



IBM Software Group

# SOA 开发第三步 – 构建服务组件

IBM Rational 姚炳雄 yaobx@cn.ibm.com



@business on demand.

# SOA 开发三步曲

 **第一步** 业务分析员

- 业务目标
- 业务模型
- 业务需求

- 分析并记录现有的业务流程及其概念
- 优化改进业务流程

**Rational. software**  
Rational Requisite Pro  
Rational Software Modeler

 **第二步** 架构设计师

- 服务模型
- 软件架构
- 企业架构

- 确定服务
- 构建服务模型
- 搭建系统架构

**Rational. software**  
Rational Software Architect

 集成开发员

- 服务流程模型
- 服务组装模型

- 实现业务流程和组合应用
- 定义服务

**WebSphere software**  
Websphere Integration Developer

 **第三步** 开发人员

- 设计模型
- 实现模型

- 设计服务组件
- 编码实现服务

**Rational. software**  
Rational Application Developer

资源组合   共享资产   公共流程

**Rational. software**   Rational Portfolio Manager   Rational ClearCase  
Rational Unified Process   Rational ClearQuest



# 内容

- 快速构建Web Service
- 快速构建SDO
- 快速构建JSF



## 构建服务组件 - SOA开发第三步

- 快速构建Web Service
- 快速构建SDO
- 快速构建JSF



# 快速构建Web Service — Web Service简介

## SOAP

- 交换信息的协议
- 多平台支持的xml协议，用于不同平台的client / Server之间通信，

## WSDL - Web Services Description Language

- 基于xml的描述的服务接口。

## UDDI - Universal Description, Discovery, Integration

- UDDI 服务提供的列表信息
- 使用 SOAP 应用 更新或查询服务列表信息，以取得要访问的服务信息



# 快速构建Web Service—Web Service简介

## JAX-RPC

- 基于XML的RPC的API
- 从Servlet中访问WebService的API
- 客户端访问Web Services的相关API
- 如何将JAVA类型映射成WSDL中的XML

## JSR-109

- 如何将J2EE应用打包，部署成WebService.
- 无状态Session Bean如何转换到WebService.



# 快速构建Web Service—RAD的开发能力

## ■ 开发

- Bottom Up – 将已经存在的 JavaBeans, EJBs, DB2进行Web Service封装
- Top Down – 从WSDL自动生成JAVA调用的桩代码

## ■ 测试

- 使用WSDL explorer 进行Web Service执行测试

## ■ 部署

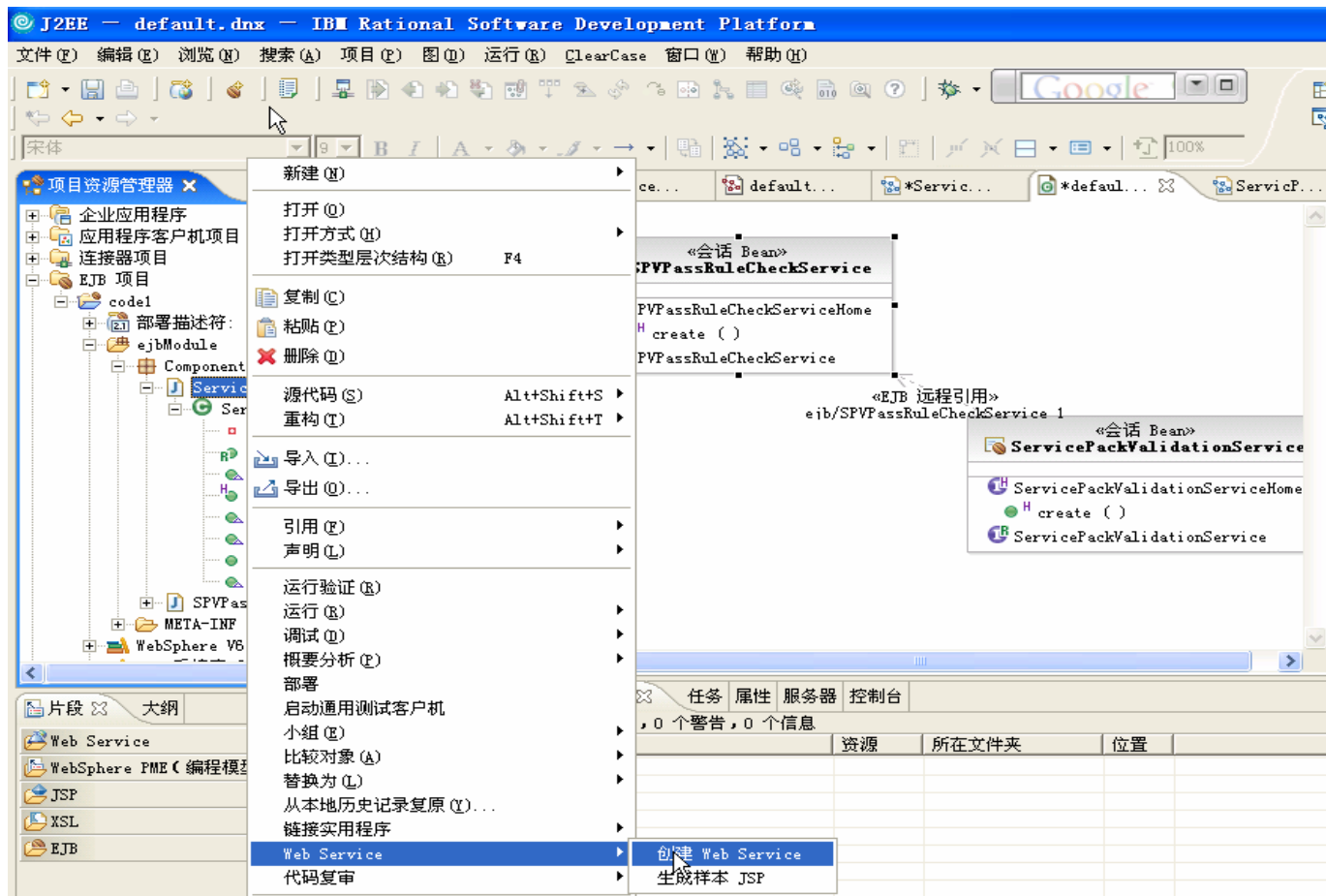
- 将Web Service 部署到WebSphere® 应用服务器 或者 Tomcat 环境

## ■ 发布

- 将Web services 发布到 UDDI v2 or v3 注册服务器上

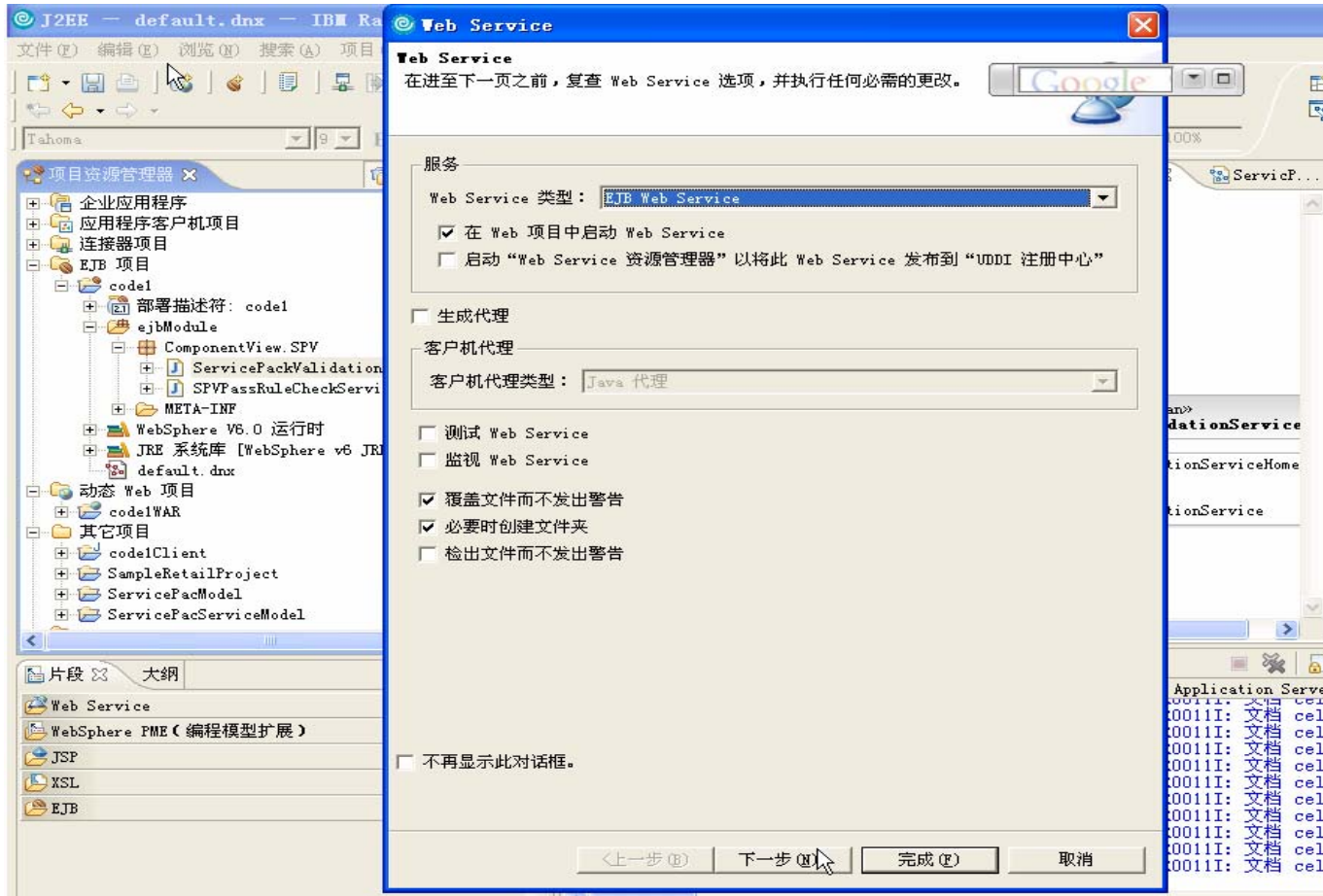


# Demo8 - Bottom Up – 从EJB创建Web Service

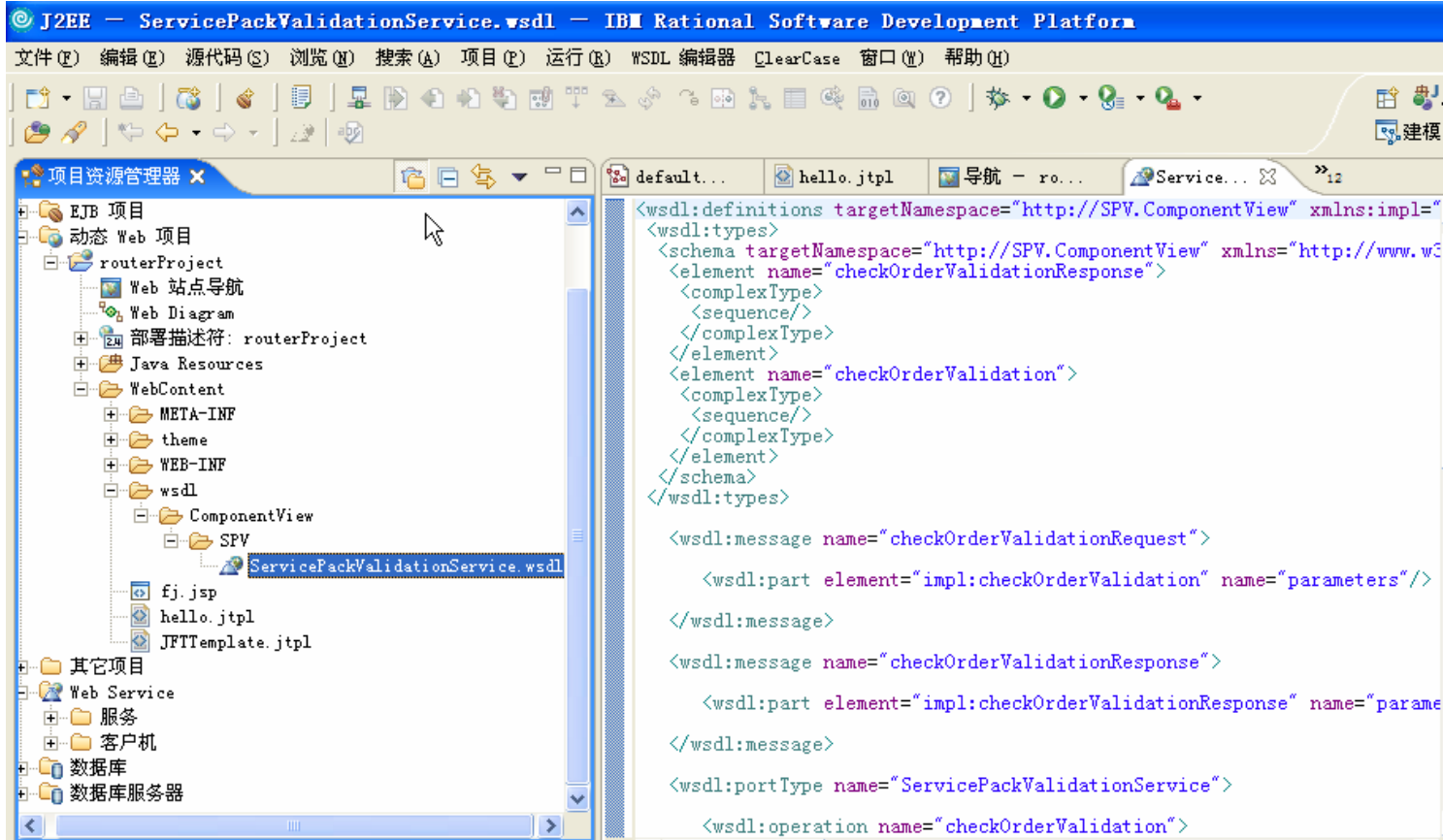




# Bottom Up – 从EJB创建Web Service



# Bottom Up – 从EJB创建Web Service



The screenshot displays the IBM Rational Software Development Platform interface. The title bar indicates the current file is `ServicePackValidationService.wsdl`. The menu bar includes options like 文件(F), 编辑(E), 源代码(S), 浏览(V), 搜索(A), 项目(P), 运行(R), WSDL 编辑器, ClearCase, 窗口(W), and 帮助(H). The toolbar contains various icons for file operations and development tools.

The **项目资源管理器** (Project Explorer) on the left shows a project structure under **EJB 项目** (EJB Project) > **动态 Web 项目** (Dynamic Web Project) > **routerProject**. The **wsdl** folder is expanded, showing `ServicePackValidationService.wsdl` selected. Other files include `fj.jsp`, `hello.jtpl`, and `JFTTemplate.jtpl`.

The main editor window displays the WSDL XML code for `ServicePackValidationService`. The code defines two messages: `checkOrderValidationRequest` and `checkOrderValidationResponse`, and a port type `ServicePackValidationService` with an operation `checkOrderValidation`.

```
<?xml version="1.0" encoding="UTF-8" ?>
<wsdl:definitions targetNamespace="http://SPV.ComponentView" xmlns:impl="http://www.ibm.com/xml/SPV/ComponentView" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <schema targetNamespace="http://SPV.ComponentView" xmlns="http://www.w3.org/2001/XMLSchema" >
      <element name="checkOrderValidationResponse">
        <complexType>
          <sequence/>
        </complexType>
      </element>
      <element name="checkOrderValidation">
        <complexType>
          <sequence/>
        </complexType>
      </element>
    </schema>
  </wsdl:types>

  <wsdl:message name="checkOrderValidationRequest">
    <wsdl:part element="impl:checkOrderValidation" name="parameters"/>
  </wsdl:message>

  <wsdl:message name="checkOrderValidationResponse">
    <wsdl:part element="impl:checkOrderValidationResponse" name="parameters"/>
  </wsdl:message>

  <wsdl:portType name="ServicePackValidationService">
    <wsdl:operation name="checkOrderValidation">
      <input message="checkOrderValidationRequest" type="text"/>
      <output message="checkOrderValidationResponse" type="text"/>
    </wsdl:operation>
  </wsdl:portType>
</wsdl:definitions>
```

# Top Down –从WSDL自动生成JAVA调用的代码

The screenshot shows the IBM WebSphere IDE interface. On the left, the 'Project Explorer' (项目资源管理器) displays a project structure under '动态 Web 项目' (Dynamic Web Project) > 'routerProject' > 'WebContent' > 'wsdl' > 'ComponentView' > 'SPV' > 'ServicePackValidationService.wsdl'. A context menu is open over this file, listing various actions. The option '生成 Java bean 框架' (Generate Java Bean Framework) is circled in red. The main editor window shows the XML content of the WSDL file, including namespace definitions and message declarations for 'checkOrderValidationRequest' and 'checkOrderValidationResponse'. At the bottom, a 'Properties' (属性) window is visible, showing details for the selected WSDL file.

**Context Menu Options:**

- 新建 (N)
- 打开 (O)
- 打开方式 (O)
- 复制 (C)
- 粘贴 (P)
- 删除 (D)
- 重构 (R) Alt+Shift+T
- 导入 (I)...
- 导出 (E)...
- 刷新 (F)
- 运行验证 (V)
- 验证 WSDL 文件
- 运行 (R)
- 调试 (D)
- 概要分析 (A)
- 部署
- 小组 (G)
- 比较对象 (A)
- 替换为 (L)
- 链接实用程序
- Web Service
- RAS
- 源代码 (S)
- 属性 (R) Alt+Enter

**WSDL Content:**

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://SPV.ComponentView" xmlns:impl="http://SPV.C
<wsdl:types>
<schema targetNamespace="http://SPV.ComponentView" xmlns="http://www.w3.org/2001/X
<element name="checkOrderValidationResponse">
<complexType>
<sequence/>
</complexType>
</element>
</wsdl:types>
<message name="checkOrderValidationRequest">
<part element="impl:checkOrderValidation" name="parameters"/>
</message>
<message name="checkOrderValidationResponse">
<part element="impl:checkOrderValidationResponse" name="parameters"/>
</message>
</wsdl:definitions>
```

**Properties Window:**

属性	值
名称	ServicePackValidationService.wsdl
位置	/routerProject/WebContent/wsdl/ComponentView/SPV/ServicePackValidationService.wsdl
大小	2591
日期	true
是否 WSDL 文件	false
是否 WSDL 文件	false
是否 WSDL 文件	false

# 使用WSDL 浏览器进行Web Service执行测试

The screenshot displays the IBM WebSphere IDE interface. On the left, the 'Project Explorer' shows a project structure including 'routerProject' and 'WebContent'. The main editor area shows an XML WSDL file with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<wSDL:definitions targetNamespace="http://SPV.ComponentView" xmlns:impl="http://SPV.C
<wSDL:types>
<schema targetNamespace="http://SPV.ComponentView" xmlns="http://www.w3.org/2001/X
<element name="checkOrderValidationResponse">
<complexType>
<sequence/>
</complexType>
</element>
<complexType>
<sequence/>
</complexType>
</element>
</types>
</wSDL:types>
<message name="checkOrderValidationRequest">
</message>
<message name="checkOrderValidationResponse">
</message>
</wSDL:messages>
<binding name="impl:checkOrderValidation" type="impl:checkOrderValidationRequest" />
<binding name="impl:checkOrderValidationResponse" type="impl:checkOrderValidationResponse" />
</wSDL:bindings>
</definitions>
```

A context menu is open over the WSDL file, with the following options:

- 新建 (N)
- 打开 (O)
- 打开方式 (O')
- 复制 (C)
- 粘贴 (V)
- 删除 (D)
- 重构 (R) Alt+Shift+T
- 导入 (I)...
- 导出 (E)...
- 刷新 (F5)
- 运行验证 (Q)
- 验证 WSDL 文件
- 运行 (R)
- 调试 (D)
- 概要分析 (A)
- 部署
- 小组 (G)
- 比较对象 (A)
- 替换为 (L)
- 链接实用程序
- Web Service
- RAS
- 源代码 (S)
- 属性 (R) Alt+Enter

The 'Web Service' submenu is open, showing the following options:

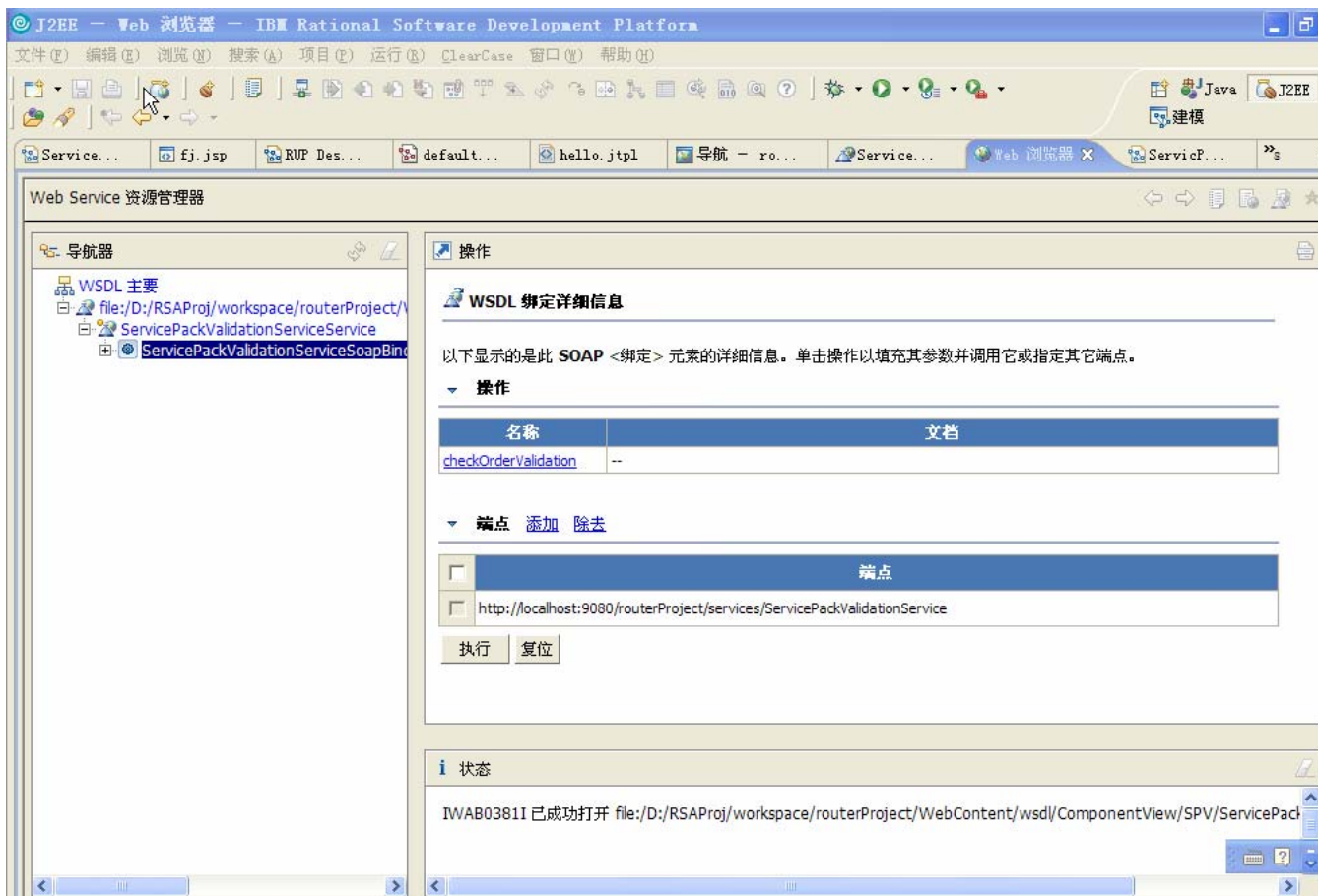
- 使用 Web Service 资源管理器测试 (highlighted with a red circle)
- 发布 WSDL 文件
- 生成 Java bean 框架
- 生成客户机
- 生成 WSIL

At the bottom right, a table shows test results:

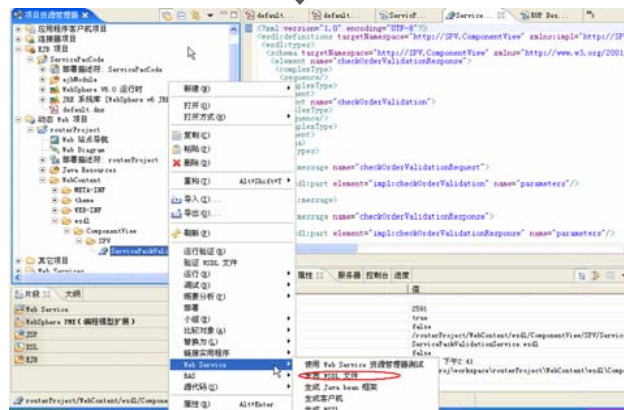
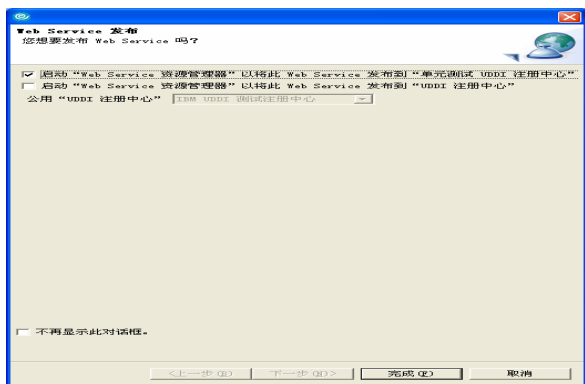
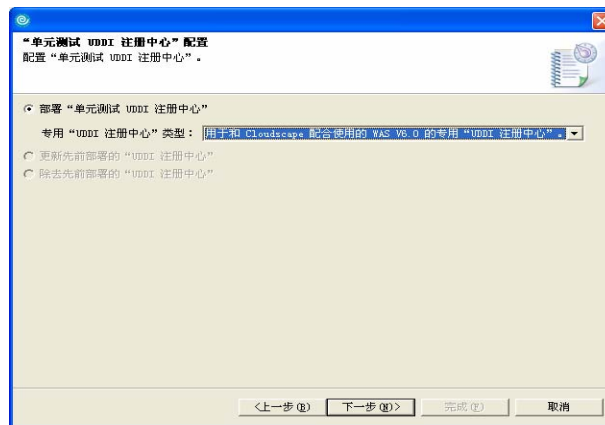
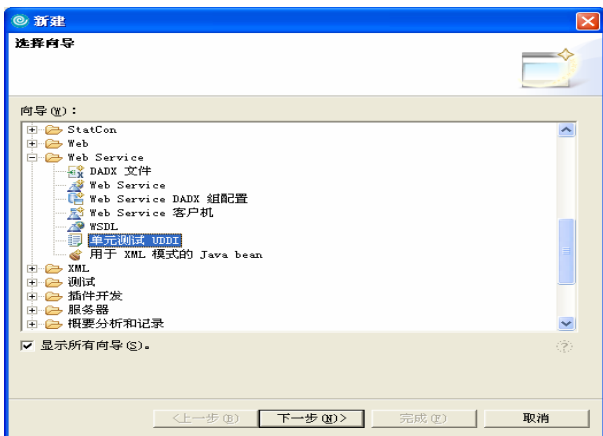
属性	值
	2591
	true
	false
	/routerProject/WebContent/wsd/ComponentView/SPV/ServicePac
	ServicePackValidationService.wsdl
	false

The status bar at the bottom shows the time as 下午2:41 and the file path: routerProject\workspace\routerProject\WebContent\wsdl\Compone...

# 使用WSDL 浏览器进行Web Service执行测试



# 创建UDDI注册器并发布WSDL



# 构建服务组件 - SOA开发第三步

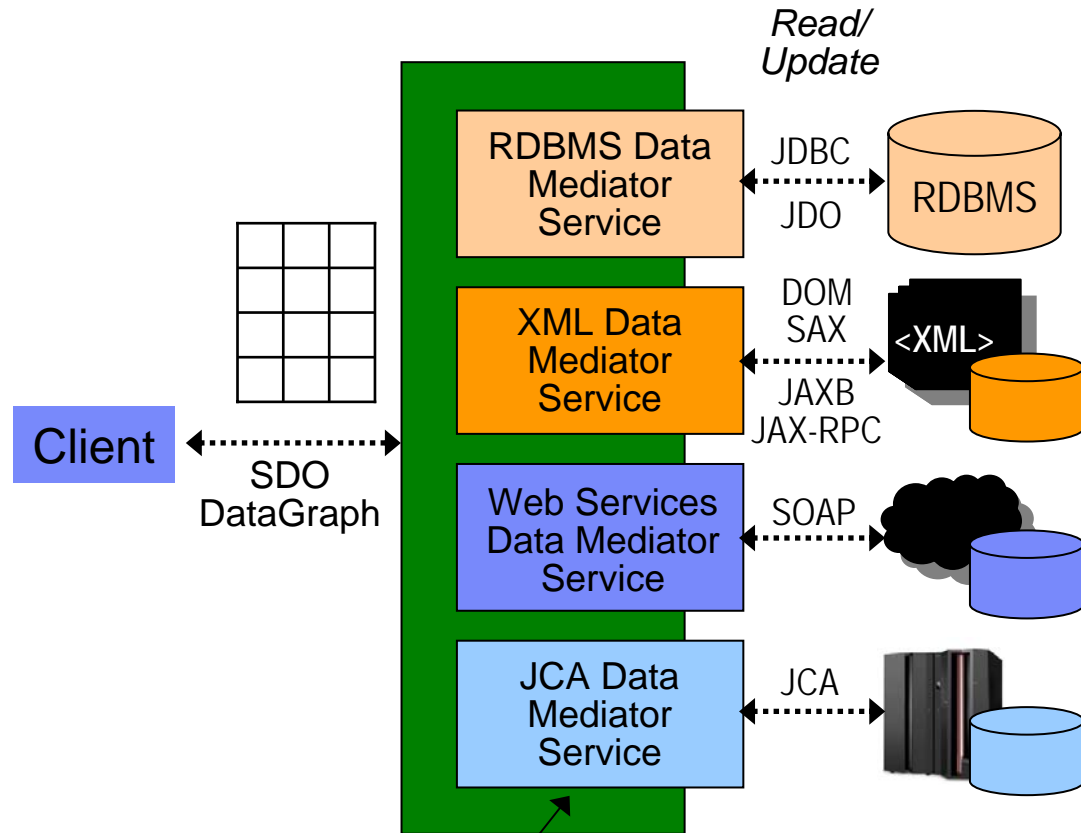
- 快速构建Web Service
- 快速构建SDO
- 快速构建JSF



Emerging Standard

# Service Data Objects (SDO): 统一的访问异构数据源

- 提供了一种通用的编程接口，用来访问异构的数据源
- 同时支持静态和动态的数据API
- 支持离线操作
- 将逻辑代码和数据访问代码剥离
- 用于提高程序员的开发效率



*Query data sources, create data graphs containing data objects, apply changes back to the data source*

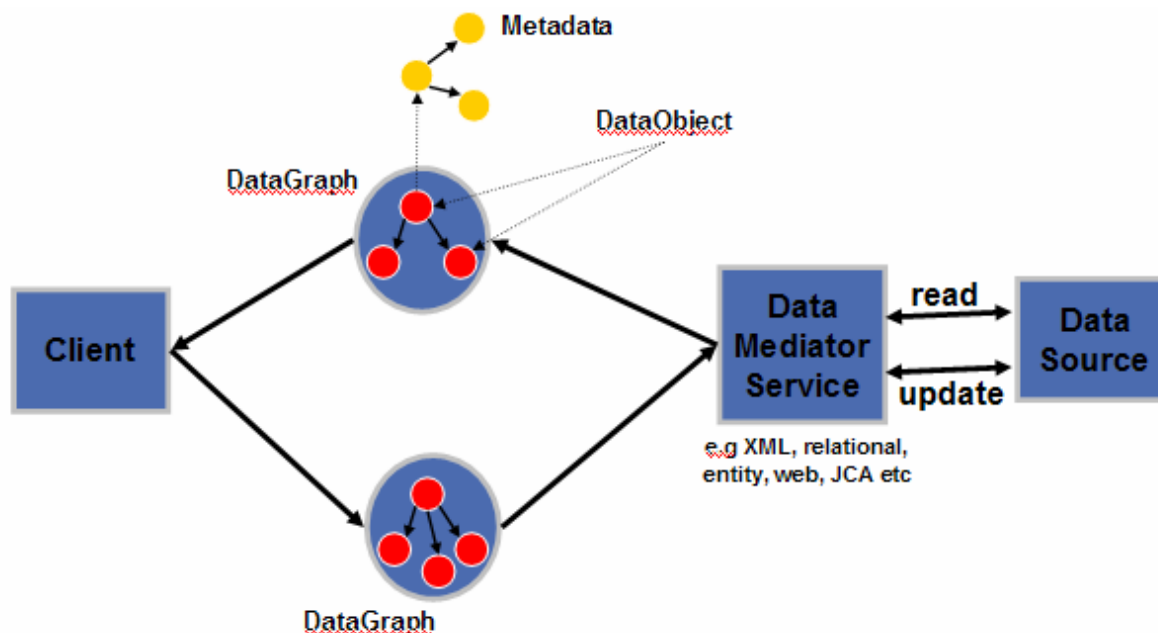
Service Data Objects (SDO) is a specification created by IBM & BEA, submitted to JCP as JSR 235 in December 2003. Expert Group Formation Stage, not part of J2EE yet



# SDO的好处

SDO带来的好处:

- 统一数据应用开发
- 简化J2EE应用层对数据层的访问
- 支持和集成各种数据源
- J2EE pattern和最佳实践的体现



# 在RAD中构建SDO的两种方式

- 通过JSF
- 通过Entity Bean

# 在RAD中构建SDO的两种方式

- 通过JSF
- 通过Entity Bean

# 进行JSF/SDO的开发

Web - customerlist.jsp - IBM Rational Software Development Platform

File Edit Toolbar Insert JSP Format Table Frame Page Tools Navigate Search Project Run Window Help

Project... MyStore

Enterprise Application  
Application Client Proj  
Connector Projects  
EJB Projects  
Dynamic Web Project  
MyStore  
Web Site Navig  
Deployment De  
Java Resources  
WebContent  
crystalrepor  
META-INF  
theme  
WEB-INF

customerlist.jsp - customerlist.jsp

h:outputText Standard

MyStore.com

设计界面

Id <sup>abc</sup>	Customer <sup>abc</sup>	Email <sup>abc</sup>	Homephone <sup>abc</sup>	Photo <sup>abc</sup>
{ID}123	Firstname: {FIRSTNAME}abc Lastname: {LASTNAME}abc City: {CITY}abc	{EMAIL}abc	{HOMEPHONE}abc	{img}

Design Source Preview

Properties Quick Edit Servers Problems

table Id: text10 Format

tbody Value: #{varcustomer.LASTNAME}

tr Style: Props:

td abc h:outputText Classes: outputText

数据对象

属性编辑器

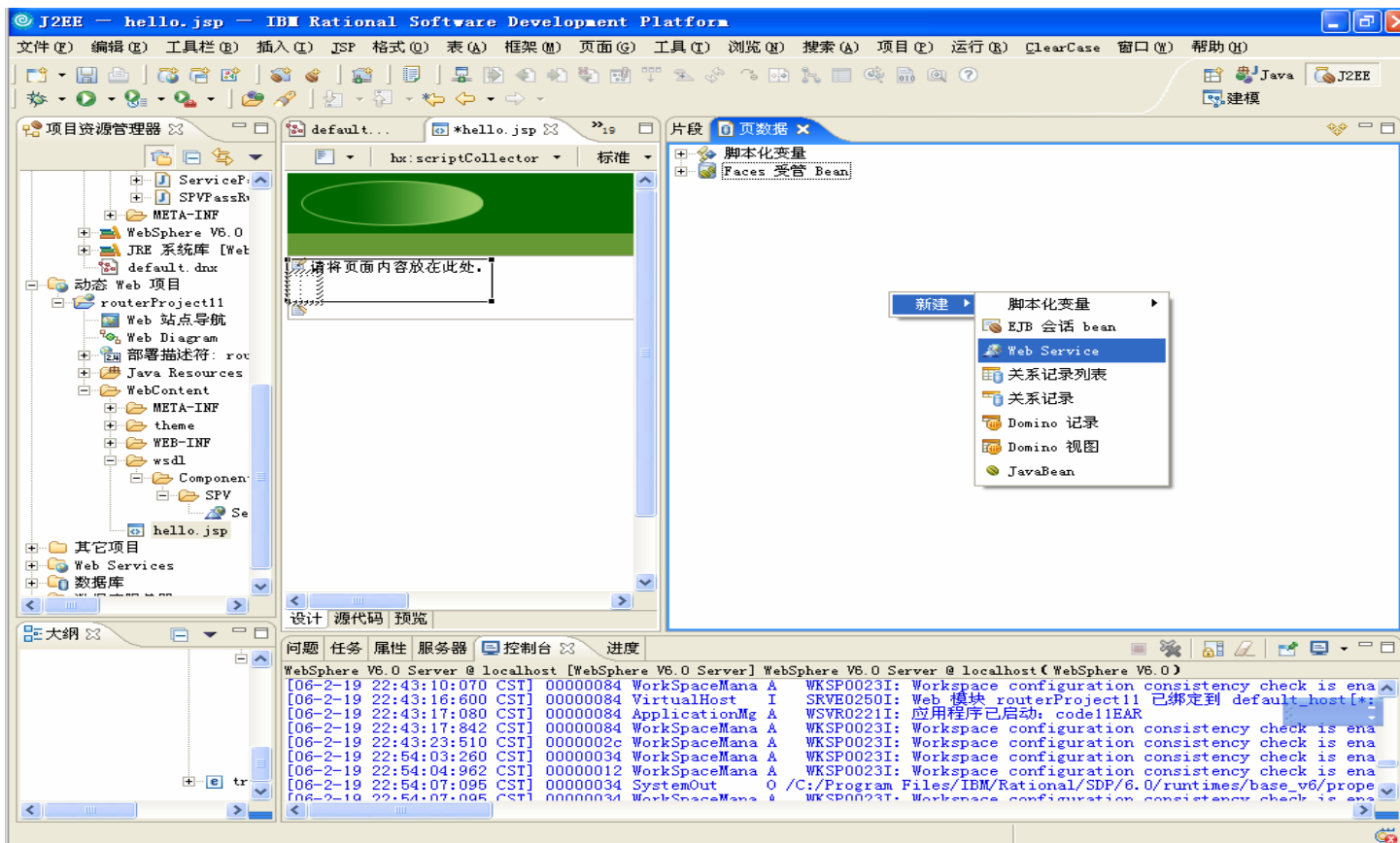
页面数据

面板

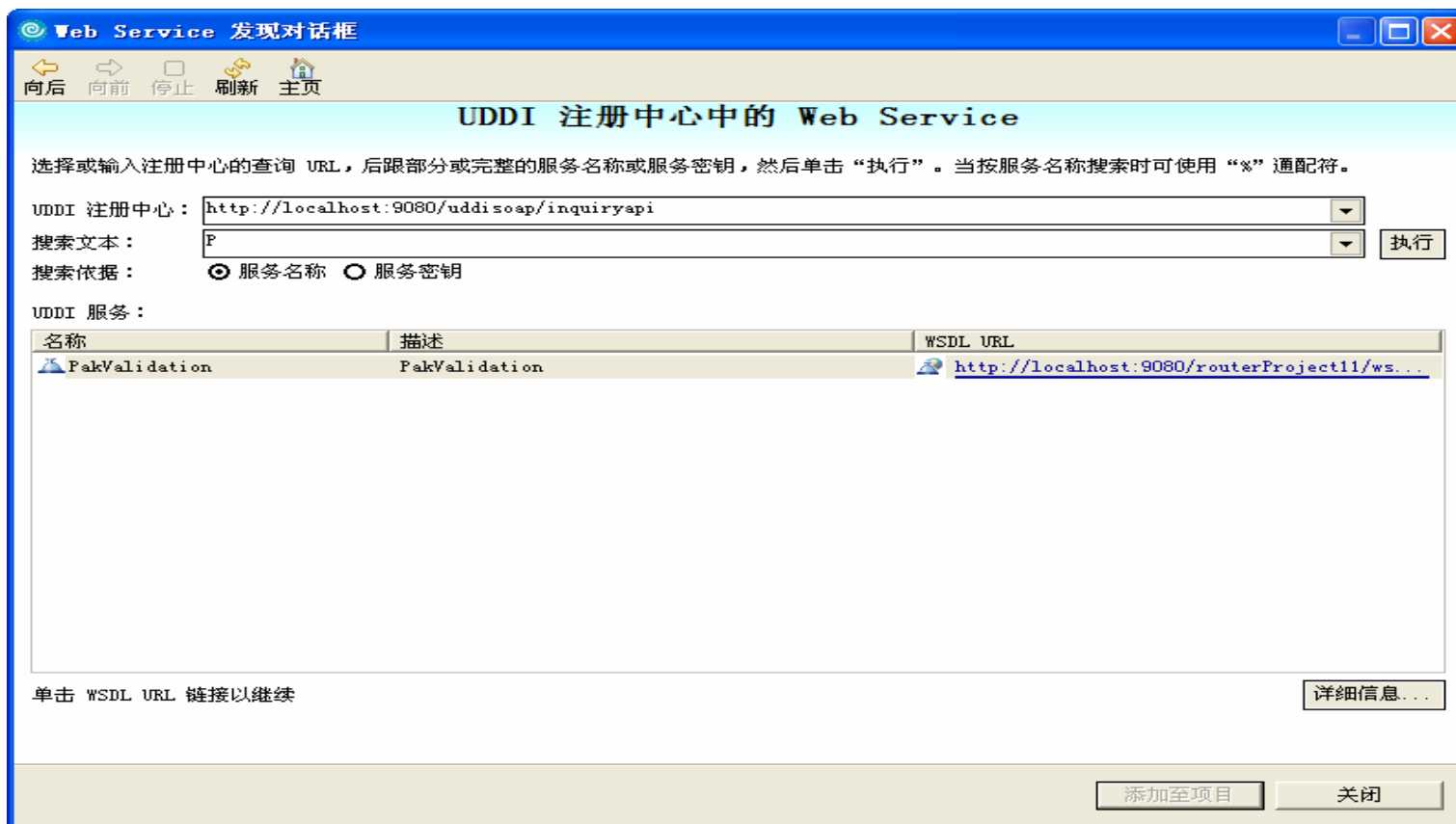
- HTML Tags
- JSP Tags
- Crystal Reports/Fa...
- Faces Components
- Data Table
- Panel - Group Box
- Panels - Tabbed
- Panel - Menu Bar
- Command - Button
- Command - Hyperlink
- Page Template
- Web Site Navigation
- Data
- Web Service
- EJB Session bean
- Relational Record
- Relational Record List
- JavaBean
- SAP BAPI
- SAP RFI

属性编辑器

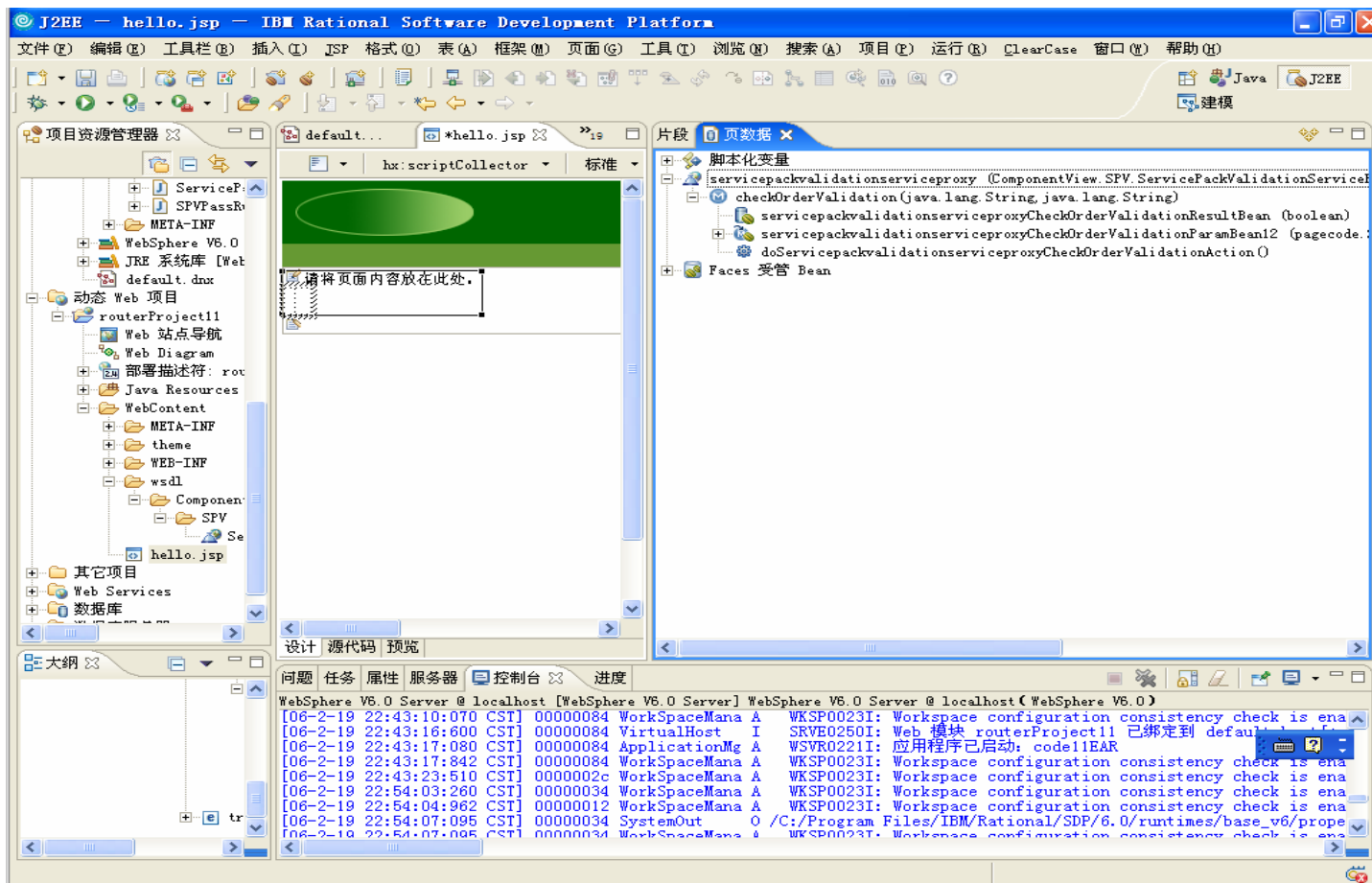
# Demo9-使用SDO连接Web Service



# 使用SDO连接Web Service



# 使用SDO连接Web Service



# 在RAD中构建SDO的两种方式

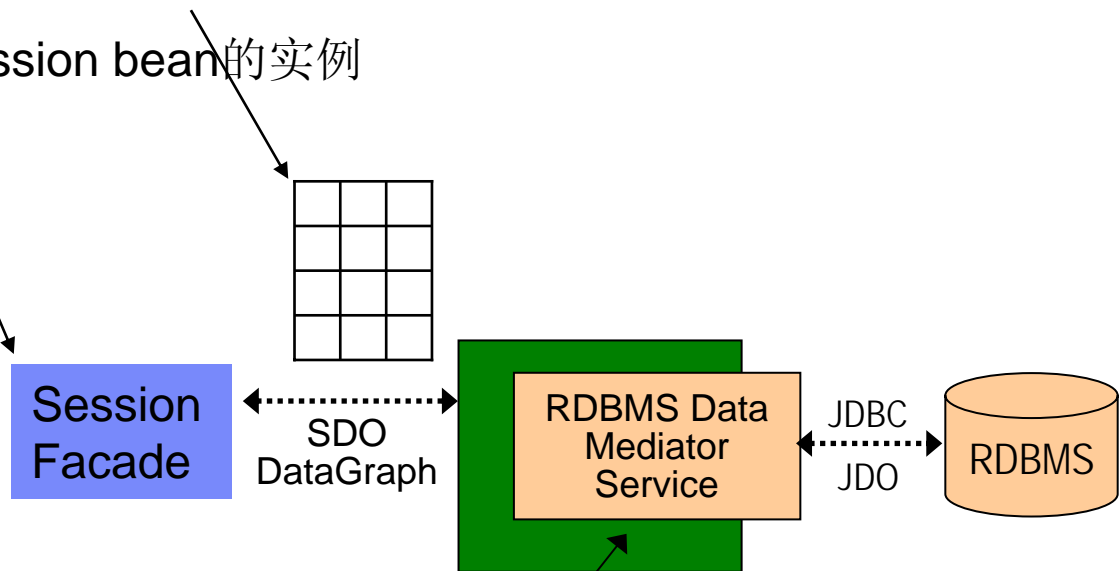
- 通过JSF
- 通过Entity Bean





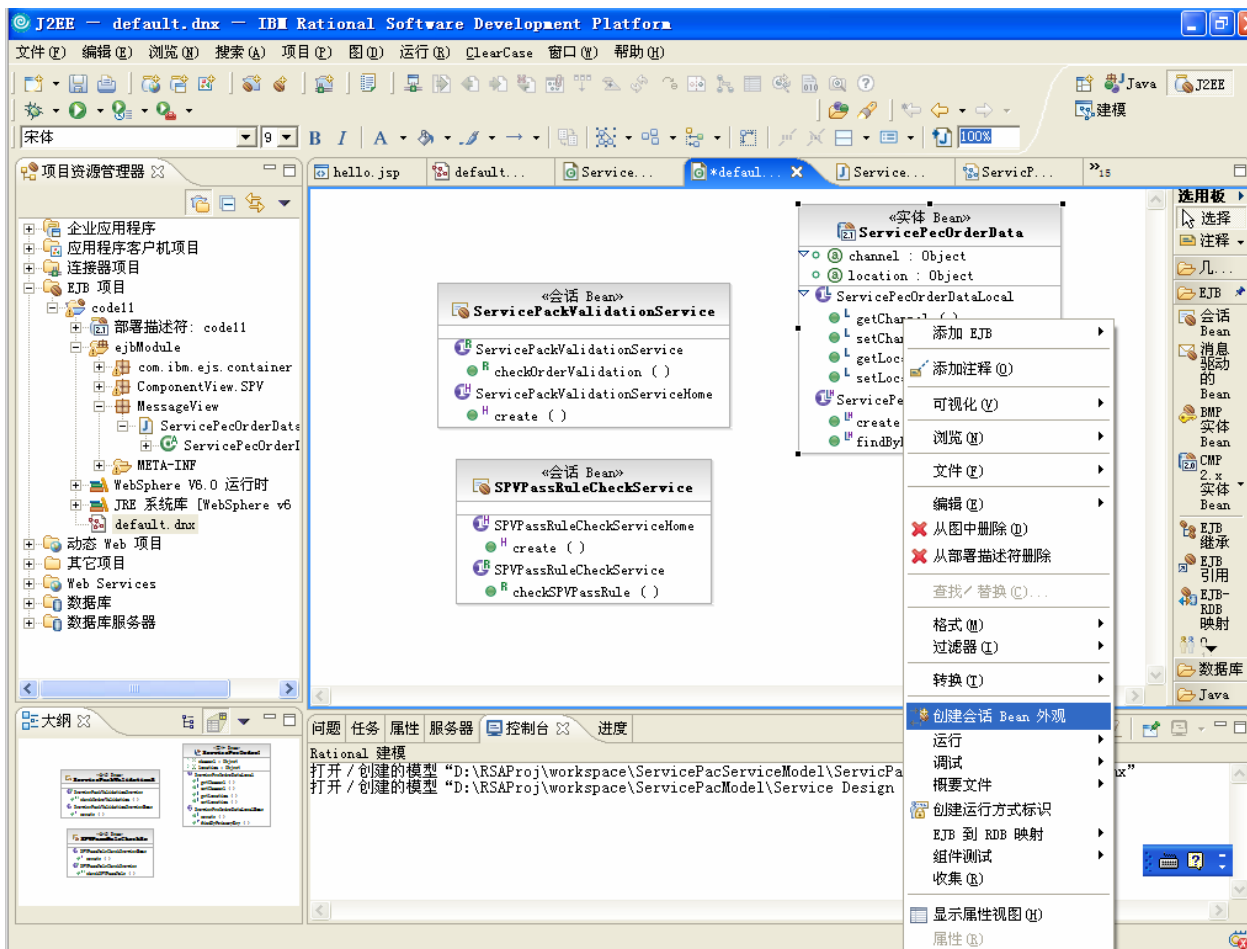
# EJB session façade及SDO 的创建

- 创建session façade bean
  - ▶ 通过CMP映射
  - ▶ 封装在一个或多个CMP实体bean中的数据和逻辑属性
  - ▶ 同时创建相应的SDO
- 结果是引用SDO的Session bean的实例

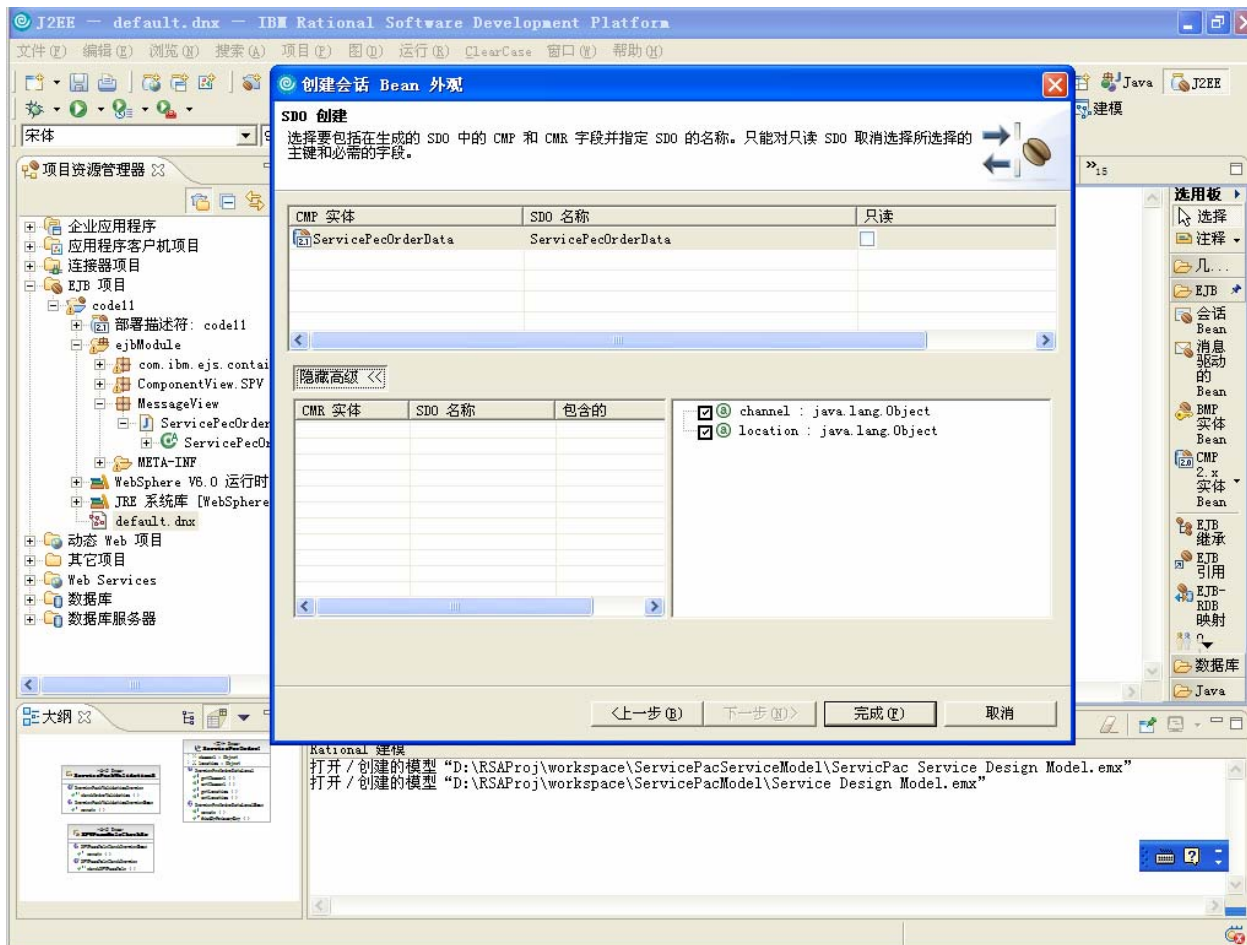


*Query data sources, create data graphs containing data objects, apply changes back to the data source*

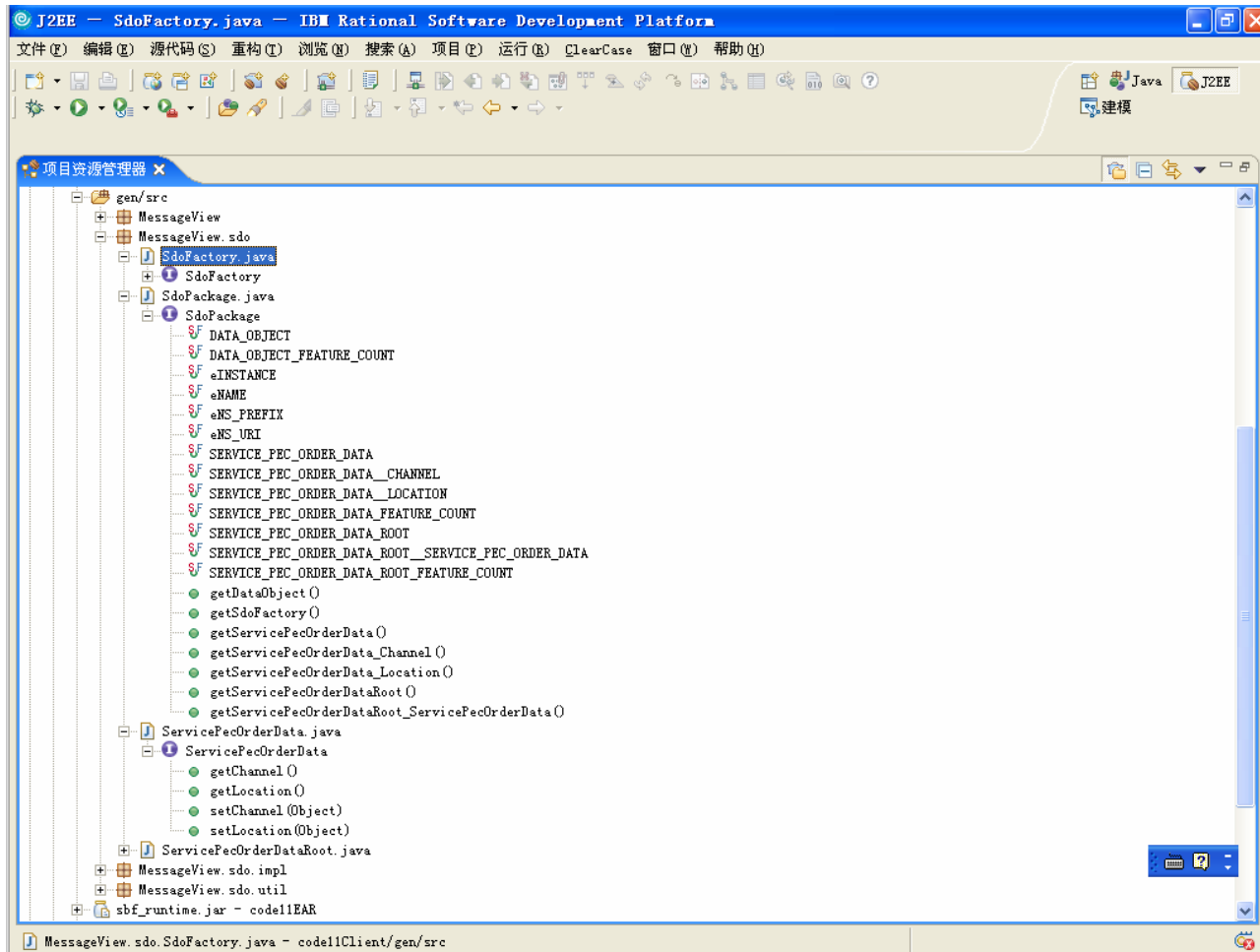
# Demo10-根据CMP产生SDO



# 根据CMP产生SDO



# 根据CMP产生SDO



## 构建服务组件 - SOA开发第三步

- 快速构建Web Service
- 快速构建SDO
- 快速构建JSF



# JSF简介

JSF为开发 Web应用用户界面提供了:

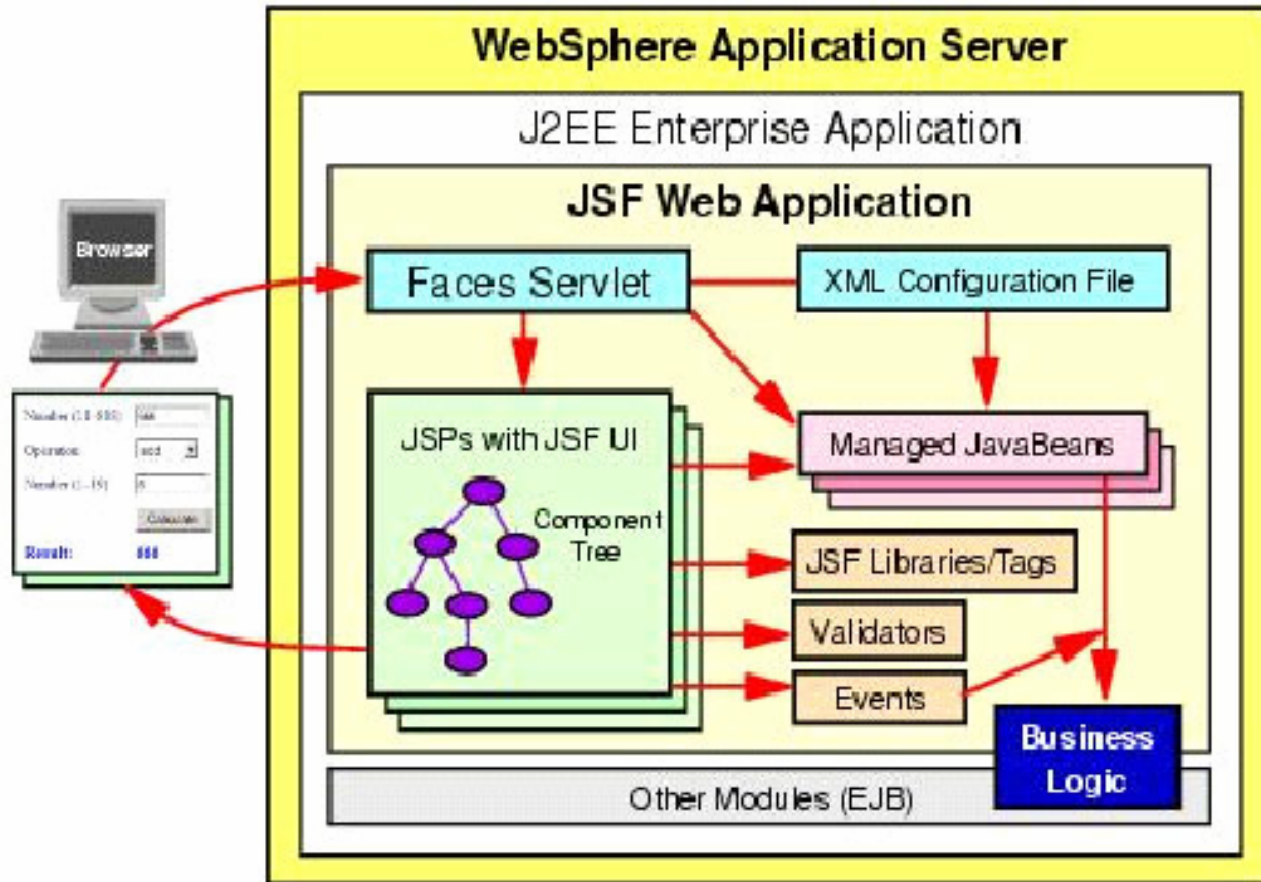
- 标准的编程接口
- 丰富可扩展的UI组件库
- 事件驱动模型等一套完整的Web应用框架

通过 JSF ， 开发人员在页面中轻松自如地使用 WEB 组件、捕获用户行为所产生的事件、执行验证、建立页面导航等

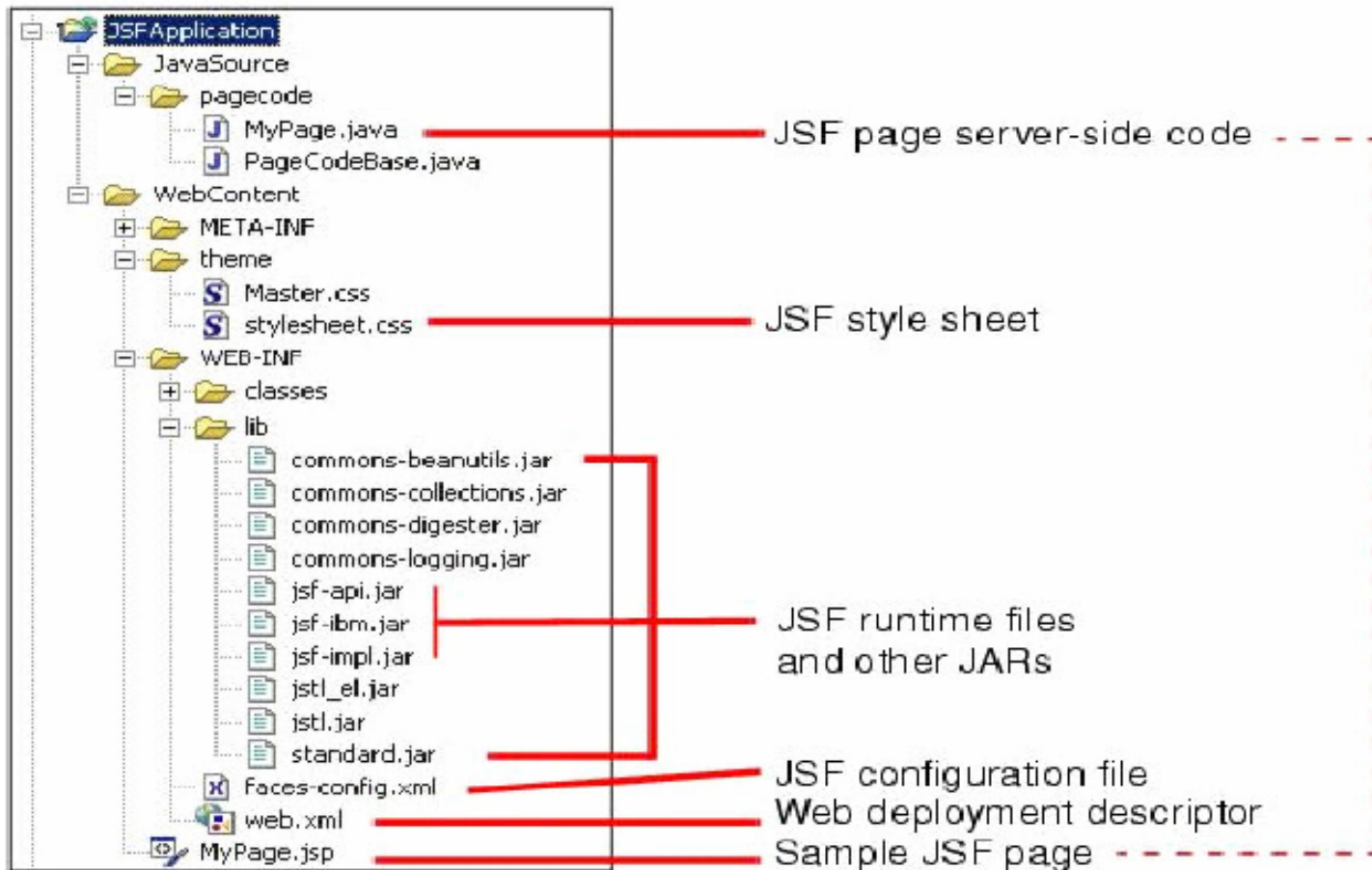
JSF:



## JSF 框架



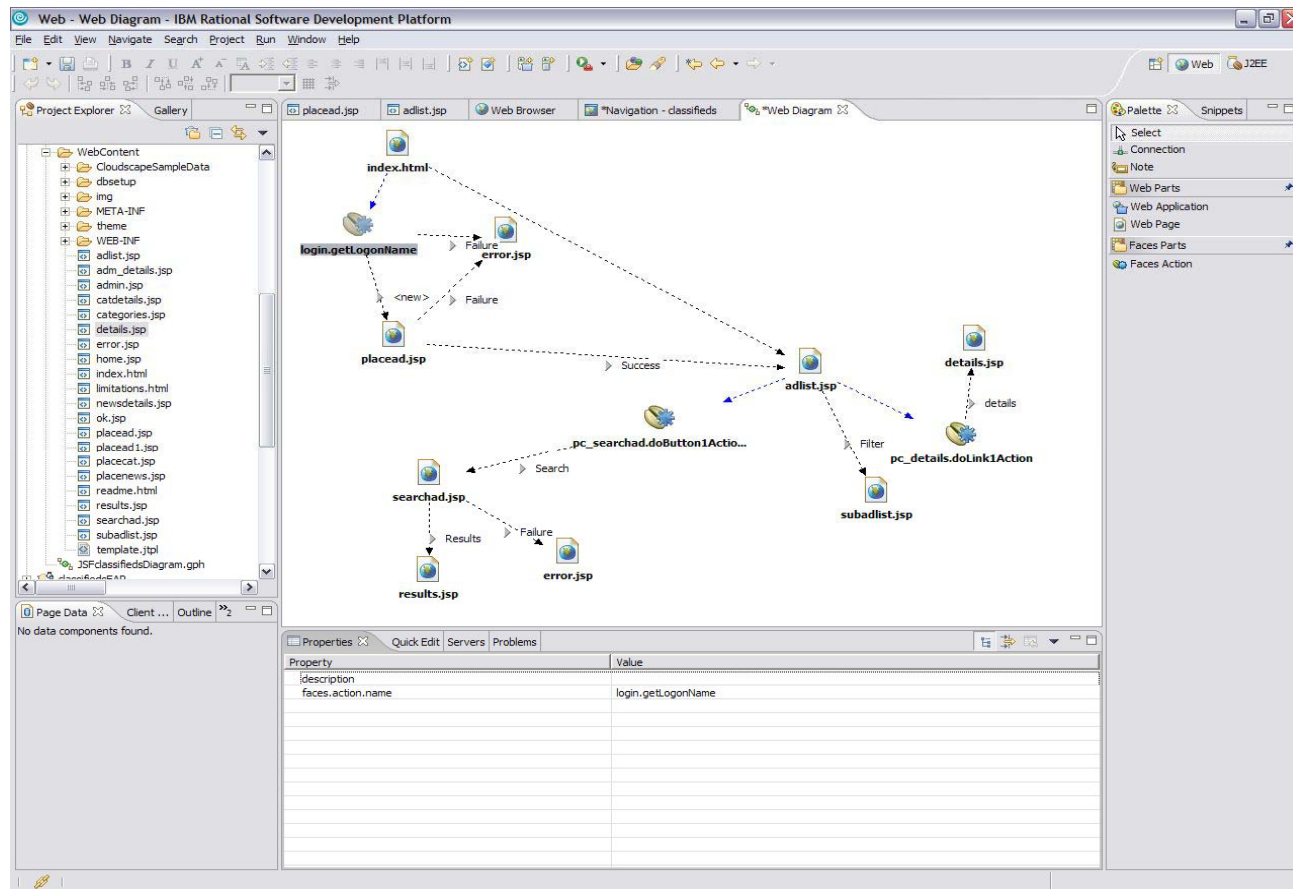
# RAD开发JSF一目录结构





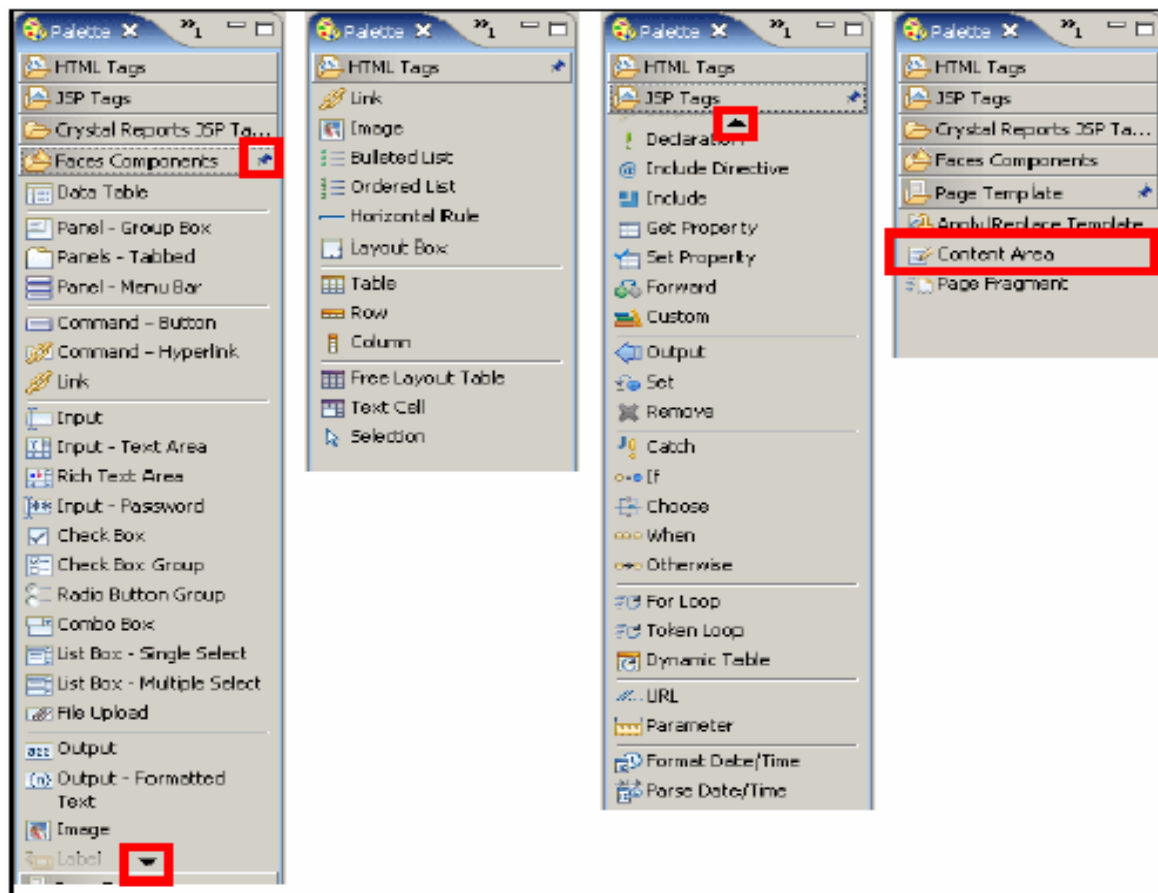
# RAD开发JSF—Web Diagram

通过Web Diagram tool 创建JSF资源如JSF pages, JSPs, Faces actions, 和 pages与actions之间的连接



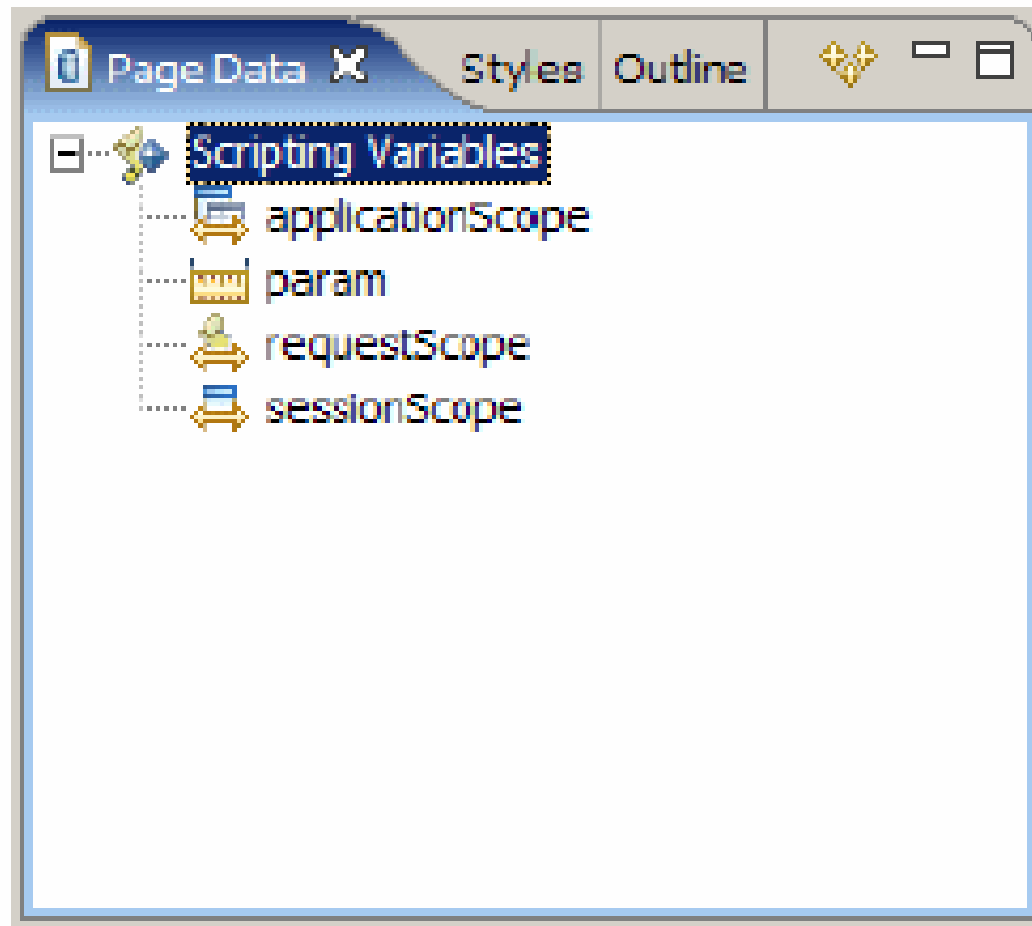
# RAD开发JSF—Palette

Palette包括一些常用的组件如JSP tags, HTML tags 和Face components 等等, 可以直接拖拽到JSF页面上



# RAD开发JSF—Page Data

可以将各种数据源  
如Web Service,  
Database (SDO) 和  
EJB Session Bean  
等加入到页面中



Thank  
You

