SWENG568: Enterprise Integration

Lesson 7: SOA/BPM: The Future of Integration

Introduction (1 of 9)

Learning Objectives

- Master a method to design and develop bottom up Web Services
- Learn BPEL (Integration through service composition at ***the process level***)
- Understand SOA/BPM – the future of integration

By the end of this week, make sure you have completed the readings and activities found in the Lesson 7 Course Schedule.

The Business Need for Process-level Integration (2 of 9)

The Business Need for Process-level Integration

It is well understood that the vast technology landscape, numerous legacy applications, heterogeneous data sources, and changing business environments are the challenges to enterprise-wide integration projects. More specifically, we now have a good understanding of the following generalized issues commonly and frequently confronted by the integration practitioners:

- **Diversity:** Operational and information needs have been diversified from department to department, facility to facility, and corporate to corporate. Therefore, different data and functionalities might be needed for sharing. When they were designed, data would take different "format" and functions would be expected to behave differently as each of them must optimally fit in its local business setting when it was designed, developed, and deployed. In other words, domain cultural specifics are part of their designs, focusing on best meeting the local domain needs.
- **Heterogeneity:** Because of the existence of diversity and the investments made at different time, information silos have been developed using different tools and methodologies, written in different programming languages, and end up running on different operating systems and communications networks. In general, different computing and networking technologies might be used to accommodate the needs and designs at the time when they were developed.
- **Complexity:** Given the issues of diversity and heterogeneity, the complexity of integrating information silos across an enterprise is obvious. To further address the business dynamic needs, corporate best practices capable of accommodating the needs of different cultures have been embodied in information systems through **complex and dynamic business processes**, which make the inherent integration complexity unceasingly increasing.

- **Scalability:** No matter how complicated an integrated enterprise system could be, it has to be scalable. An enterprise couldn't rapidly grow its business unless the integrated information system can continuously deliver all the expected information services to all the employees and customers. The increased service volume and varieties should not degrade the well expected service quality.
- **Agility:** An integrated system has to be flexible, responsive, and adaptable as market demands fluctuate and technologies advance.

Data/functionality sharing at different levels (i.e., operational- or strategic-level) enabled by IT systems (i.e., integrated software systems) in an enterprise essentially provides common data/information/knowledge services in a coordinated and collaborative manner, aimed at assisting or serving employees and customers at the point of need. We also know that different levels of integration will be needed to meet the different needs, operationally, financially, and technically (Figure 7.1).

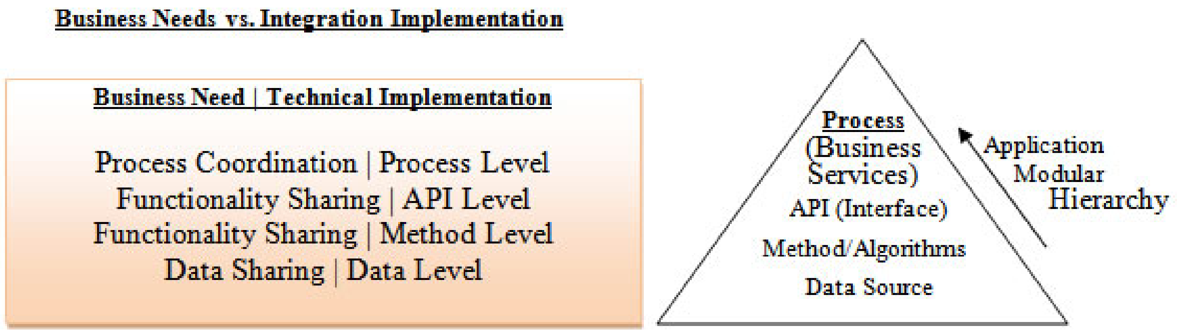

Business Needs vs. Integration Implementation
Business Need | Technical Implementation
Process Coordination | Process Level
Functionality Sharing | API Level
Functionality Sharing | Method Level
Data Sharing | Data Level
Process (Business Services)
API (Interface)
Method/Algorithms
Data Source
Application Modular Hierarchy

Figure 7.1 Business Needs vs. Integration Approaches

According to the Workflow Management Coalition (WFMC) [2], a workflow is "the automation of a business process, in whole or in part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules". The participants can be human, machines, or software systems. The automation of processes is achieved with the help of a software system called a "workflow engine". A process controlled by a workflow engine is viewed as a collection of tasks executed by various resources within a value system comprising one or more integrating organization units (Figure 7.2).

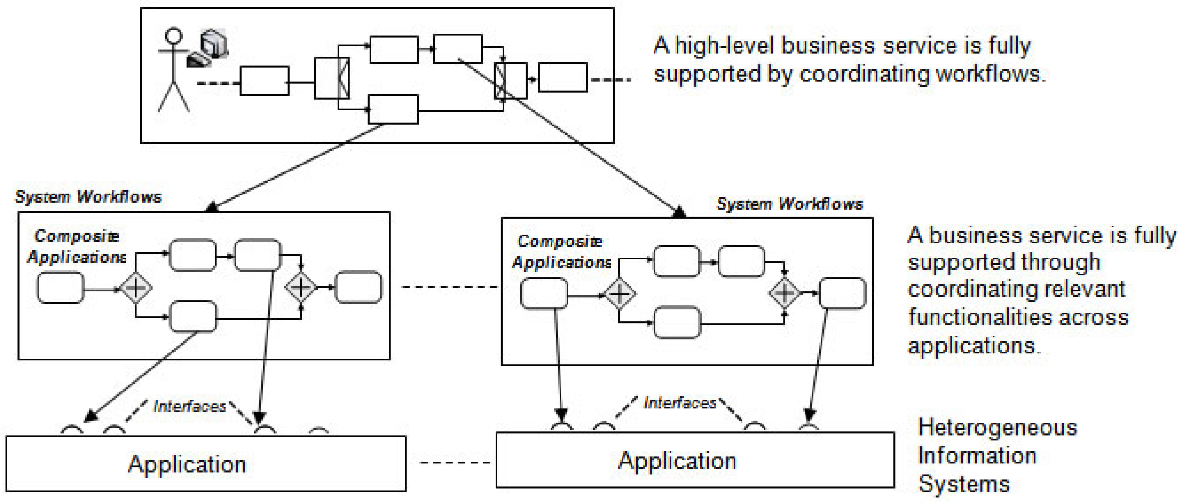

A high-level business service is fully supported by coordinating workflows.
System Workflows
Composite Applications
System Workflows
Composite Applications
A business service is fully supported through coordinating relevant functionalities across applications.
Interfaces
Interfaces
Application
Application
Heterogeneous Information Systems

Figure 7.2 Illustration of Workflows in support of Business Services

As discussed in last lesson, it would be the best scenario if all applications can have their supported processes coordinated to function as a whole to serve the business needs throughout an enterprise. In other words, instead of integrating applications for enabling a common view of data and individual task based coordination, it will be more cost-effective by leveraging enterprise integration to have **business activities** coordinated for the end delivery to directly meet the business needs. However, it is complicated and challenging to integrate workflows across an enterprise.

Figure 7.3 illustrates a simple example of an integration scenario at the process-level. As a business service managing student information that is defined at the process level in the "Enrollment/Bursar Operation" Department and the "Course Registration" Department respectively. Note that their corresponding ***Studentinfo*** workflows can be implemented using different technologies as their underlying information systems are technically not the same. No matter how each workflow is technically implemented and which workflow will be triggered at a given time, as soon as it is executed the workflow logic will ensure that the updated student information will be synchronized between two applications.

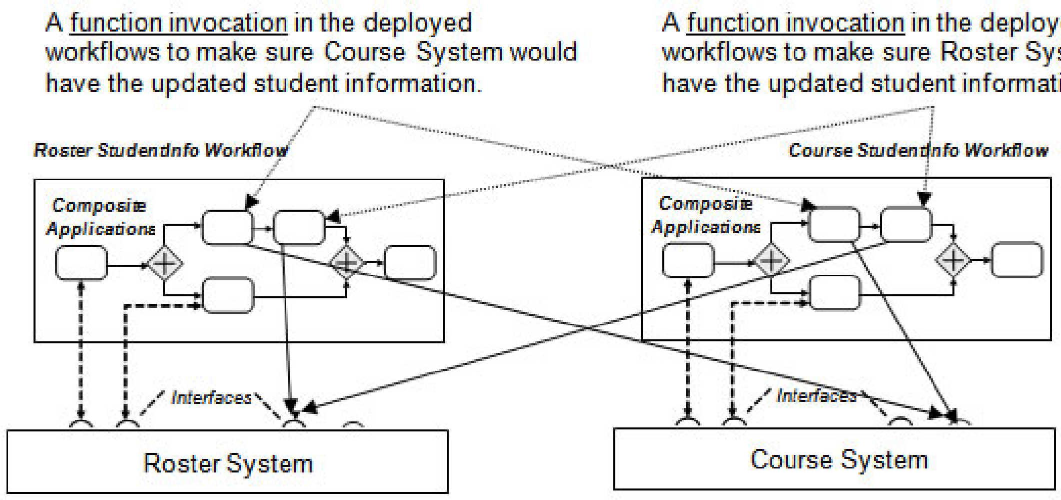

A function invocation in the deployed workflows to make sure Course System would have the updated student information.
A function invocation in the deployed workflows to make sure Roster System would have the updated student information.
Roster Studendnfo Workflow
Course Studendnfo Workflow
Composite Applications
Composite Applications
Interfaces
Interfaces
Roster System
Course System

Figure 7.3 Example of Integrating Workflows in support of Business Services

Although this example is extremely simple, it clearly demonstrates the basic concept of integration at a process level. That is, using functionality sharing while coordinating the invocations at the proves level, the business need in terms of data sharing to ensure a common view between applications is well supported.

Assume ***White-Blue iShopper*** is an online retailer, selling iPads, laptops, desktop computers, and accessories; two applications: web-based ***Online Store*** system and ***Warehouse*** management system. The Online Store is java-based and uses J2EE architecture, running on a UNIX box and relying on MySQL database; and the Warehouse is C# based client-server application, running on PC servers and using MS SQL server. Compared with integrated applications for enabling a common view of data and individual task based coordination, the following example shows the difference how we can cost-effectively have business activities coordinated at the process level to accomplish an order fulfillment business operation need (Figure

7.4). The coordinated interactions between applications follow well designed business logic, so the expected business productivity and customer service quality can be more likely assured. Table 7.1 highlights the main potential positive outcomes and differences acquired from different integration methods.

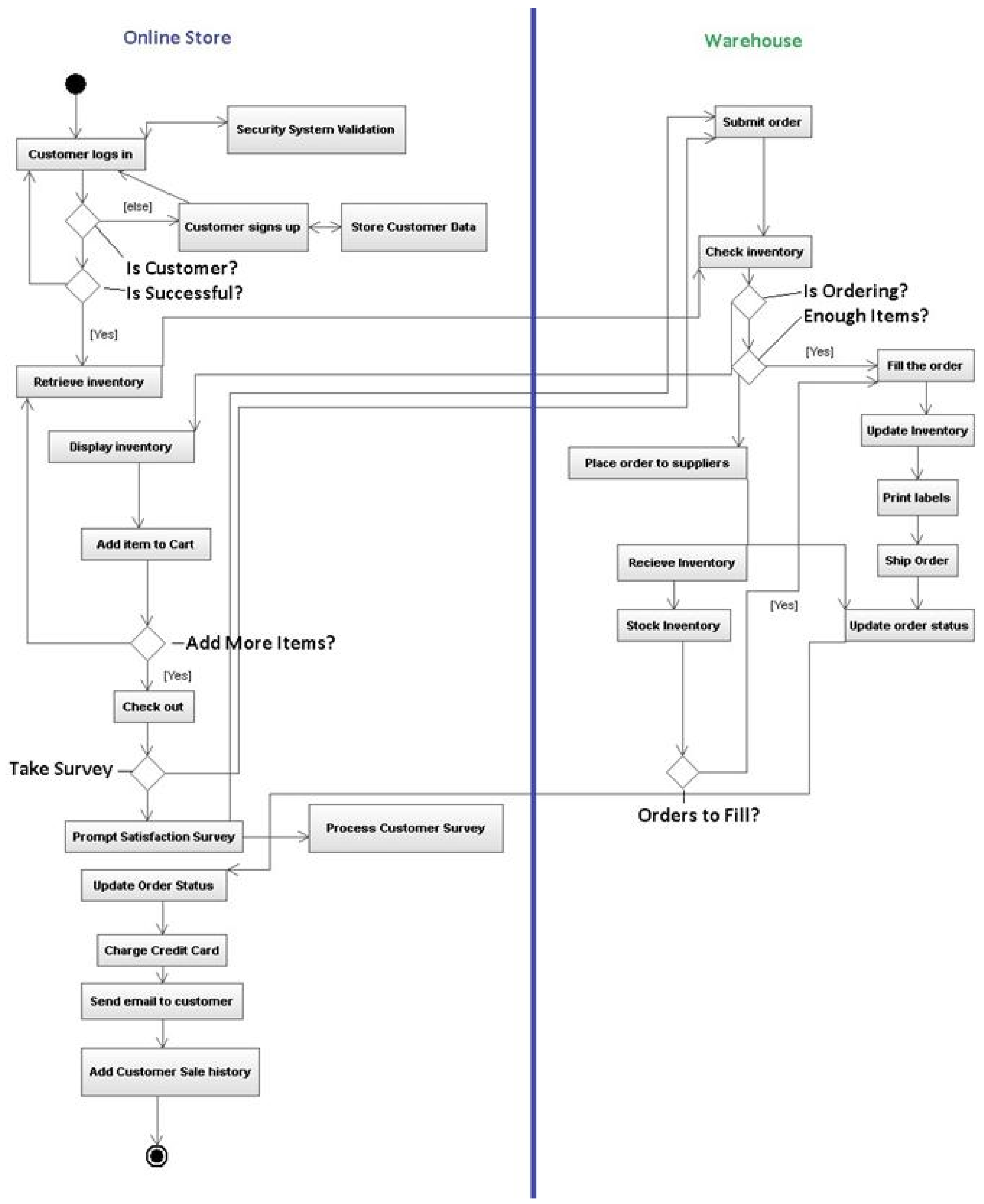


Figure 7.4 Example of Integrating Business Activities at the Process Level

Integration Methods	**Potential Outcomes and Differences**
Data-level	Orders related data will be synchronized. It is purely data-driven.

Integration Methods	Potential Outcomes and Differences
Method-level or API-level	Certain tasks are coordinated. Coordination occurs only when certain individual task is executed. It is task-based.
Process-level	All the tasks are coordinated based the defined business logic. A task can be triggered only if all the preconditions are satisfied, which is warranted by the workflow logic. Business goals are realized through collaborated business activities. It is business-goal oriented.

Table 7.1 Comparisons among different integration methods using an order fulfillment example

Challenges and the Solutions

Although we see the benefits of integrating applications at the process level, it remains extremely challenging if inappropriate computing technologies are used to implement it. If not well planned and executed, the integrated applications might deliver the information services as needed, but enterprise integration as a whole could be proprietary and rigid without good reliability, scalability, and adaptability.

In Lesson 6, a generic service-oriented IT computing architecture for the development of a state-of-the-art enterprise network is illustrated in Figure 6.14. The top two layers represent services operations from the business process perspective while the bottom three layers shows the value-adding services processes from the computing perspective. As Web services are standard runtime technologies over the Internet, providing best ever mechanisms for addressing heterogeneous computing issues, Web services evolves as the desirable computing technology to support high performance, scalability, reliability, interoperability, and availability of distributed service-oriented IT systems in an intra- or inter-enterprise setting [3].

By converging service-oriented architecture (SOA) and business process management (BPM) technologies, the concept of enterprise service computing emerges, which enables an "optimal" approach to addressing enterprise integration challenges to meet the future needs. The enabled approach can fully take advantage of the state-of-the-art computing technologies and new IT solutions while optimally leveraging the existing information systems. It allows an enterprise to meet its growth needs, financially and technically. Figure 7.5 clearly indicates how enterprise integrations should be architected for today and the future.

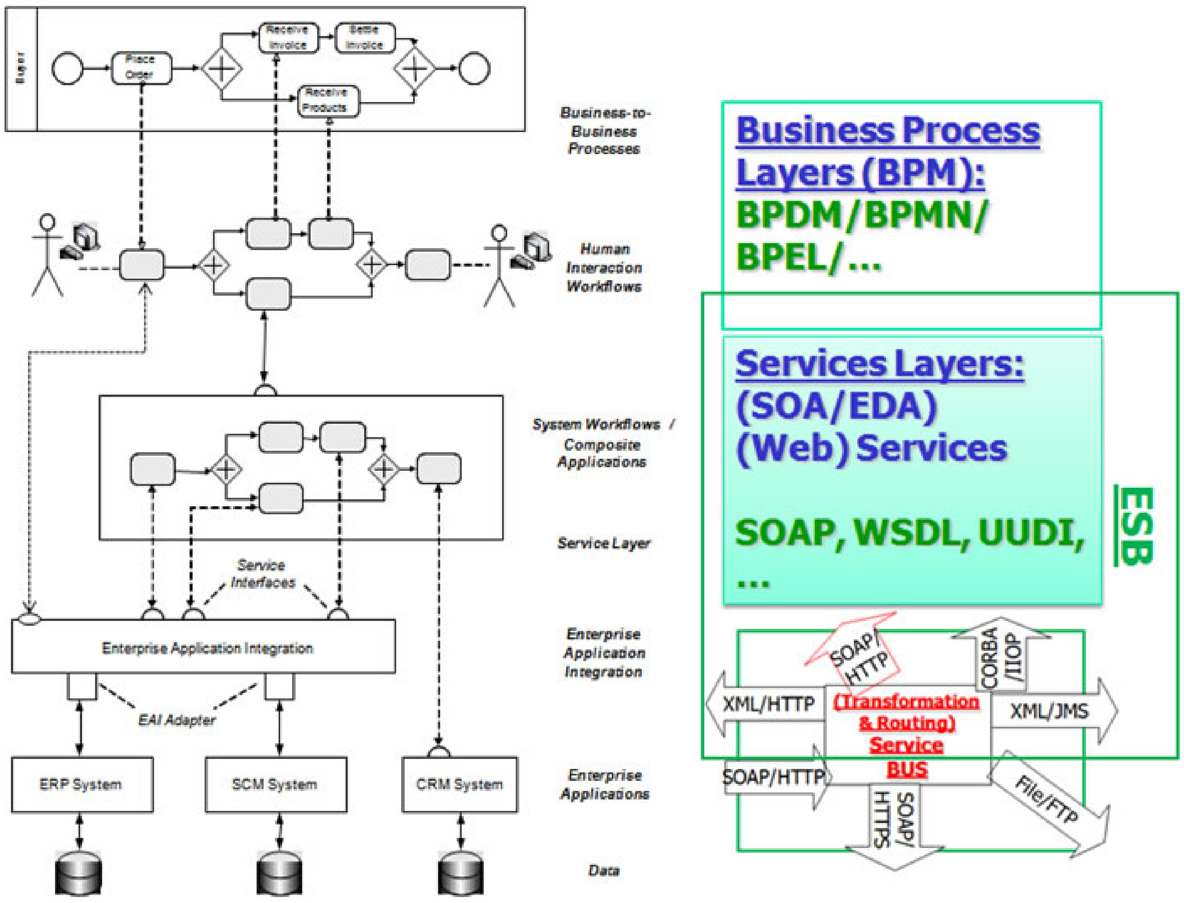

Buyer
Place Order
Receive Invoice
Settle Invoice
Receive Products
Business-to-Business Processes
Human Interaction Workflows
System Workflows / Composite Applications
Service Layer
Service Interfaces
Enterprise Application Integration
EAI Adapter
ERP System
SCM System
CRM System
Enterprise Applications
Data
Business Process Layers (BPM): BPDM/BPMN/ BPEL/...
Services Layers: (SOA/EDA) (Web) Services
SOAP, WSDL, UUDI, ...
ESB
SOAP/ HTTP
CORBA /IIOP
XML/HTTP
(Transformation & Routing) Service BUS
XML/JMS
SOAP/HTTP
SOAP/ HTTPS
File/FTP

Figure 7.5 BPM/SOA based Enterprise Integration Model

Business Process Management

By focusing on aligning all aspects of an enterprise to meet the needs of end users and customers, business process management (BPM) is a holistic management approach that promotes business effectiveness and efficiency while fully striving for innovation, flexibility, and **integration** with the start-of-the-art technology. By moving away from conventional and proprietary workflow mindsets, BPM attempts to improve business processes continuously through executing a well-defined life cycle management model (Figure 7.6), its life cycle activities are essentially grouped into five categories: design, modeling, execution, monitoring, and optimization. When BPM is practically implemented, it unceasingly iterates throughout the life cycle of design, modeling, execution, monitoring, and optimization. Within a new iteration it focuses on further improving enterprise's competitiveness [4].

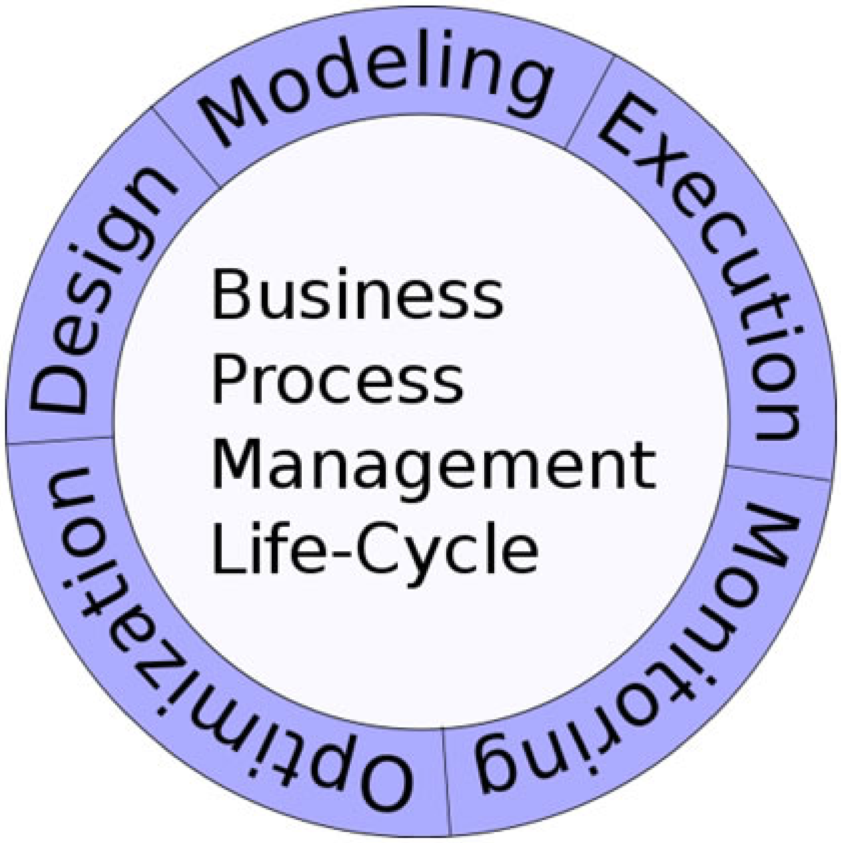


Figure 7.6 BPM Activities throughout its Life Cycle (Copyright © http://en.wikipedia.org/wiki/Business_process_management (http://en.wikipedia.org/wiki/Business_process_management))

Technically, BPM is supported by a suite of standards and specification using enabling **XML-based** computing technologies, such as business process definition metamodel (BPDM), business process modeling notation (BPMN), and business process execution language (BPEL) (Figure 7.5) [4]. According to [4], there are four critical generalized components of a BPM Suite in practice (Figure 7.7):

- *Process Engine – a robust platform for modeling and executing process-based applications, including business rules*
- *Business Analytics – enable managers to identify business issues, trends, and opportunities with reports and dashboards and react accordingly*
- *Content Management – provides a system for storing and securing electronic documents, images, and other files*
- *Collaboration Tools – remove intra- and interdepartmental communication barriers through discussion forums, dynamic workspaces, and message boards*

To make sure that the above critical components are fully operational to meet the business goal of an enterprise, BPM addresses the following critical enterprise integration issues from the holistic management perspective [4]:

- *Managing end-to-end, customer-facing processes*
- *Consolidating data and increasing visibility into and access to associated data and information*

- *Increasing the flexibility and functionality of current infrastructure and data*
- *Integrating with existing systems and leveraging emerging service oriented architecture (SOAs)*
- *Establishing a common language for business-IT alignment*

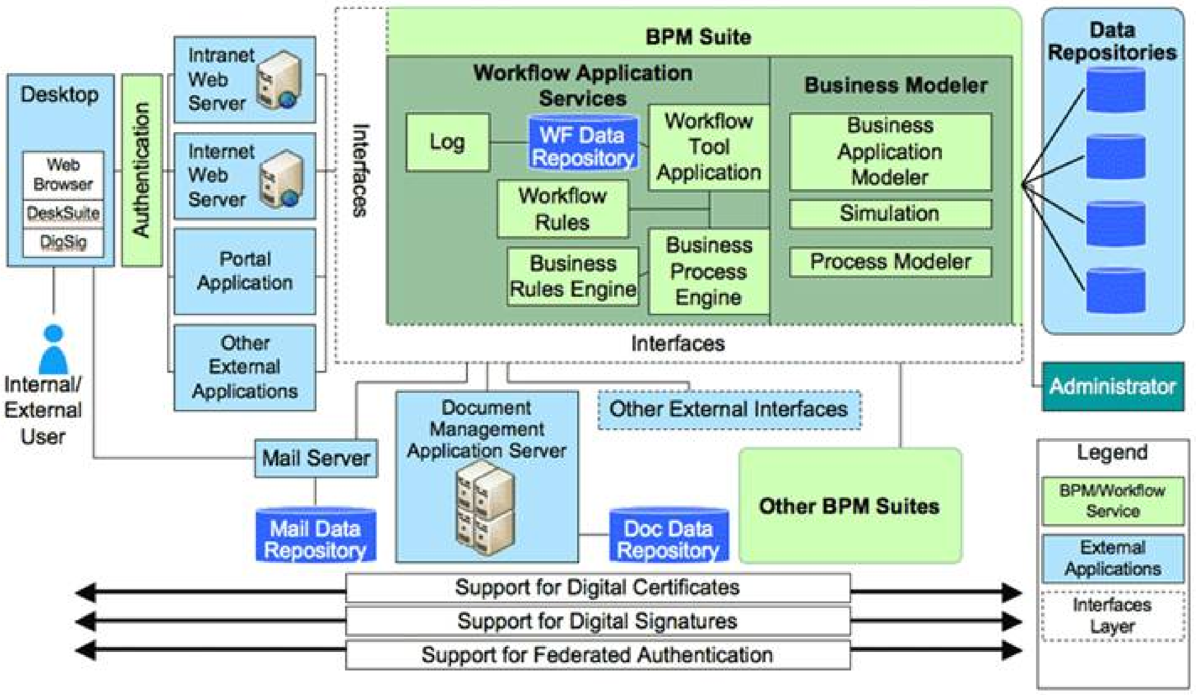


Figure 7.7 Schematic View of BPM Suite in Practice (Copyright © http://en.wikipedia.org/wiki/Business_process_management (http://en.wikipedia.org/wiki/Business_process_management))

Service-oriented Architecture

A **service-oriented architecture** (**SOA**) is a set of design principles used during the phases of systems development and integration. A deployed SOA-based architecture will provide a loosely-integrated suite of services that can be used within multiple business domains. A business process is a collection of related, structured activities that produce a service or product that meet the business needs. Enterprise integration essentially focuses on how these business activities can be well supported (ultimately automated) using IT systems. When these IT services all can be Web services based, enterprise integration becomes more cost-effective and competitive.

SOA defines how to integrate widely disparate applications that are Web services based and running on different platforms. Rather than defining an API, SOA defines the interface in terms of protocols and functionality. In general, service-orientation requires loose coupling of services with operating systems, programming languages, and other technologies that underlie these integrated applications. SOA separates functions into distinct units, or services, and these services and their corresponding consumers communicate with each other in a coordinated fashion through passing messages in a well-defined and shared format [4].

The following guiding principles define the ground rules for development, maintenance, and usage of the SOA:

- *reuse, granularity, modularity, composability, componentization and interoperability.*

- *standards-compliance (both common and industry-specific).*
- *services identification and categorization, provisioning and delivery, and monitoring and tracking.*

More specifically, SOA promotes and fosters the following specific architectural principles for design and service definition in practice [4]:

- *Service encapsulation – Many services are consolidated for use under the SOA. Often such services were not planned to be under SOA.*
- *Service loose coupling – Services maintain a relationship that minimizes dependencies and only requires that they maintain an awareness of each other.*
- *Service contract – Services adhere to a communications agreement, as defined collectively by one or more service-description documents.*
- *Service abstraction – Beyond descriptions in the service contract, services hide logic from the outside world.*
- *Service reusability – Logic is divided into services with the intention of promoting reuse.*
- *Service composability – Collections of services can be coordinated and assembled to form composite services.*
- *Service autonomy – Services have control over the logic they encapsulate.*
- *Service discoverability – Services are designed to be outwardly descriptive so that they can be found and accessed via available discovery mechanisms.*

In reality, business activities are triggered by a variety of business or operation events. Event-driven architecture (EDA) promotes the production, detection, consumption of, and reaction to events in IT systems. EDA complements SOA because services are essentially activated by triggers fired on these events. SOA and EDA architectures together provide a richer and more robust level of enterprise integration by leveraging previously unknown causal relationships to form a new event pattern in implementations [4].

A Simplified Example

To demonstrate this emerging integration pattern in enterprises, we will use the Oracle process manager, a lightweight version of SOA suite from Oracle (Figure 7.8). We will accomplish this demo by doing the following:

Step 1. Wrapping up services for integration

- Wrapping up a managing student information service for the Roster System
- Wrapping up a managing student information service for the Course System

Step 2. Creating a process using BPEL

- Using the wrapped up services to create a simple process to manage student information to make sure that two applications are synchronized (enabling a common view across the campus: the “Enrollment/Bursar Operation” Department and the “Course Registration” Department will provide an end user with the same student information)

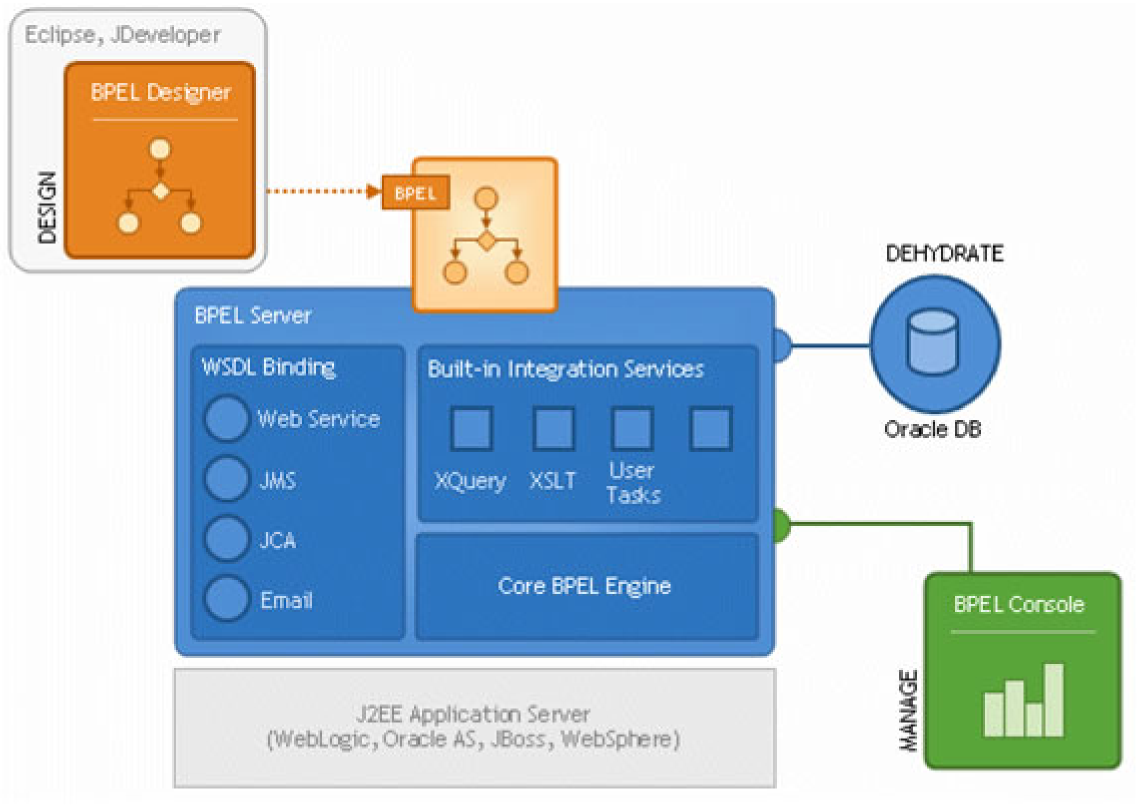
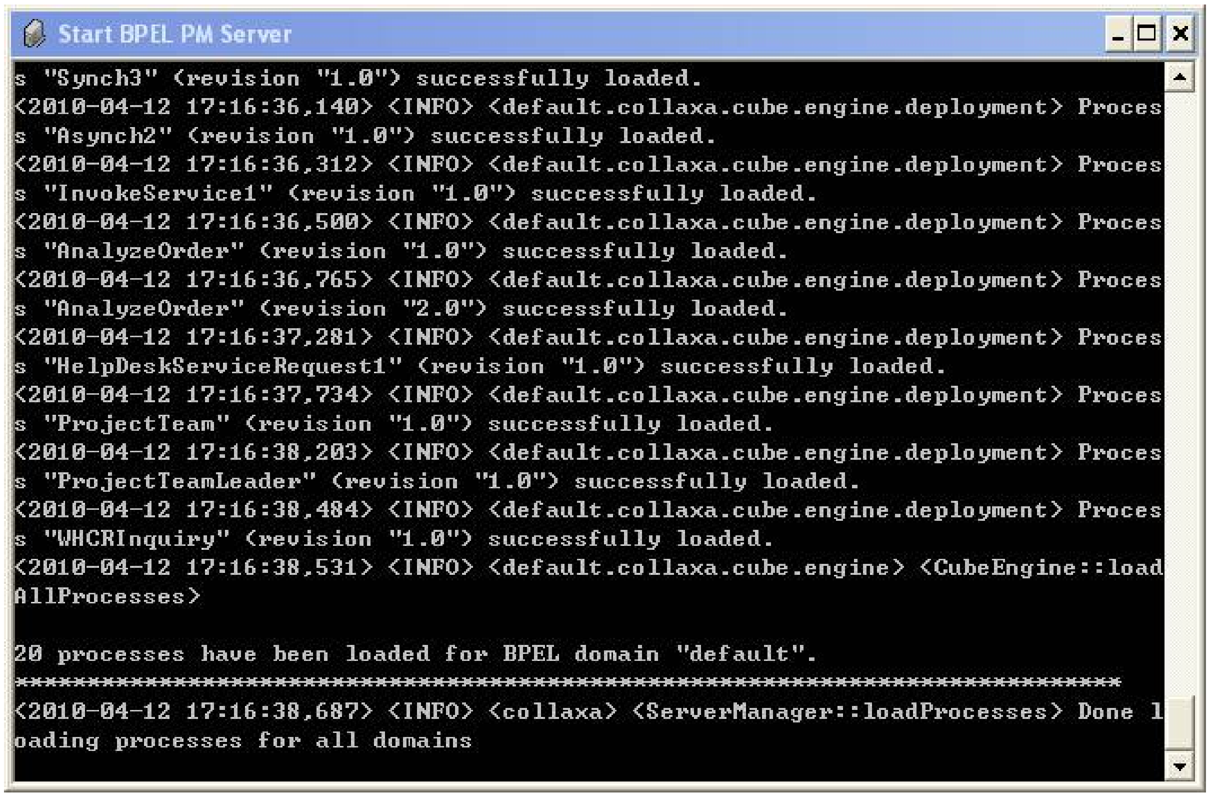


Figure 7.8 Oracle BPEL Process Manager Tools (Copyright © http://www.oracle.com/technetwork/middleware/bpel/overview/index.html (http://www.oracle.com/technetwork/middleware/bpel/overview/index.html))

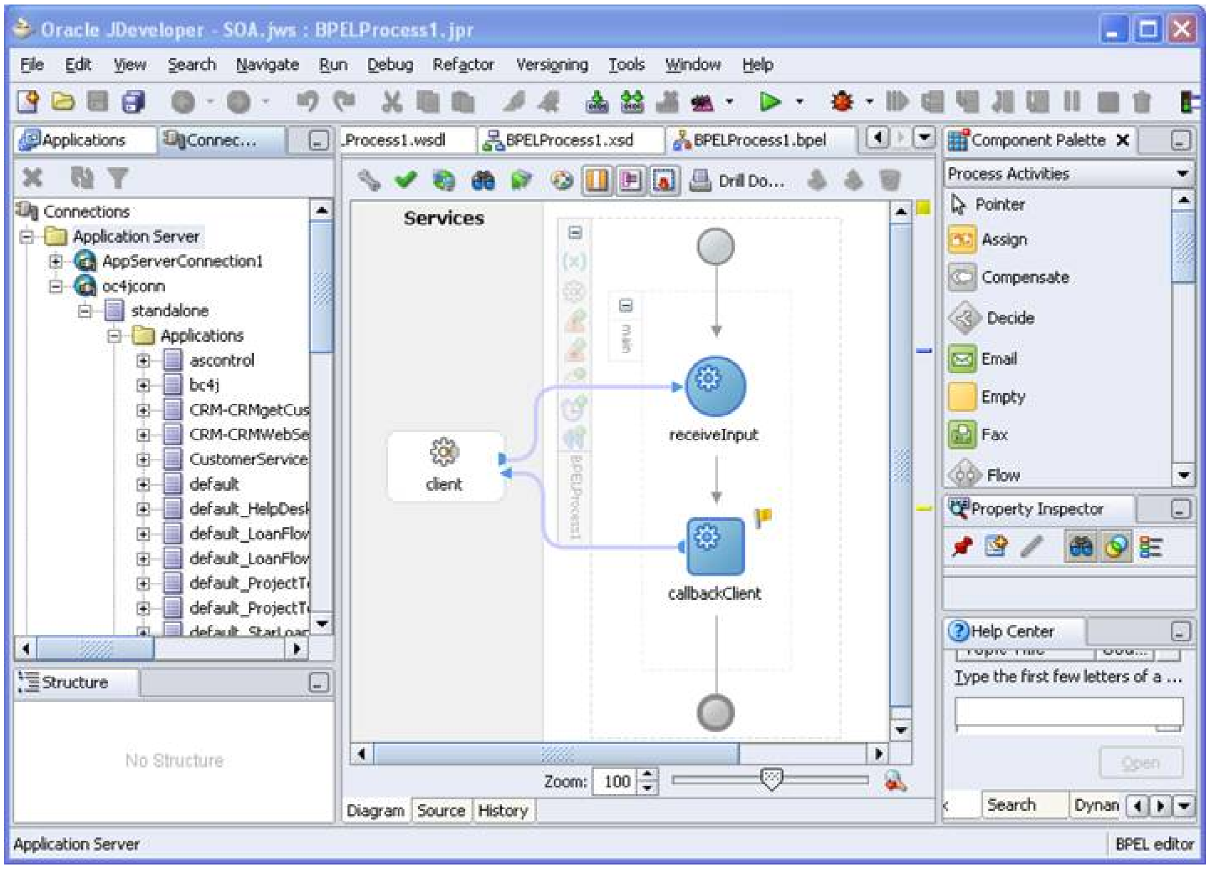
Design and Develop Bottom up Web Services using Oracle Solutions



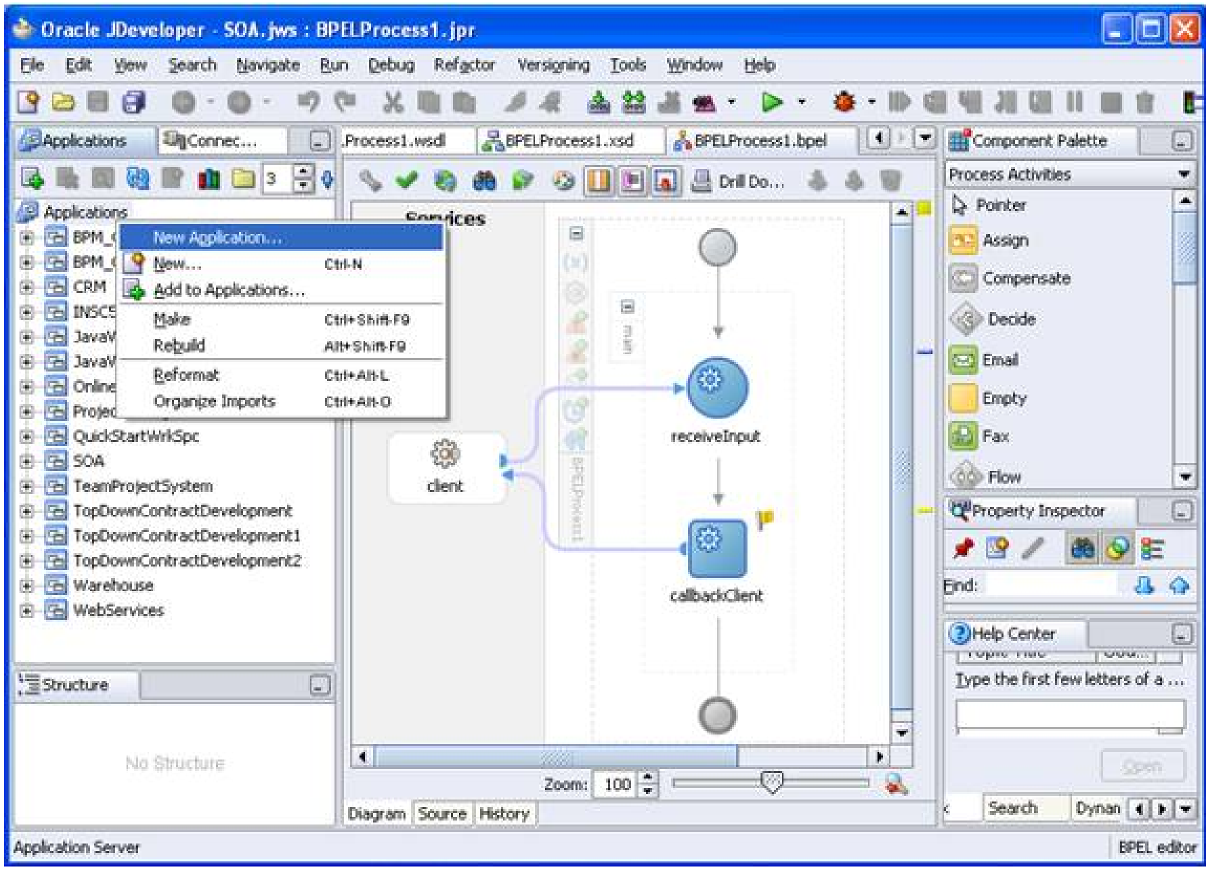
```
s "Synch3" (revision "1.0") successfully loaded.
<2010-04-12 17:16:36,140> <INFO> <default.collaxa.cube.engine.deployment> Proces
s "Asynch2" (revision "1.0") successfully loaded.
<2010-04-12 17:16:36,312> <INFO> <default.collaxa.cube.engine.deployment> Proces
s "InvokeService1" (revision "1.0") successfully loaded.
<2010-04-12 17:16:36,500> <INFO> <default.collaxa.cube.engine.deployment> Proces
s "AnalyzeOrder" (revision "1.0") successfully loaded.
<2010-04-12 17:16:36,765> <INFO> <default.collaxa.cube.engine.deployment> Proces
s "AnalyzeOrder" (revision "2.0") successfully loaded.
<2010-04-12 17:16:37,281> <INFO> <default.collaxa.cube.engine.deployment> Proces
s "HelpDeskServiceRequest1" (revision "1.0") successfully loaded.
<2010-04-12 17:16:37,734> <INFO> <default.collaxa.cube.engine.deployment> Proces
s "ProjectTeam" (revision "1.0") successfully loaded.
<2010-04-12 17:16:38,203> <INFO> <default.collaxa.cube.engine.deployment> Proces
s "ProjectTeamLeader" (revision "1.0") successfully loaded.
<2010-04-12 17:16:38,484> <INFO> <default.collaxa.cube.engine.deployment> Proces
s "WHCRInquiry" (revision "1.0") successfully loaded.
<2010-04-12 17:16:38,531> <INFO> <default.collaxa.cube.engine> <CubeEngine::load
AllProcesses>

20 processes have been loaded for BPEL domain "default".
*****************************************************************************
<2010-04-12 17:16:38,687> <INFO> <collaxa> <ServerManager::loadProcesses> Done l
oading processes for all domains
```

Step 1 :Start your BPEL PM Server

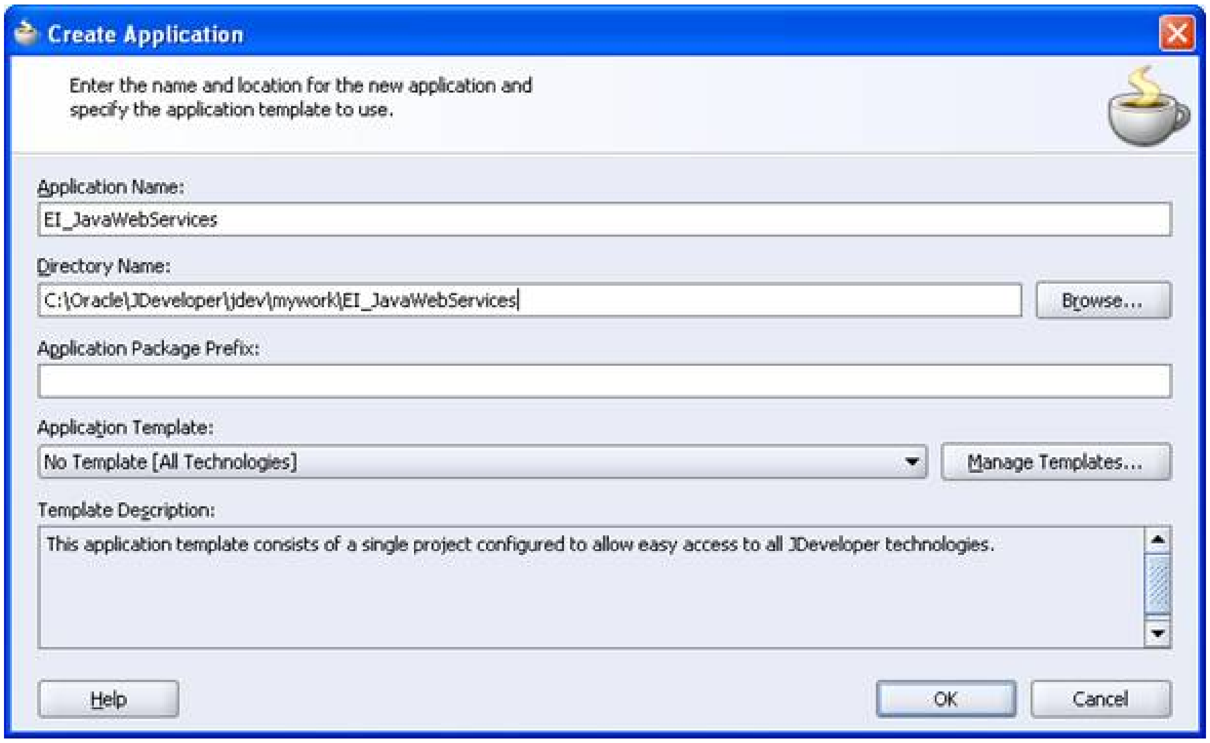
When the loading process is completed, you should see a message that indicates a list of loaded processes.



Step 2: Connect to the application server



Step 3: Create a new application



Enter the name and location for the new application and specify the application template to use.

Application Name: EI_JavaWebServices

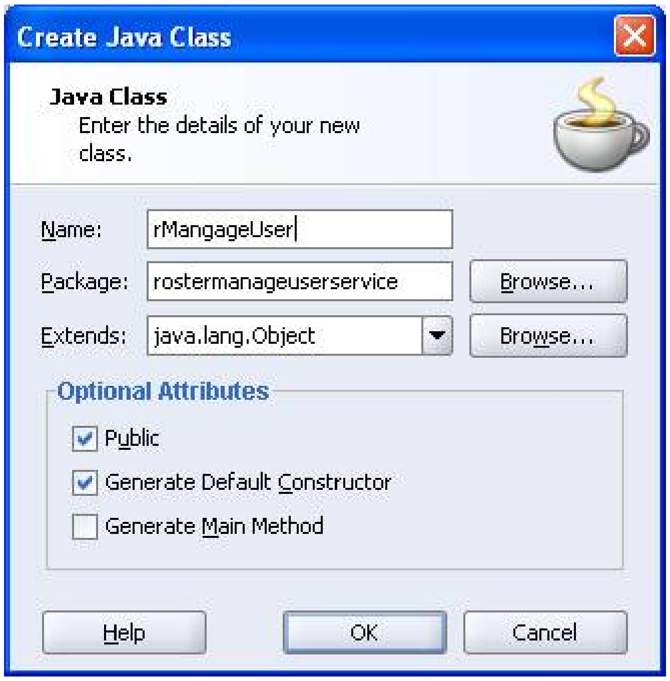
Directory Name: C:\Oracle\JDeveloper\jdev\mywork\EI_JavaWebServices

Application Package Prefix:

Application Template: No Template [All Technologies]

Template Description: This application template consists of a single project configured to allow easy access to all JDeveloper technologies.

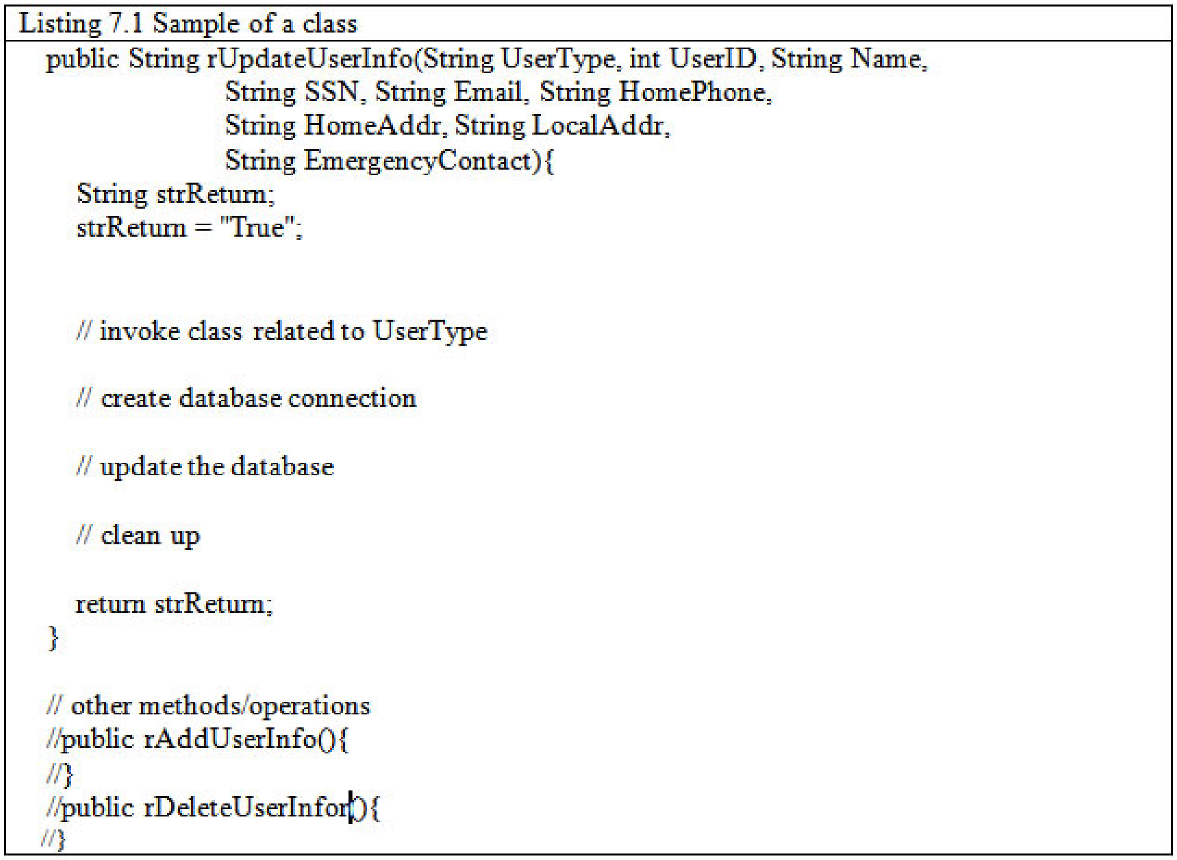
Step 4a: Create a java class



Name: rMangageUser
Package: rostermanageuserservice
Extends: java.lang.Object

Optional Attributes
- [✓] Public
- [✓] Generate Default Constructor
- [ ] Generate Main Method

Step 4b: Create java class (continued)

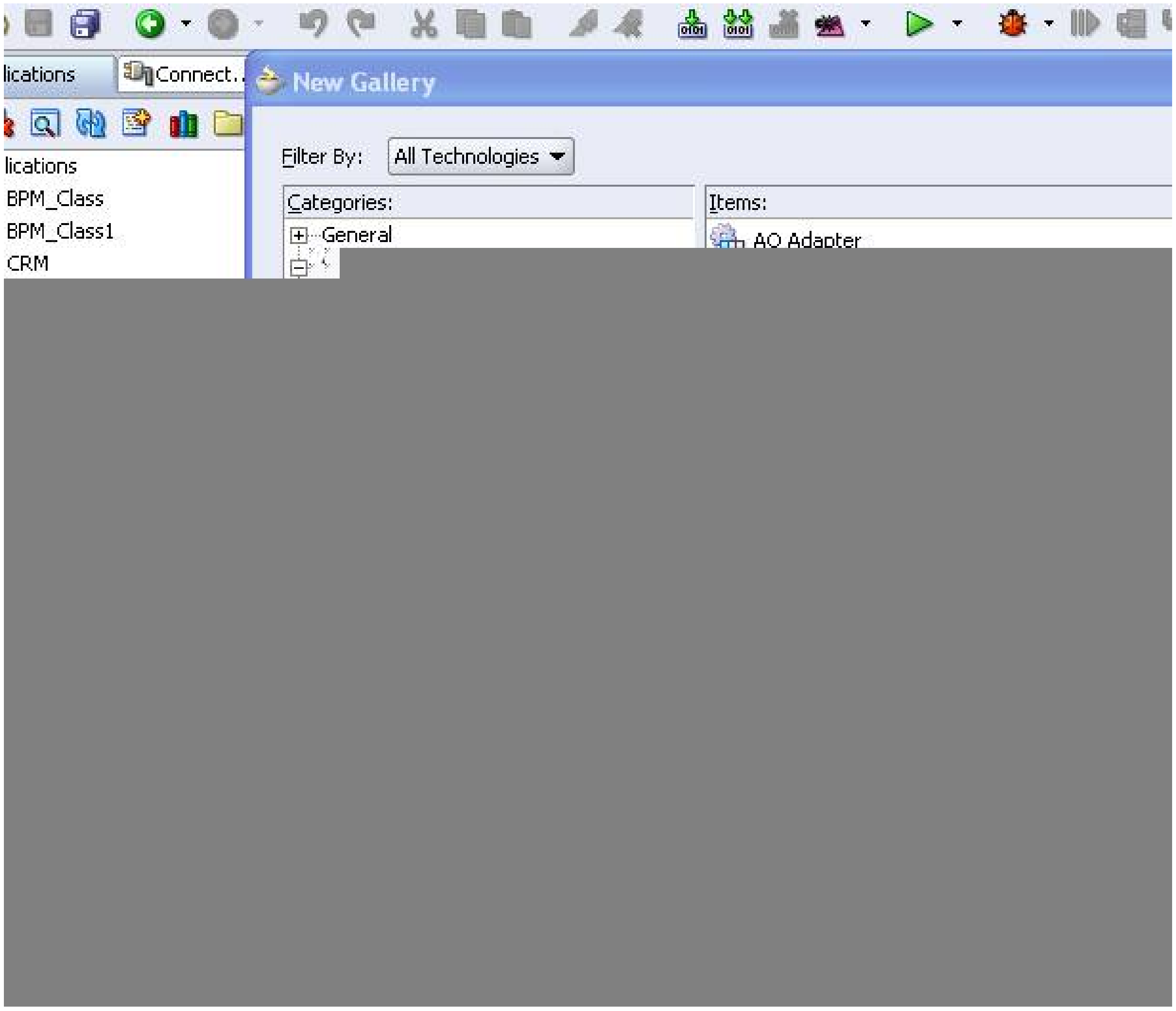


Listing 7.1 Sample of a class

```
public String rUpdateUserInfo(String UserType, int UserID, String Name,
                        String SSN, String Email, String HomePhone,
                        String HomeAddr, String LocalAddr,
                        String EmergencyContact){
    String strReturn;
    strReturn = "True";


    // invoke class related to UserType

    // create database connection

    // update the database

    // clean up

    return strReturn;
}

// other methods/operations
//public rAddUserInfo(){
//}
//public rDeleteUserInfor(){
//}
```

Step 5: Code the class

6) Compile to debug bugs, then create its Web service using the Wizard

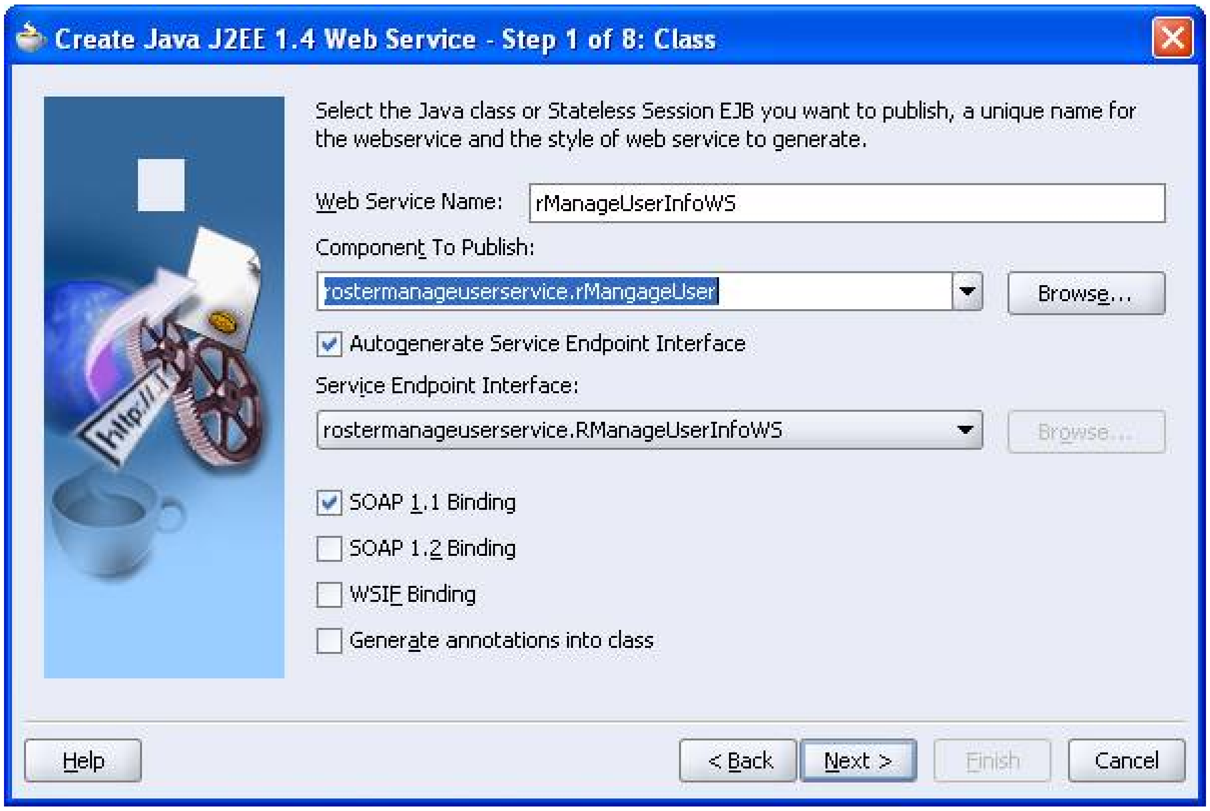


Filter By: All Technologies

Categories:
General

Items:
AQ Adapter

Step 6a


Create Java J2EE 1.4 Web Service - Step 1 of 8: Class

Select the Java class or Stateless Session EJB you want to publish, a unique name for the webservice and the style of web service to generate.

Web Service Name: rManageUserInfoWS

Component To Publish: rostermanageuserservice.rMangageUser

[✓] Autogenerate Service Endpoint Interface

Service Endpoint Interface: rostermanageuserservice.RManageUserInfoWS

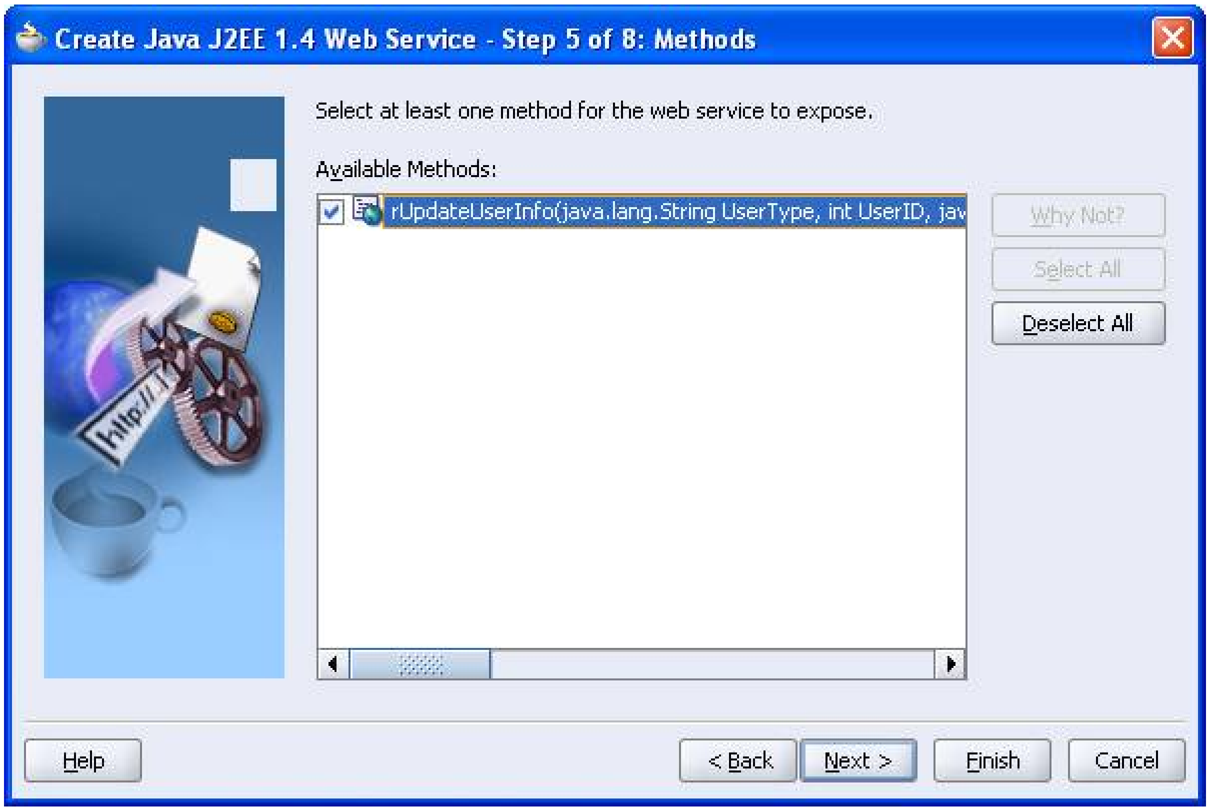
[✓] SOAP 1.1 Binding

[ ] SOAP 1.2 Binding

[ ] WSIF Binding

[ ] Generate annotations into class

Step 6b

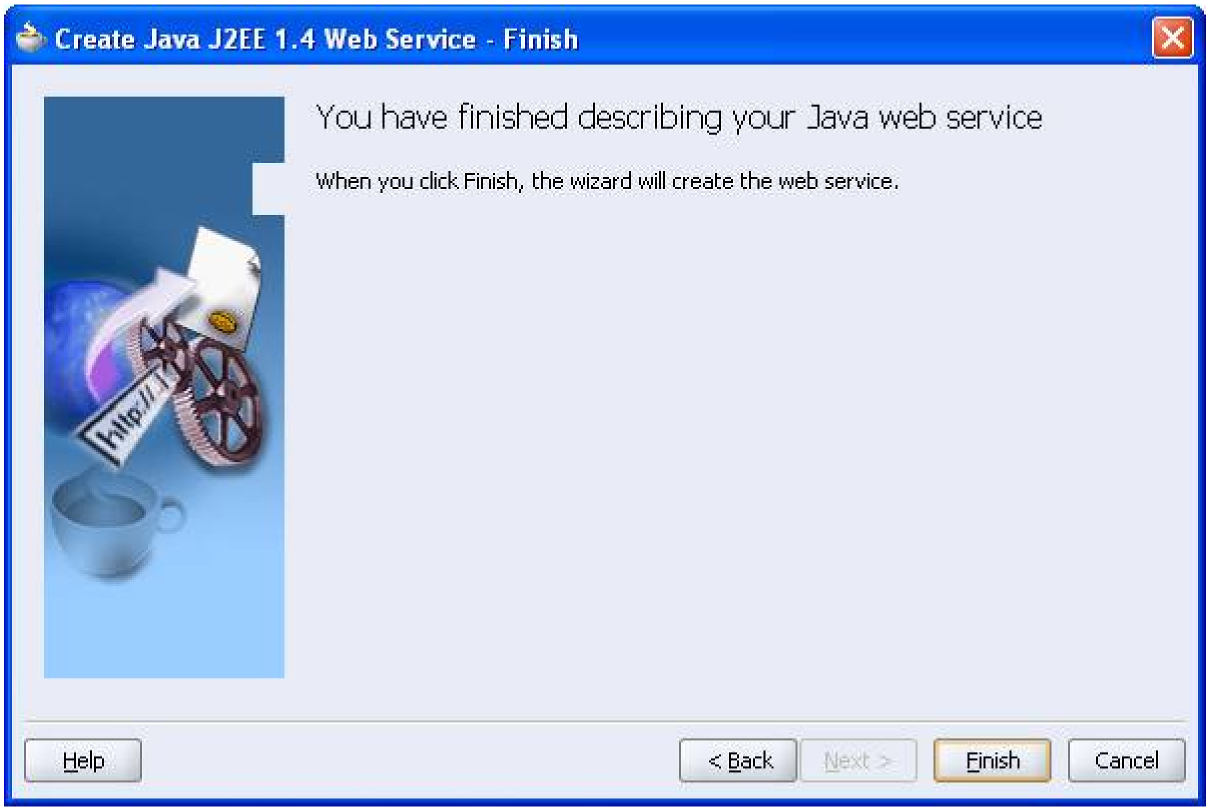

Create Java J2EE 1.4 Web Service - Step 5 of 8: Methods

Select at least one method for the web service to expose.

Available Methods:

[✓] rUpdateUserInfo(java.lang.String UserType, int UserID, jav

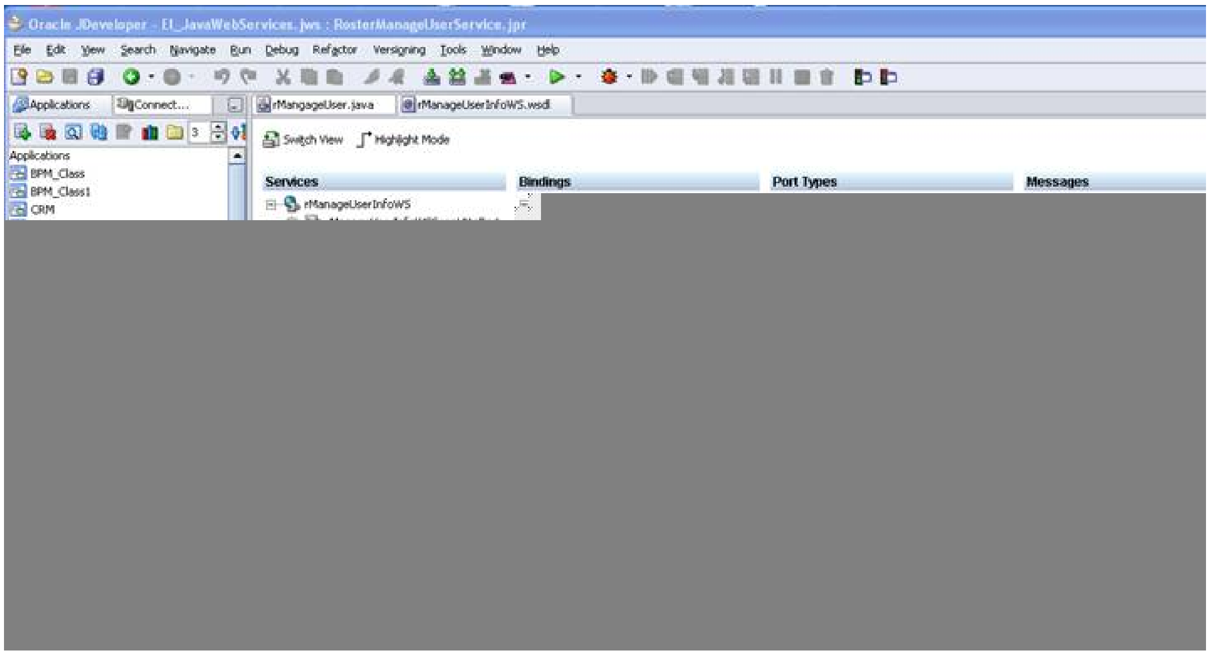
Step 6c



You have finished describing your Java web service

When you click Finish, the wizard will create the web service.

Step 6d



Step 7: WSDL is created!

Step 8: Change soap:address to match your host information in the WSDL

Step 9: Save all the files, then deploy it to your application sever

Step 10: If no error, you should be able to test it

Design and Develop Bottom Up Web Services using Microsoft Visual Studio (2005 or 2008)

Step 1: Create a new Web services project

Step 2a: Add a new class to create a Web service called “cManageUserInfo.asmx”

Step 2b: Add a new class to create a Web service called “cManageUserInfo.asmx” (continued)

Step 3: Create a cUpdateUserInfo method

Step 4:Start your IIS service

Step 5: Publish your Web services

Step 6: Create a new Virtual Directory

Step 7a: Create an Alias

Step 7b: Create an alias (continued)

Step 7c: Create an alias (continued)

Step 8: Check it out

Learn BPEL (Integration through service composition at the process level)

As the core part of BPM, Business Process Execution Language (BPEL), short for Web Services Business Process Execution Language (WS-BPEL), is an open system standard executable language. By relying on Web services technologies, BPEL is XML-based, essentially enabling standard mechanisms to define process behavior and interactions with Web Services. By supporting business process executions via exporting and importing functionality using Web Service interfaces exclusively, BPEL-based workflow processes realizes better interoperability, scalability, and adaptability [4].

The basic structure of the language is illustrated in Listing 7.3. A BPEL process specifies the exact order in which participating Web services should be invoked, either sequentially or in parallel. With BPEL, conditional behaviors can be clearly defined. For example, an invocation of a Web service can depend on the value of a previous invocation. A process can construct loops, declare variables, copy and assign values, define fault handlers, and so on. By combining all these constructs, complex business processes in an algorithmic manner can be successfully implemented.

Listing 7.3: The Basic Structure of BPEL

```
Listing 7.3 The Basic Structure of BPEL
<process name="ncname" targetNamespace="uri"
            queryLanguage="anyURI"?
            expressionlanguage="anyURI"?
            suppressJoinFailure="yes|no"?
            enableInstanceCompensation="yes|no"?
            abstractProcess="yes|no"?
            xmlns="http://schemas.xmlsoap.org/ws/2003/03/business-process/">

    <partnerLinks>?
        < ! --Note: At least one role must be specified. -->
        <partnerLink name="ncname" partnerlinkType="qname"
            myRole="ncname"? partnerRole="ncname"?>+
        </partnerLink>
    </partnerLinks>

    <variables>?
        <variable name="ncname" messageType="qname"?
            type="qname"? element="qname"?/>+
    </variables>

    <faultHandlers>?
        < ! --Note: There must b e at least one fault handler or default. -->
        <catch faultName="qname"? faultVariable="ncname"?>*
            activity...
        </catch>
        <catchAll>?
            activity...
        </catchAll>
    </faultHandlers>

    <compensationHandler>?
        activity...
    </compensationHandler>



    <eventHandlers>?
        < ! --Note: There must be at least one onMessage or onAlarm handler. -->
        <onMessage partnerLink="ncname" portType="qname"
            operation="ncname" variable="ncname"?>
            activity...
        </onMessage>

        <onAlarm for="duration-expr"? until="deadline-expr"?>*
            activity...
        </onAlarm>
    </eventHandlers>

   activity...

</process>
```

Step 1: Create a new application

Caption

Caption

Caption

Caption

Caption

Caption

Caption

Caption

Caption

Caption

Caption

Caption

Caption

Caption

Caption

A Realistic Approach to Enterprise Integration

By integrating fine-grained and coarse-grained software modules as needed from bottom to top, we could integrate applications gradually using different methods practically applicable to different circumstances:

- At the data source level, when applicable, applications can share their data using data-level integration method. Certain middleware or ESB can be used if technically available and proven cost-effectively.
- At the application level, when possible and justifiable, method-level integration can be used by relying on the support of middleware or ESB. It typically would make more sense to use API-level integration approach as minimum changes are needed within applications.
  ---a) Whenever possible, Web services should be used to enable open source and standardized interfaces for better interoperability.
  ---b)Web services based IT services will also lay the necessary foundation for the adoption of SOA/BPM across an enterprise.
- By relying on a variety of integration modules at both the data source and the application level, applications could be further integrated using workflow management systems to better

support certain well-defined rule-based (e.g., best practices) business services within specific business function domain or across business function domains.

- Process-level integrations can be implemented using proprietary workflow management systems if necessary. However, Web services based interfaces should be provided for external invocations.
- Optimally, open systems and standard workflow management systems are used for this process-level integration.
- By relying on the emerging ESBs and with the support of service-oriented architecture (SOA) and event-driven architecture (EDA), an enterprise can integrate applications at the process level. As discussed earlier, by supporting business process executions via exporting and importing functionality using Web Service interfaces exclusively, BPEL-based workflow processes realizes better interoperability, scalability, and adaptability.

References (9 of 9)

References

[1] Roshen, W. 2009. SOA-based Enterprise Integration: A Step-by-Step Guide to Services-based Application Integration. McGraw-Hill, New York, USA.

[2] WFMC: www.wfmc.org.

[3] Qiu, R. Information Technology as a Service. Enterprise Service Computing: From Concept to Deployment. (Chapter 1) Ed. By R. Qiu. IGI Publishing, Hershey, PA.

[4] Wikipedia:

- BPM: http://en.wikipedia.org/wiki/Business_process_management,
- BPDM: http://en.wikipedia.org/wiki/Business_Process_Definition_Metamodel,
- BPMN: http://en.wikipedia.org/wiki/BPMN,
- BEPL: http://en.wikipedia.org/wiki/Business_Process_Execution_Language,
- SOA: http://en.wikipedia.org/wiki/Service_oriented_architecture,
- EDA: http://en.wikipedia.org/wiki/Event-driven_architecture

[5] Oracle Process Manager: Handouts>Oracle Packages

[6] JDeveloper: Handouts>Oracle Packages

[7] Juric, Matjaz B., 1) A Hands-on Introduction to BPEL, 2) A Hands-on Introduction to BPEL, Part 2: Advanced BPEL: Handouts>Week7