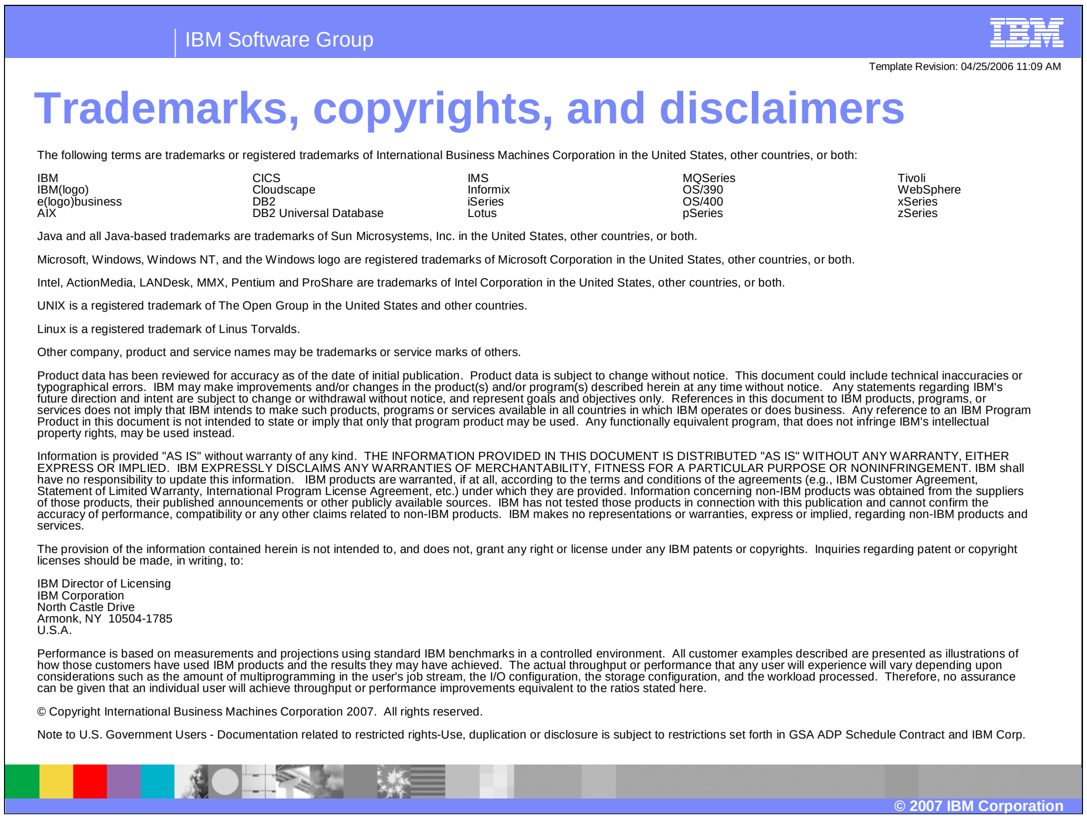
Your feedback is valuable

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

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



Template Revision: 04/25/2006 11:09 AM

Trademarks, copyrights, and disclaimers