IBM Rational Software Conference 2009

RAD Extensibility for Development based Analytics

Rajesh Kalyanaraman,
Staff Software Engineer, RAD
IBM Software Labs, Bangalore

S. Srilakshmi,
Architect – Technology
Java Center of Excellence – GTO, Cognizant

Rational software

Cognizant
Passion for building stronger businesses

Evans Data Corporation
EDC

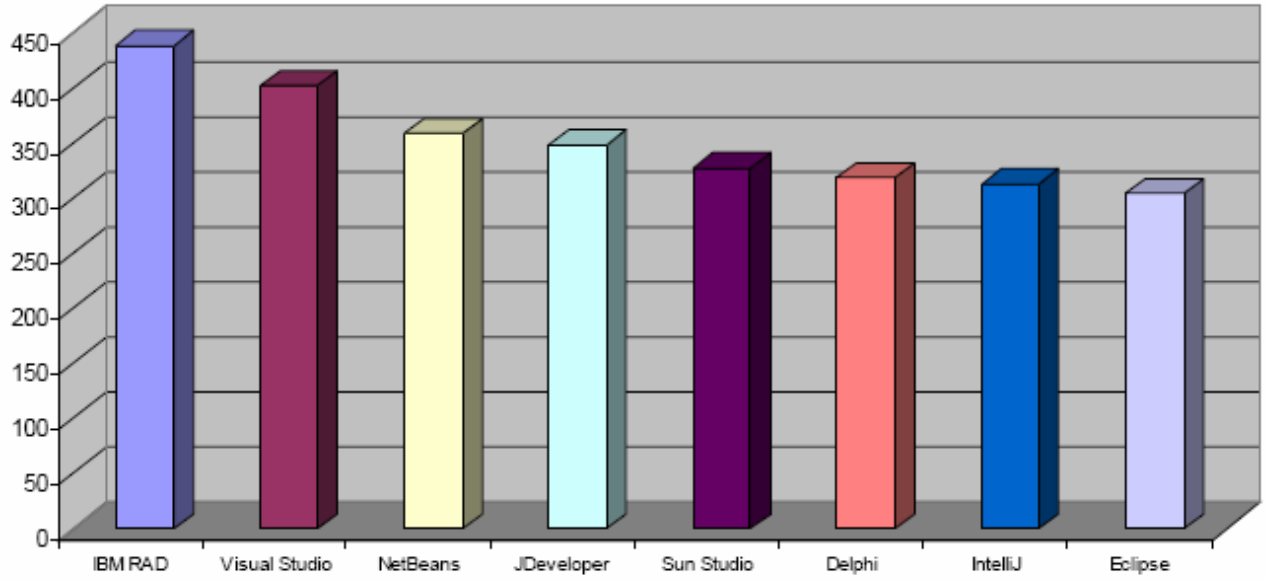
Users' Choice:
2009 Software Development Platforms

A comprehensive user satisfaction survey of over 1200 software developers

June 2009

Evans Data Corp, 740 Front St, Santa Cruz, CA 95060
www.evansdata.com (800) 831 3080

Overall Software Development Platform Ranking



Agenda

- **RAD Extensibility**
  - Project Metrics API using JDT & AST
  - Custom Plug-in Development
  - Reporting Infrastructures – BIRT & Crystal Reports
  - Building Custom JSF Web Components
  - Building Visual Custom Tags
- **JACP System Overview**
- **JACP Demo**
- **Q & A**

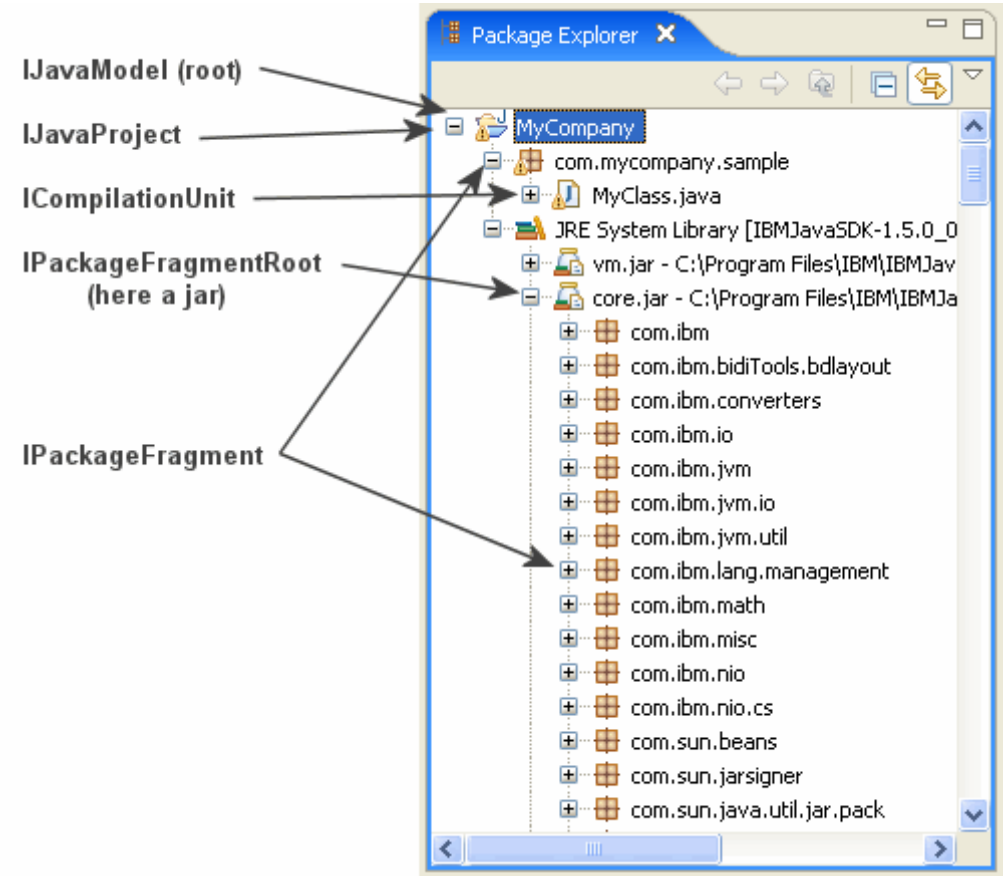
Project Metrics from JDT

- **Java Development Tooling**
  - JDT Core - the headless infrastructure for compiling and manipulating Java code.
  - JDT UI - the user interface extensions that provide the IDE.
  - JDT Debug - program launching and debug support specific to the Java programming language.
- **You can**
  - Programmatically manipulate Java resources, such as creating projects, generating Java source code, performing builds, or detecting problems in code.
  - Programmatically launch a Java program from the platform
  - Provide a new type of VM launcher to support a new family of Java runtimes
  - Add new functions and extensions to the Java IDE itself

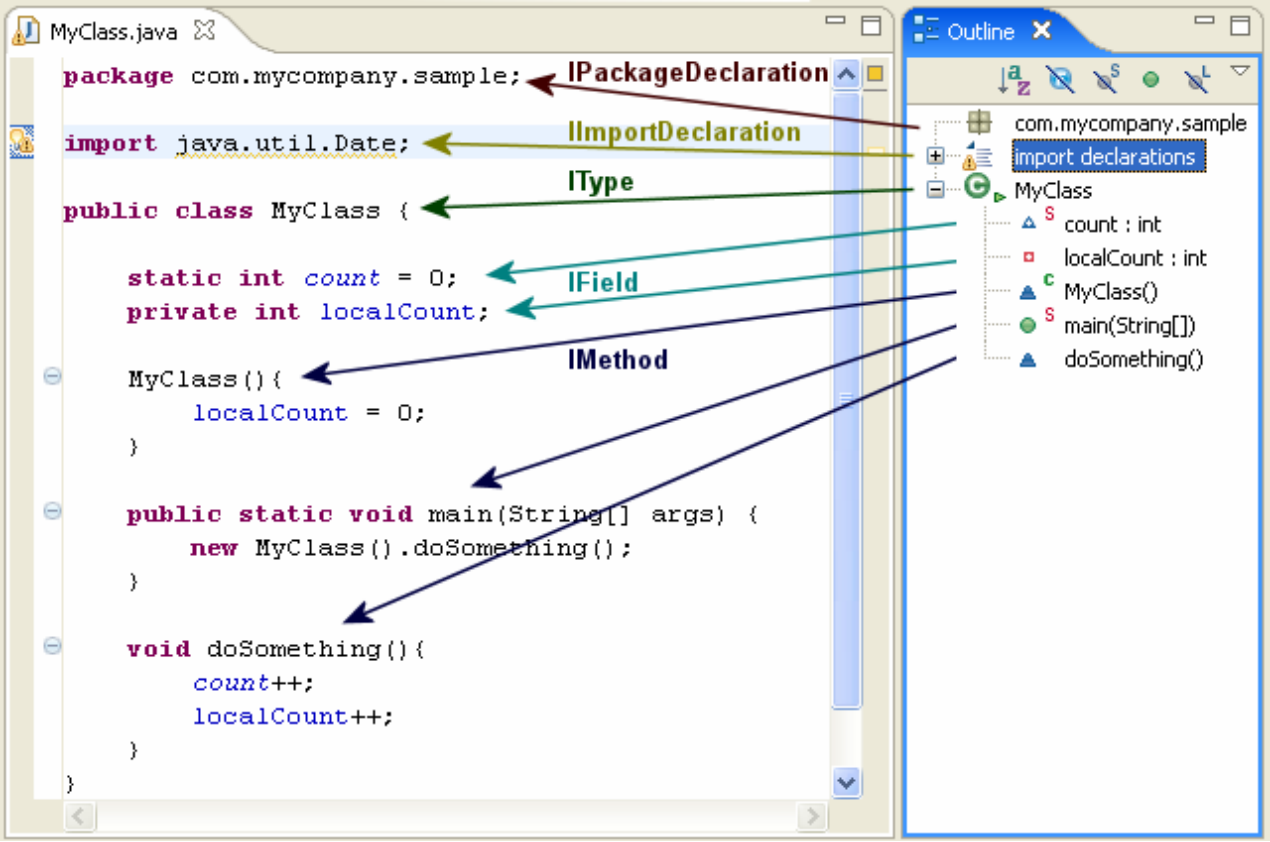
JDT Core APIs

- org.eclipse.jdt.core - defines the classes that describe the Java model.
- org.eclipse.jdt.core.compiler - defines an API for the compiler infrastructure.
- org.eclipse.jdt.core.dom - supports Abstract Syntax Trees (AST) that can be used for examining the structure of a compilation unit down to the statement level.
- org.eclipse.jdt.core.dom.rewrite - supports rewriting of Abstract Syntax Trees (AST) that can be used for manipulating the structure of a compilation unit down to the statement level.
- org.eclipse.jdt.core.eval - supports the evaluation of code snippets in a scrapbook or inside the debugger.
- org.eclipse.jdt.core.formatter - supports the formatting of compilation units, types, statements, expressions, etc.
- org.eclipse.jdt.core.jdom - supports a Java Document Object Model (DOM) that can be used for walking the structure of a Java compilation unit. (deprecated – use org.eclipse.jdt.core.dom)
- org.eclipse.jdt.core.search - supports searching the workspace's Java model for Java elements that match a particular description.
- org.eclipse.jdt.core.util - provides utility classes for manipulating .class files and Java model elements.

Java Model


IJavaModel (root)
IJavaProject
ICompilationUnit
IPackageFragmentRoot
(here a jar)
IPackageFragment
Package Explorer
MyCompany
com.mycompany.sample
MyClass.java
JRE System Library [IBMJavaSDK-1.5.0_0
vm.jar - C:\Program Files\IBM\IBMJav
core.jar - C:\Program Files\IBM\IBMJa
com.ibm
com.ibm.bidiTools.bdlayout
com.ibm.converters
com.ibm.io
com.ibm.jvm
com.ibm.jvm.io
com.ibm.jvm.util
com.ibm.lang.management
com.ibm.math
com.ibm.misc
com.ibm.nio
com.ibm.nio.cs
com.sun.beans
com.sun.jarsigner
com.sun.java.util.jar.pack

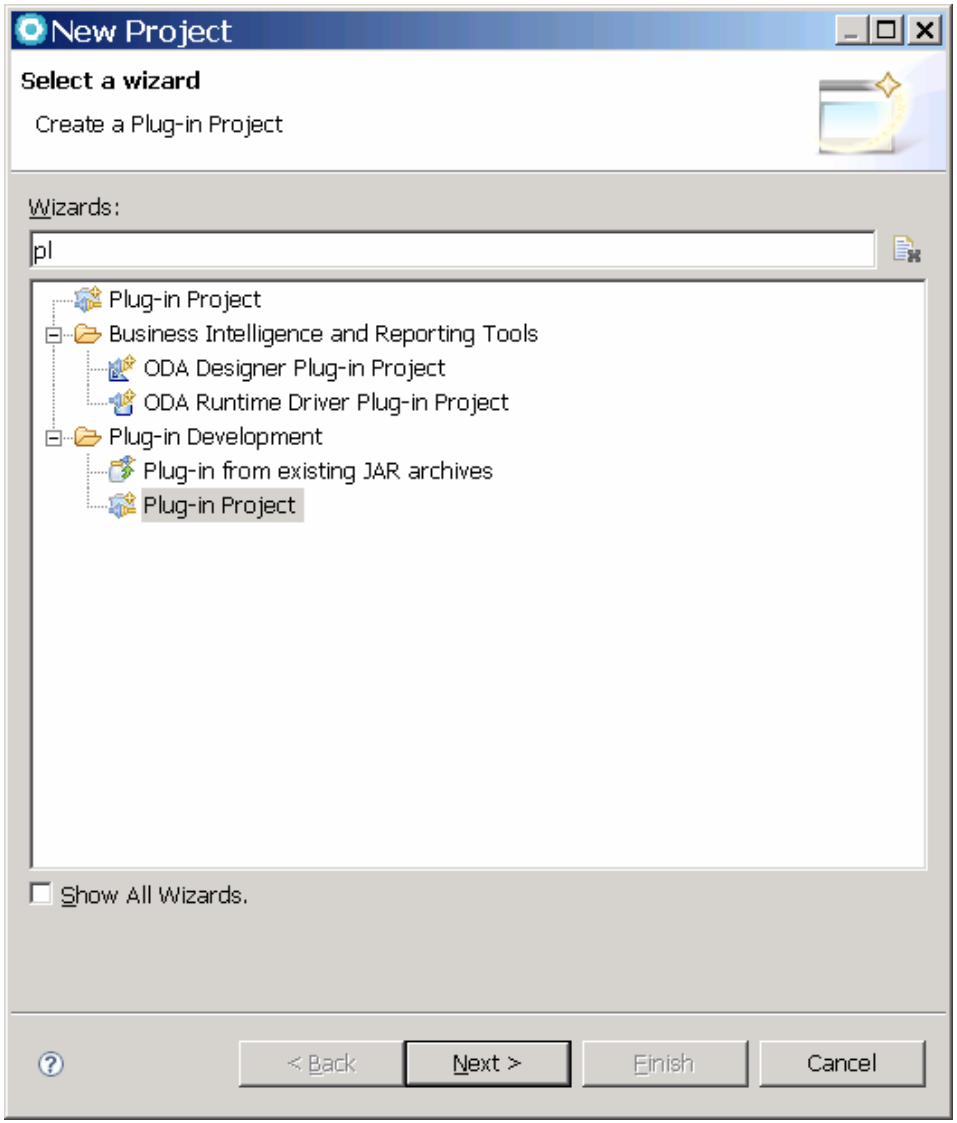
Java Compilation Unit



Code modification using the DOM/AST API

- **AST (Abstract Syntax Tree)**
- **Ways to create a compilation unit**
  - **ASTParser**
  - **ICompilationUnit#reconcile(...)**
    - start from scratch using the factory methods on **AST** (Abstract Syntax Tree).
- **Creating AST from Source code – ASTParser.** createAST(IProgressMonitor)
  - **setSource(char[])**: to create the AST from source code
  - **setSource(IClassFile)**: to create the AST from a classfile
  - **setSource(ICompilationUnit)**: to create the AST from a compilation unit

Plug-in Development Wizard



New Plug-in Project

Plug-in Project

Create a new plug-in project

Project name: com.ibm.example.cts2

☑ Use default location

Location: C:\Documents and Settings\rajeshk\Desktop\CTS Worksl Browse...

Project Settings

☑ Create a Java project

Source folder: src

Output folder: bin

Target Platform

This plug-in is targeted to run with:

◉ Eclipse version: 3.4

○ an OSGi framework: Equinox

Working sets

☐ Add project to working sets

Working sets: Select...

< Back | Next > | Finish | Cancel

New Plug-in Project

Plug-in Content

Enter the data required to generate the plug-in.

Plug-in Properties

Plug-in ID: com.ibm.example.cts2

Plug-in Version: 1.0.0

Plug-in Name: Cts2 Plug-in

Plug-in Provider: IBM

Execution Environment: JavaSE-1.6 Environments...

Plug-in Options

☑ Generate an activator, a Java class that controls the plug-in's life cycle

Activator: com.ibm.example.cts2.Activator

☑ This plug-in will make contributions to the UI

☐ Enable API Analysis

Rich Client Application

Would you like to create a rich client application? ○ Yes ◉ No

< Back | Next > | Finish | Cancel

New Plug-in Project

Templates

Select one of the available templates to generate a fully-functioning plug-in.

☑ Create a plug-in using one of the templates

Available Templates:

- Custom plug-in wizard
- Figure definitions converter
- Hello, World
- Hello, World Command
- ODA Data Source Designer
- ODA Data Source Runtime Driver
- Plug-in with a multi-page editor
- Plug-in with a popup menu
- Plug-in with a property page
- Plug-in with a view
- Plug-in with an editor
- Plug-in with an incremental project bui
- Plug-in with sample help content

This wizard creates standard plug-in directory structure and adds the following:

- **Sample view**. This template create a workbench view. The view is contributed to the workbench by creating a category. The view can b opened by selecting **Window**, **Show View** and then **Other...** on the men bar. The template demonstrates implementation of pop-up menu support, local tool bar, double-click, sorting and filtering. There is also ar option to add context-sensitive help the view.

Extensions Used

- org.eclipse.ui.views
- org.eclipse.ui.perspectiveExtensions

< Back | Next > | Finish | Cancel

New plug-in project with a sample view

Main View Settings

Choose the way the new view will be added to the plug-in.

Java Package Name:	com.ibm.example.cts2.views
View Class Name:	SampleView
View Name:	Sample View CTS2
View Category ID:	com.ibm.example.cts2
View Category Name:	Sample Category

Select the viewer type that should be hosted in the view:

◉ Table viewer (can also be used for lists) ○ Tree viewer

☑ Add the view to the java perspective

☑ Add context help to the view

< Back | Next > | Finish | Cancel

Plug-in Development - com.ibm.example.cts2/META-INF/MANIFEST.MF - Rational® Application Developer™ for WebSphere® Software

File Edit Navigate Search Project Run Window Help

Package Explorer | Plug-ins

- com.ibm.example.cts1
  - JRE System Library [JavaSE-1.6]
  - Plug-in Dependencies
  - src
  - icons
  - META-INF
  - build.properties
  - contexts.xml
  - plugin.xml
- com.ibm.example.cts2
  - JRE System Library [JavaSE-1.6]
  - Plug-in Dependencies
  - src
  - icons
  - META-INF
    - MANIFEST.MF
  - build.properties
  - contexts.xml
  - plugin.xml
- JDTProj1
- NewVersion_ProjMetricReport1
- ProjBIRTReport1
- ProjJavaReports
- TestJavaProj1

com.ibm.example.cts2 | contexts.xml

Overview

General Information

This section describes general information about this plug-in.

ID: com.ibm.example.cts2
Version: 1.0.0
Name: Cts2 Plug-in
Provider: IBM
Platform Filter:
Activator: com.ibm.example.cts2.Activato [Browse...]

☑ Activate this plug-in when one of its classes is loaded
☑ This plug-in is a singleton

Execution Environments

Specify the minimum execution environments required to run this plug-in.

JavaSE-1.6 [Add...] [Remove] [Up] [Down]

Configure JRE associations...
Update the classpath settings

Plug-in Content

The content of the plug-in is made up of two sections:

- Dependencies: lists all the plug-ins required on this plug-in's classpath to compile and run.
- Runtime: lists the libraries that make up this plug-in's runtime.

Extension / Extension Point Content

This plug-in may define extensions and extension points:

- Extensions: declares contributions this plug-in makes to the platform.
- Extension Points: declares new function points this plug-in adds to the platform.

Testing

Test this plug-in by launching a separate Eclipse application:

- Launch an Eclipse application
- Launch an Eclipse application in Debug mode

Exporting

To package and export the plug-in:

1. Organize the plug-in using the Organize Manifests Wizard
2. Externalize the strings within the plug-in using the Externalize Strings Wizard
3. Specify what needs to be packaged in the deployable

Overview | Dependencies | Runtime | Extensions | Extension Points | Build | MANIFEST.MF | plugin.xml | build.properties

Outline

- Overview
- Dependencies
  - org.eclipse.ui
  - org.eclipse.core.runtime
- Runtime
- Extensions
  - org.eclipse.ui.views
  - org.eclipse.ui.perspec
  - org.eclipse.help.conte
- Extension Points
- Build

Context help with contexts.xml

Plug-in Development - com.ibm.example.cts2/contexts.xml - Rational® Application Developer™ for WebSphere® Software

File Edit Navigate Search Project Run Window Help

Package Explorer | Plug-ins

- com.ibm.example.cts1
  - JRE System Library [JavaSE-1.6]
  - Plug-in Dependencies
  - src
  - icons
  - META-INF
  - build.properties
  - contexts.xml
  - plugin.xml
- com.ibm.example.cts2
  - JRE System Library [JavaSE-1.6]
  - Plug-in Dependencies
  - src
  - icons
  - META-INF
    - MANIFEST.MF
  - build.properties
  - contexts.xml
  - plugin.xml
- JDTProj1
- NewVersion_ProjMetricReport1
- ProjBIRTReport1
- ProjJavaReports
- TestJavaProj1

com.ibm.example.cts2 | contexts.xml

Context Help — Register this context help file

Context Help

Edit the structure of the context help file in this section.

type filter text

- contexts
  - viewer
    - This is the context help for th
    - Context-sensitive help

Add Context | Add Topic | Add Command | Remove | Up | Down

Topic Details

Specify the display label for this topic:

Label: Context-sensitive help

Specify the location of an HTML file containing content:

Location: /PLUGINS_ROOT/org.eclipse.platform. Browse...

Definition | Source

Outline

- Definition
  - contexts
    - viewer
      - This is the conte
      - Context-sensitiv

Run Configuration for the plug-in

Run Configurations

Create, manage, and run configurations

Create a configuration to launch an Eclipse application.

type filter text

- Eclipse Application
  - New_configuration
- Eclipse Data Tools
- Generic Server
- Generic Server(External Launc
- HTTP Preview
- J2EE Preview
- Java Applet
- Java Application
- JET Transformation
- JUnit
- JUnit Plug-in Test
- OSGi Framework
- Web Browser
- WebSphere Administrative Sc
- WebSphere Application Server
- WebSphere Application Server
- WebSphere Application Server
- XSL Transformation

Filter matched 22 of 22 items

Name: New_configuration

Main | Arguments | Plug-ins | Configuration | Tracing | Environment | Common

Launch with: plug-ins selected below only

type filter text

- Workspace
  - com.ibm.example.cts1 (1.0.0)
  - com.ibm.example.cts2 (1.0.0)
- Target Platform
  - com.businessobjects.crystalreports.designer.3.4.dseintegration (1.0.5.v1246)
  - com.businessobjects.crystalreports.designer.core (1.0.5.v1246)
  - com.businessobjects.crystalreports.designer.editor (1.0.5.v1246)
  - com.businessobjects.crystalreports.designer.enginepreferences (1.0.5.v1246)
  - com.businessobjects.crystalreports.designer.preview (1.0.5.v1246)

Select All | Deselect All | Add Working Set... | Add Required Plug-ins | Restore Defaults

Only show selected plug-ins

1815 out of 1816 selected

Include optional dependencies when computing required plug-ins

Add new workspace plug-ins to this launch configuration automatically

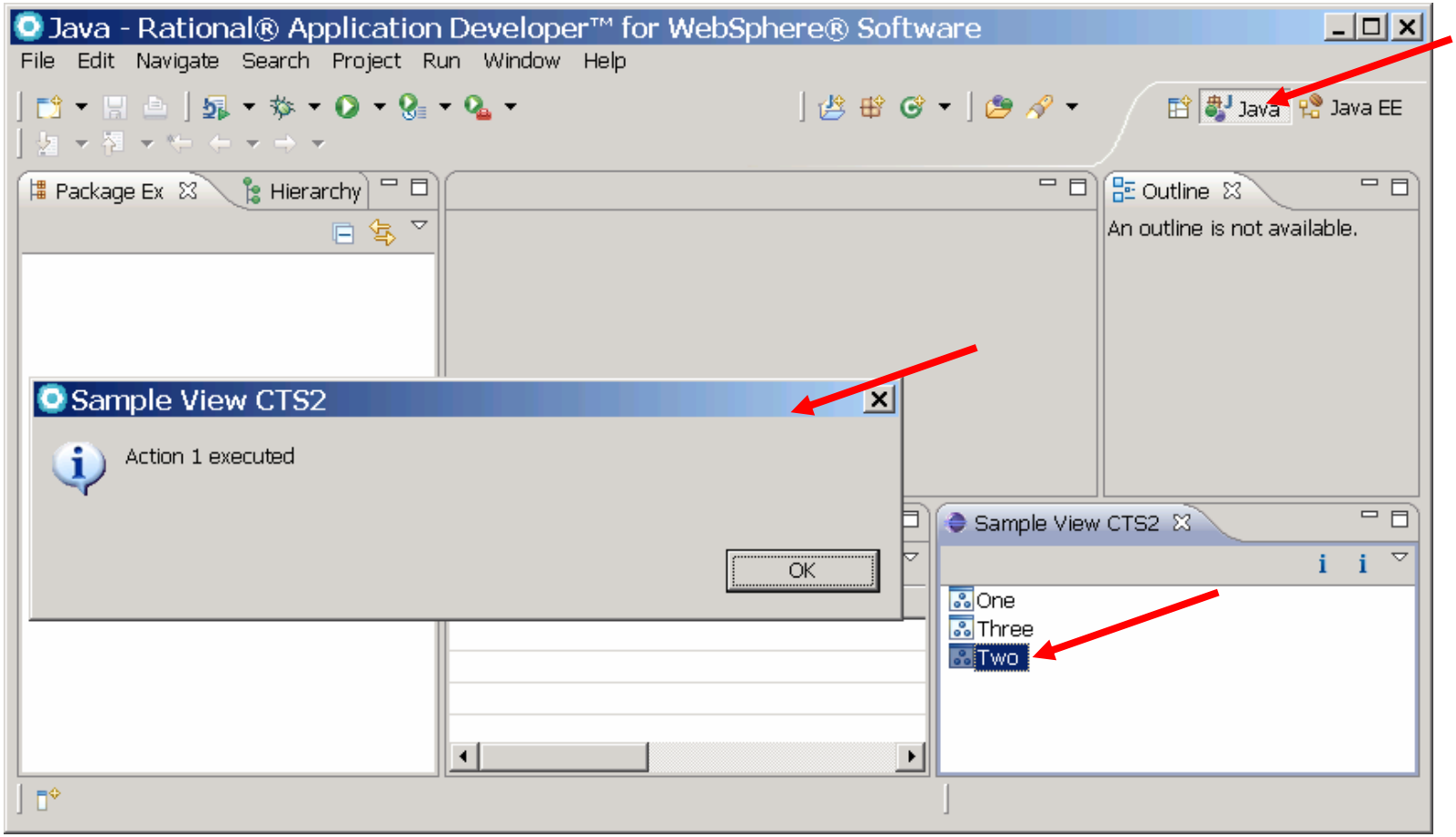
Validate plug-ins automatically prior to launching

Validate Plug-ins

Apply | Revert

Run | Close

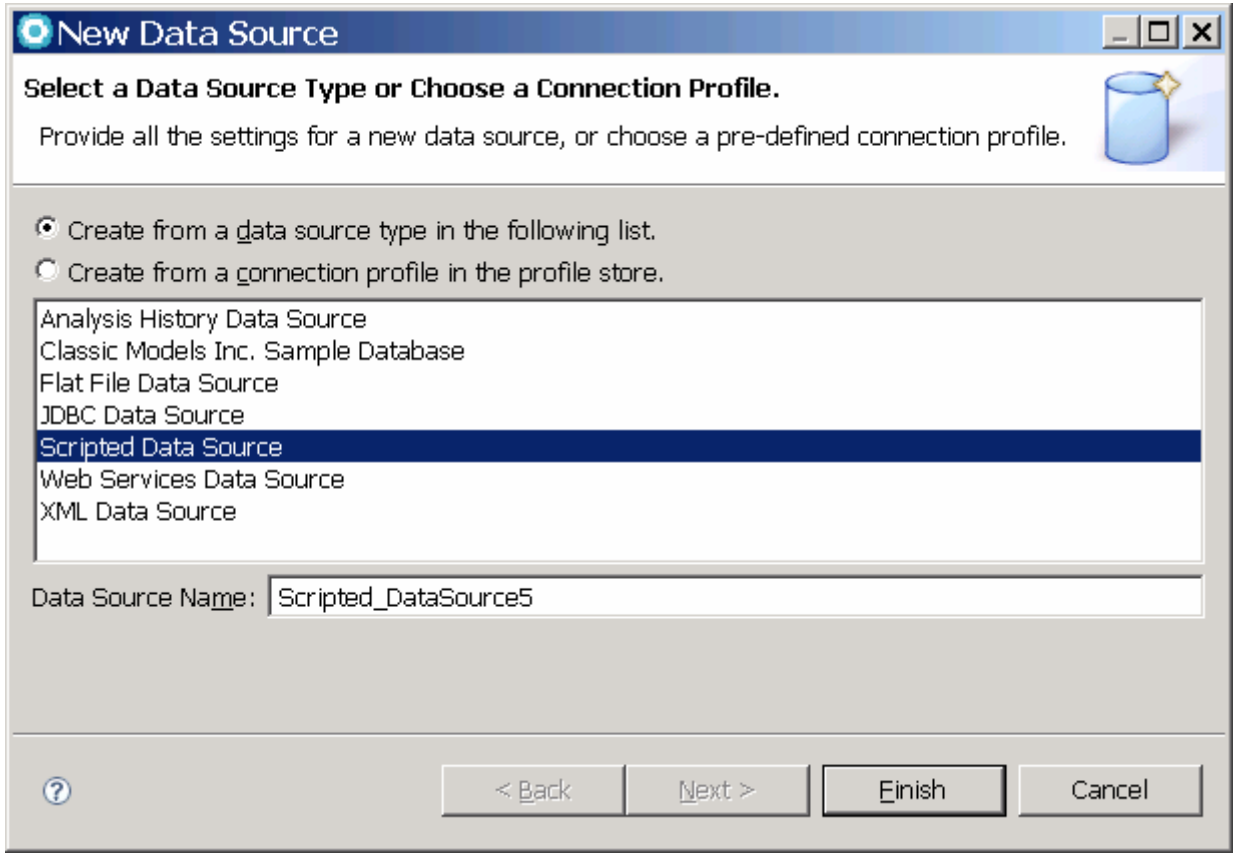
Our view in action !



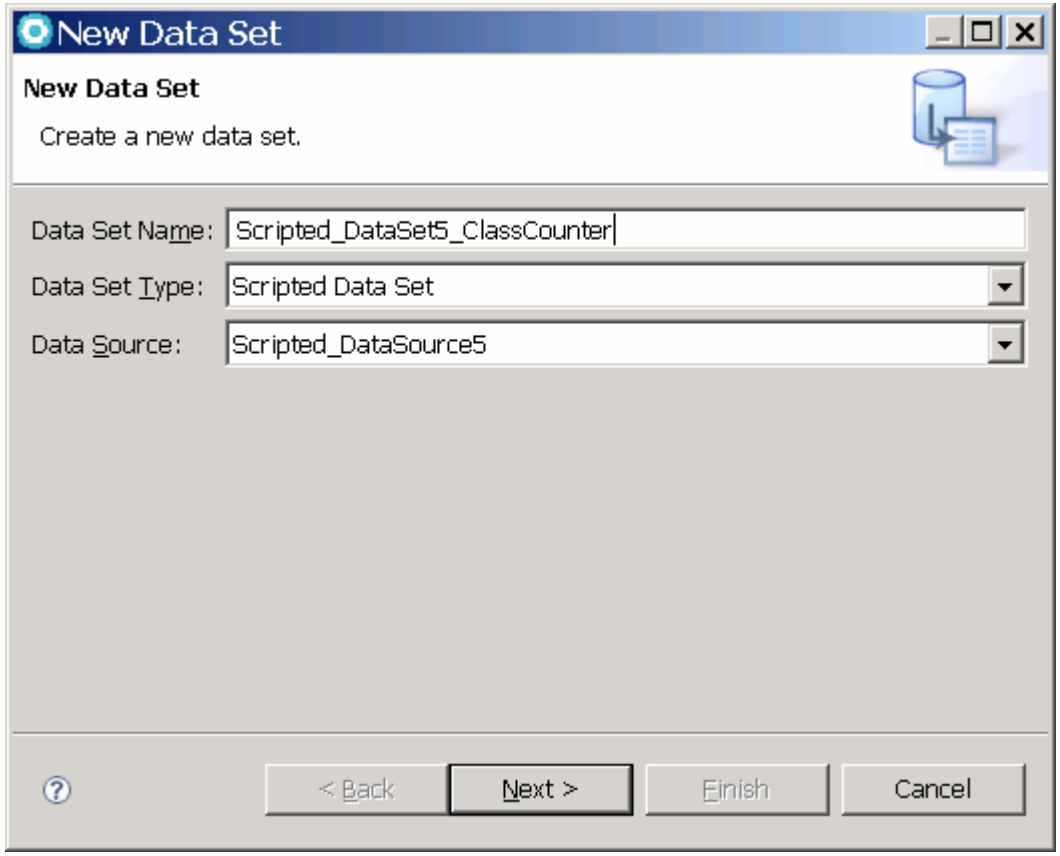
Reporting Infrastructure

- **RAD supports 2 ways for building reports**
  - BIRT
  - Crystal Reports
- **Designing Reports with BIRT**
  - Report Layout
    - Colors, fonts and positioning
  - DataSources
    - Can be JDBC /XML/Scripted Data Source/Web Service
  - DataSets
    - Corresponds to data records used in the details added dynamically to the report
- **Caching Build Reports**
  - Either Data or the built report can be cached

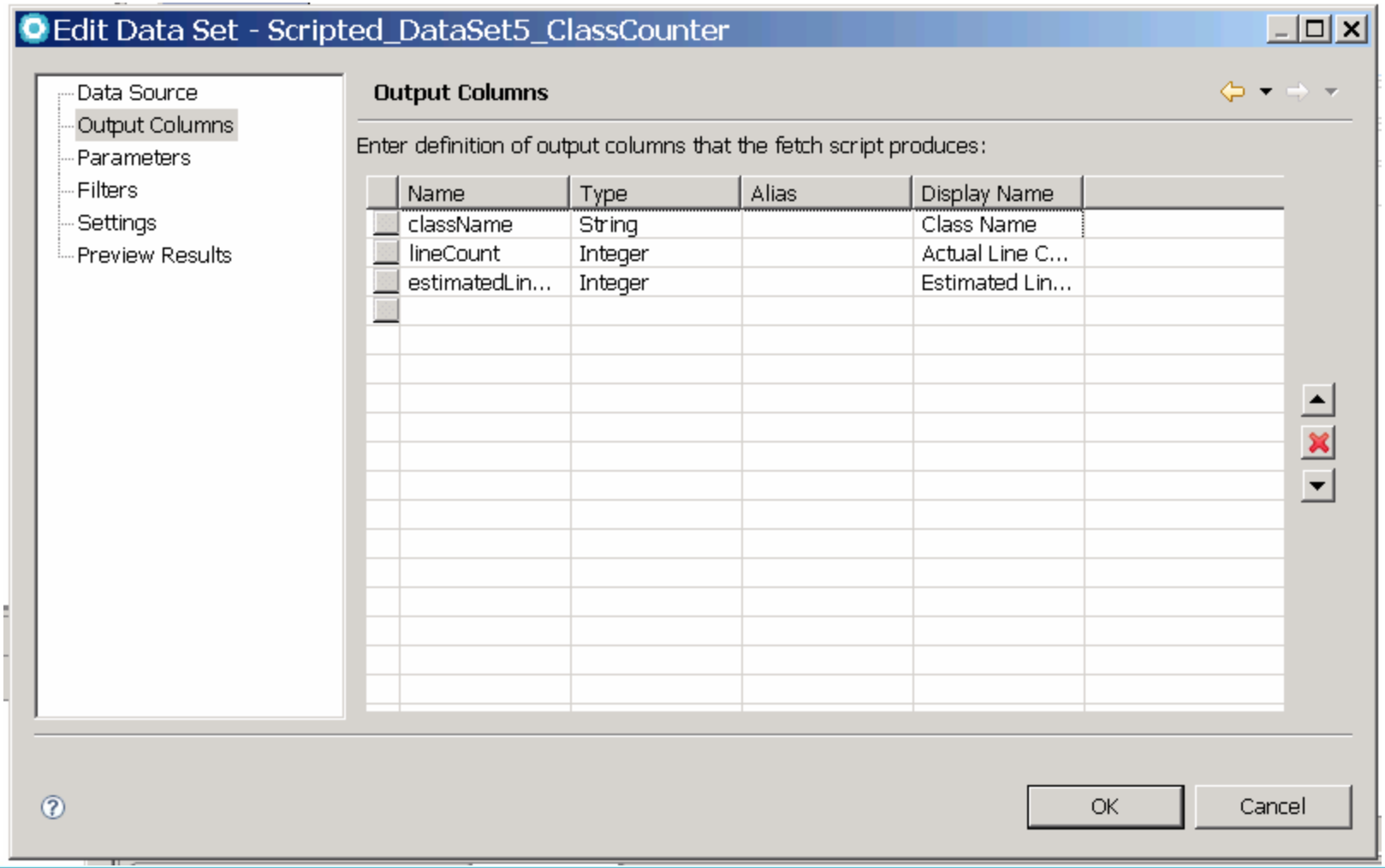
Data Source Types



Scripting Data Set -Steps



Scripting Data Set -Steps



Scripting Data Set -Steps

Palette | Data Explorer | Resource Expl

- Data Sources
- Data Sets
  - Scripted_DataSet5_ClassCounter
- Data Cubes
- Report Parameters

Navigator | Outline

- com.ibm.example.cts1
- JDTProj1
- NewVersion_ProjMetricReport1
- ProjBIRTReport1
  - .project

class_count3.rptdesi | SampleView.java | MyReportCounter.java | ClassLister.java

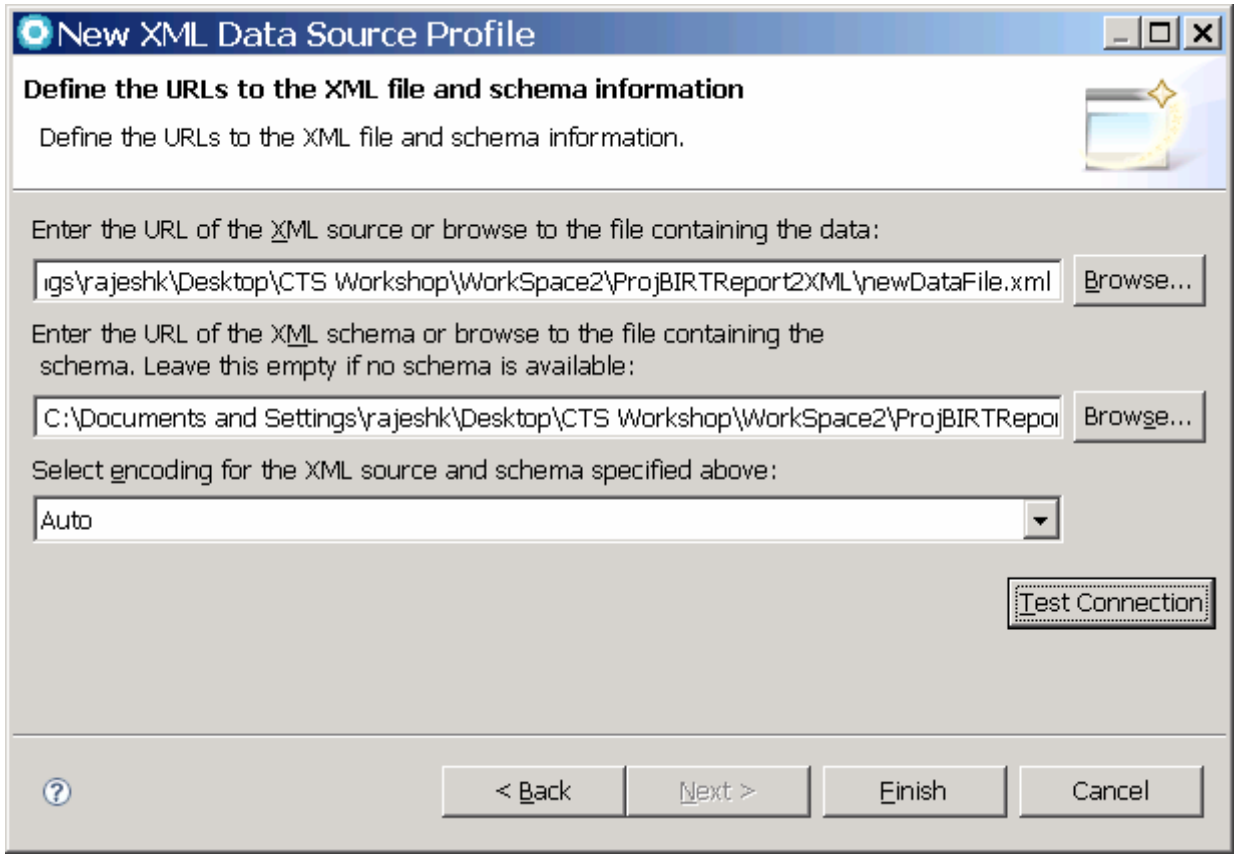
Script: fetch | Reset Script | Scripted_DataSet5_(

fetch
close
beforeOpen
beforeClose
onFetch

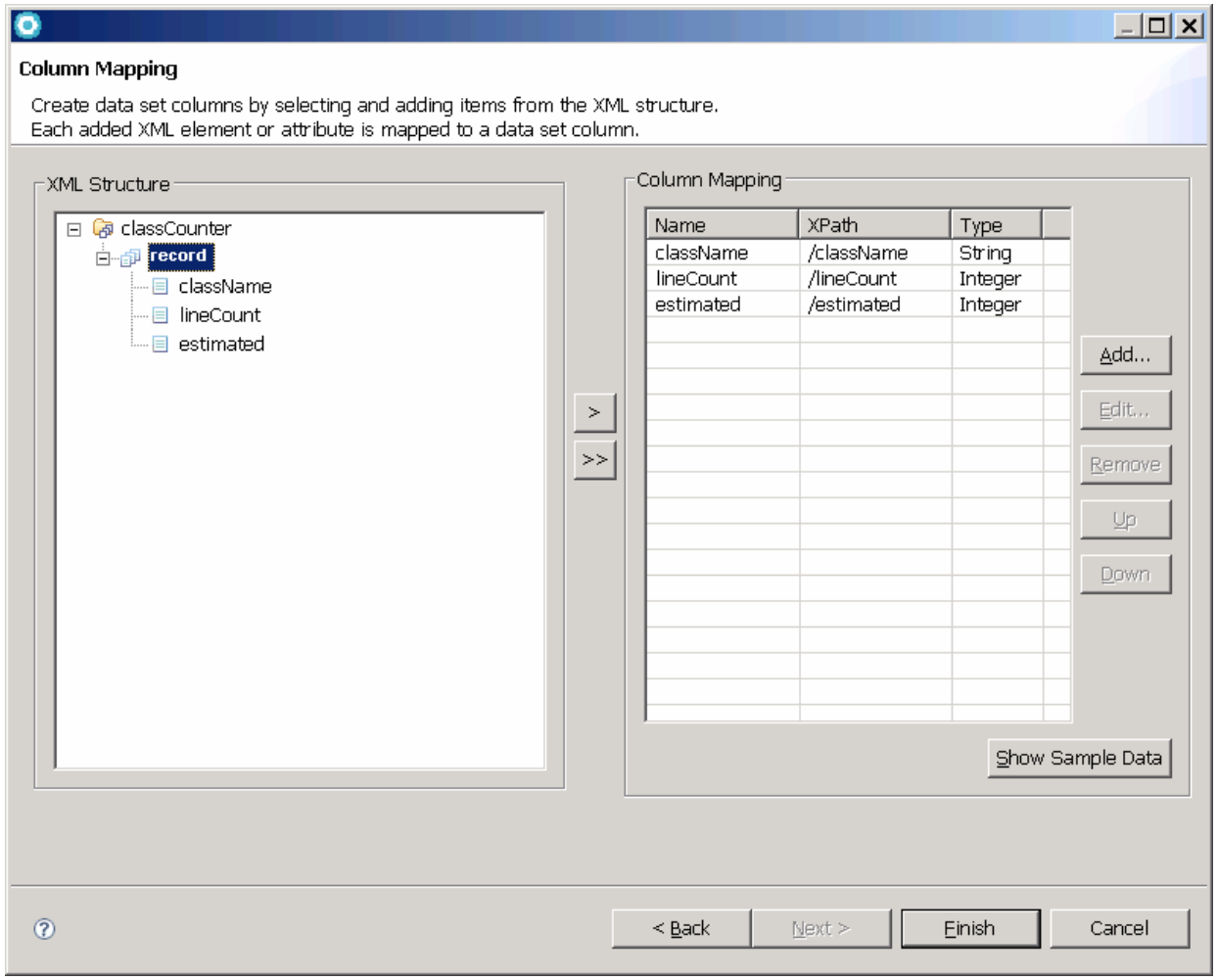
```
if(count < listCount ){

    //fetching the record from program


    className1 = dataRecords.strData1[count];
    lineCount1 = dataRecords.intData1[count];
    estimatedLineCount1 = dataRecords.intData2[count];

    //setting values to report record
    row["className"] = className1;
    row["lineCount"] =  lineCount1;
    row["estimatedLineCount"] = estimatedLineCount1 ;


    count++;

    return true;
}
return false;
```

Layout | Master Page | Script | XML Source | Preview

XML Data Source – Schema & Source



XML Data Set - Column Mapping



XML Data Set - Computed Columns

Edit Data Set - XMLClassCountDataSet2

- Data Source
- Sample XML Settings
- Row Mapping
- Column Mapping
- Output Columns
- Computed Columns
- Filters
- Property Binding
- Settings
- Preview Results

Computed Columns

Define computed columns:

Column Name	Data Type	Expression

New... Edit... Remove Up Down

New Computed Column.

Column Name: Difference
Data Type: Integer
Aggregation:
Expression:
Filter:

Expression can not be blank

OK Cancel

Expression Builder

Expression Builder

Type an expression in the Expression field. Browse the lists of available objects and double-click to copy

```
Math.abs(row["estimated"]-row["lineCount"])
```

Operators: + - * / ! = < > & | ( )

lineCount

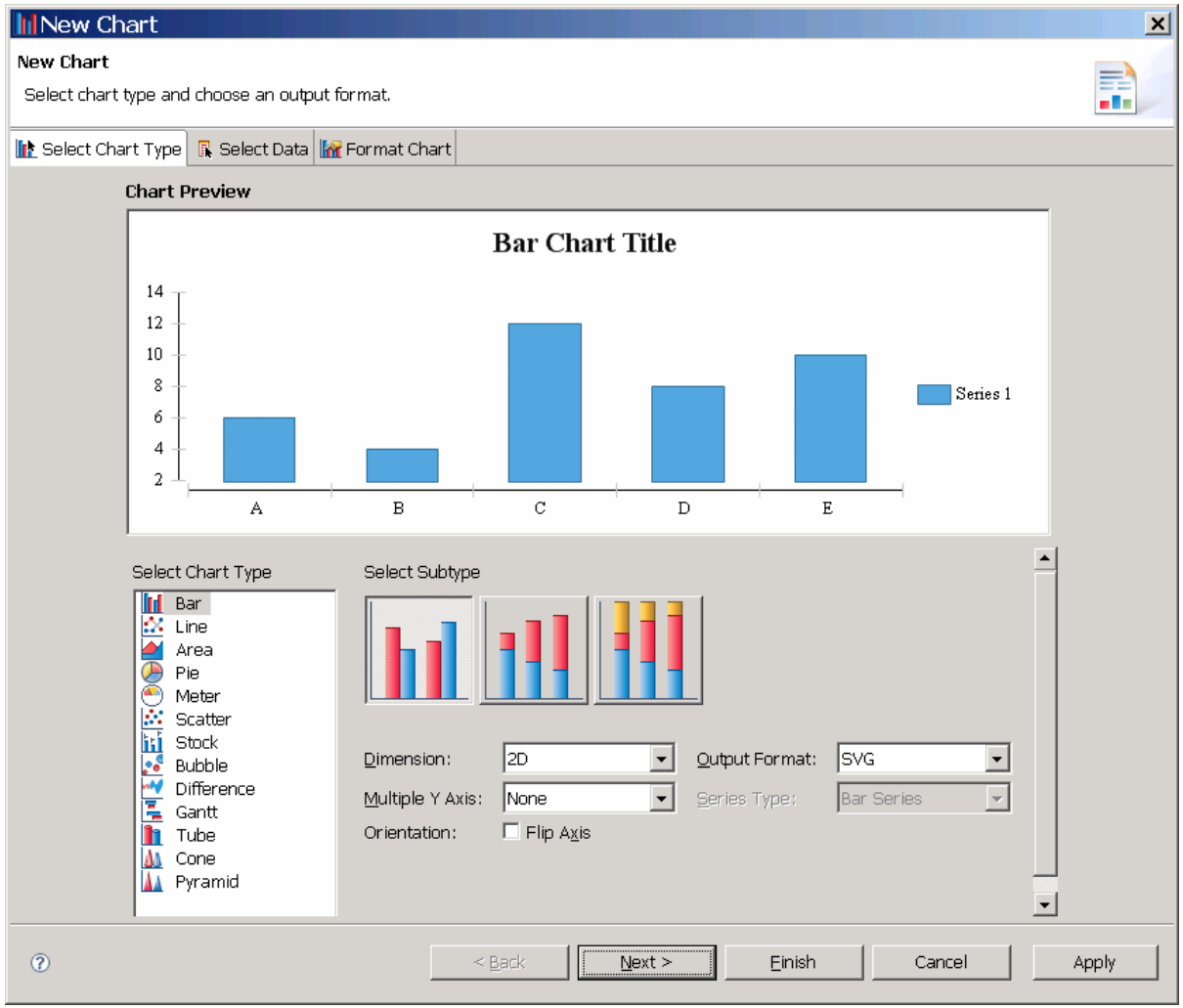
Category: Available Data Sets, Native JavaScript Functions, BIRT Functions, Operators

Sub-Category: XMLClassCountDataSet2

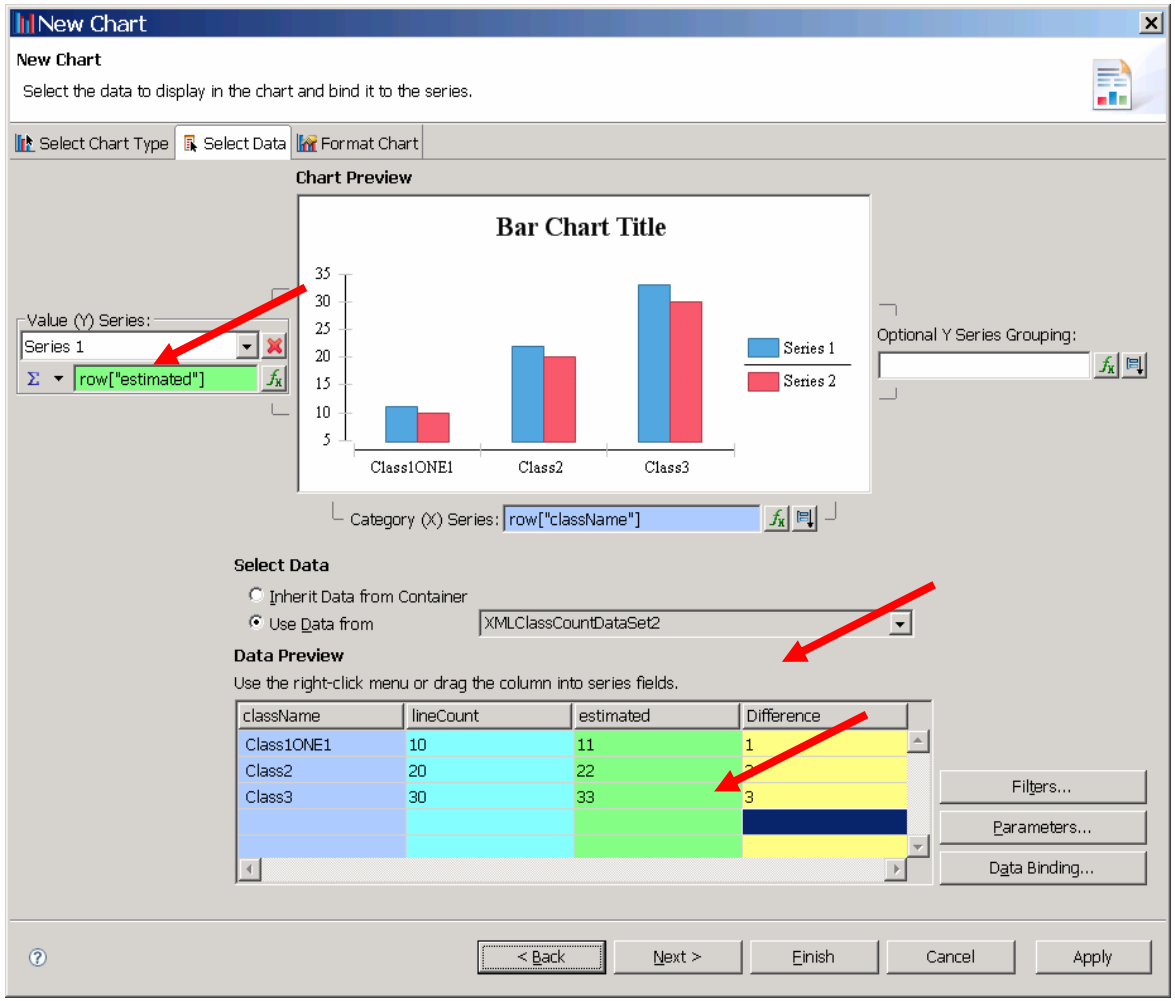
Double Click: className, lineCount, estimated

Native JavaScript Functions

Building a chart



Getting Chart Data from the Data Set



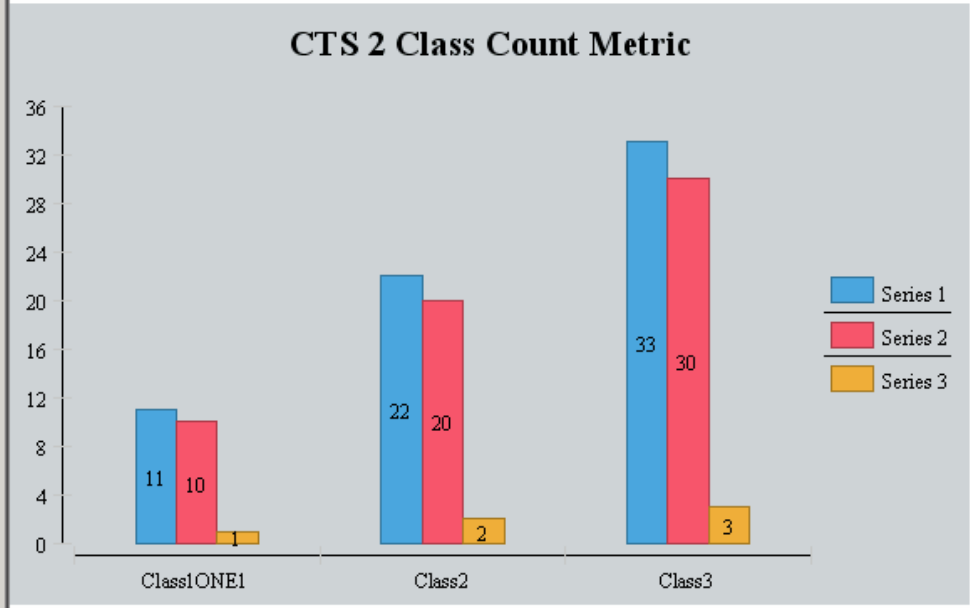
Preview Report

XmlDataReport.rptdesign | newDataFileSchema.xsd | XMLBasedReport.rptdesign | newDataFile.xml | ClassCount5_Chart.rptdesign

Show Report Parameters

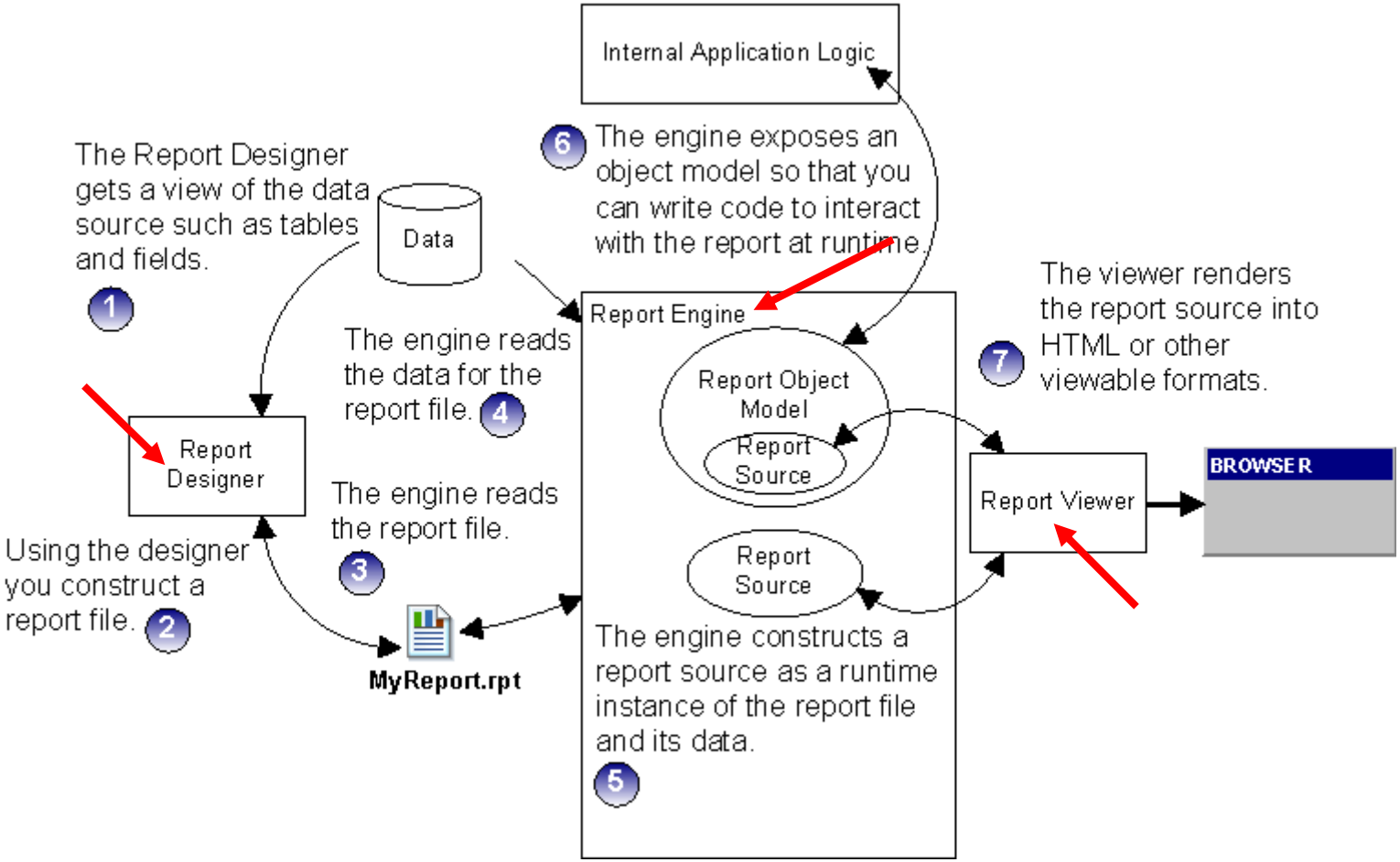
Note: Current maximum number of data rows is limited to 500. (Click to change)

className	lineCount	estimated
Class1ONE1	10	11
Class2	20	22
Class3	30	33



Layout | Master Page | Script | XML Source | Preview

Crystal Reports



CR Reporting Models

- CR Embedded Reporting Model
  - uses the Java Reporting Component (JRC) and Crystal Reports Viewers Java SDK
    - to enable users to view and export reports in a web browser
    - functionality required to create and customize a report viewer object, process the report, and then render the report in DHTML.
  - The JRC (jars) keeps report processing completely internal to the Java application server to process Crystal Reports report (.rpt) files within the application itself, no external report servers

- CR Enterprise Reporting Model
  - uses the Crystal Enterprise Java SDK to leverage an external Crystal Enterprise server
  - additional functionality
    - runtime report creation
    - persisting runtime report modification back to the Crystal Reports report (.rpt) file
    - report management, security, and scheduling
    - The Crystal Enterprise server also improves scalability and increases performance to support extensive user concurrency demands.

Developing Custom JSF Web Components

- **RAD provides for**
  - Importing Custom component Libraries
  - Building Custom JSF Component Library
  - Adding new custom JSF widgets in the library
  - Adding custom library widgets to the RAD palette
  - Sharing and using custom widgets by drag and drop from the palette

```
<h:outputText value="Name:" / >
<h:inputText value="#{person.name}" />
```

```
<my:inputLabel value="#{person.name}" label="Name:" />
```

New Faces Component Library wizard

New Faces Component Library

Faces Component Library

Create a new Faces Component Library

Project name: MyComponents

Contents

Use default

Directory: C:\Program Files\IBM\SDP75\workspace1\MyComponents

Target Runtime

WebSphere Application Server v7.0

Dynamic Web Module version

2.5

Configuration

Faces Component Library

This configuration includes all of the functionality that you require to crea
component library and custom Faces components.

EAR Membership

Add project to an EAR

EAR Project Name: MyComponentsEAR New...

< Back Next > Finish Cancel

New Faces Component Library

Component Library settings

Configure Component Library

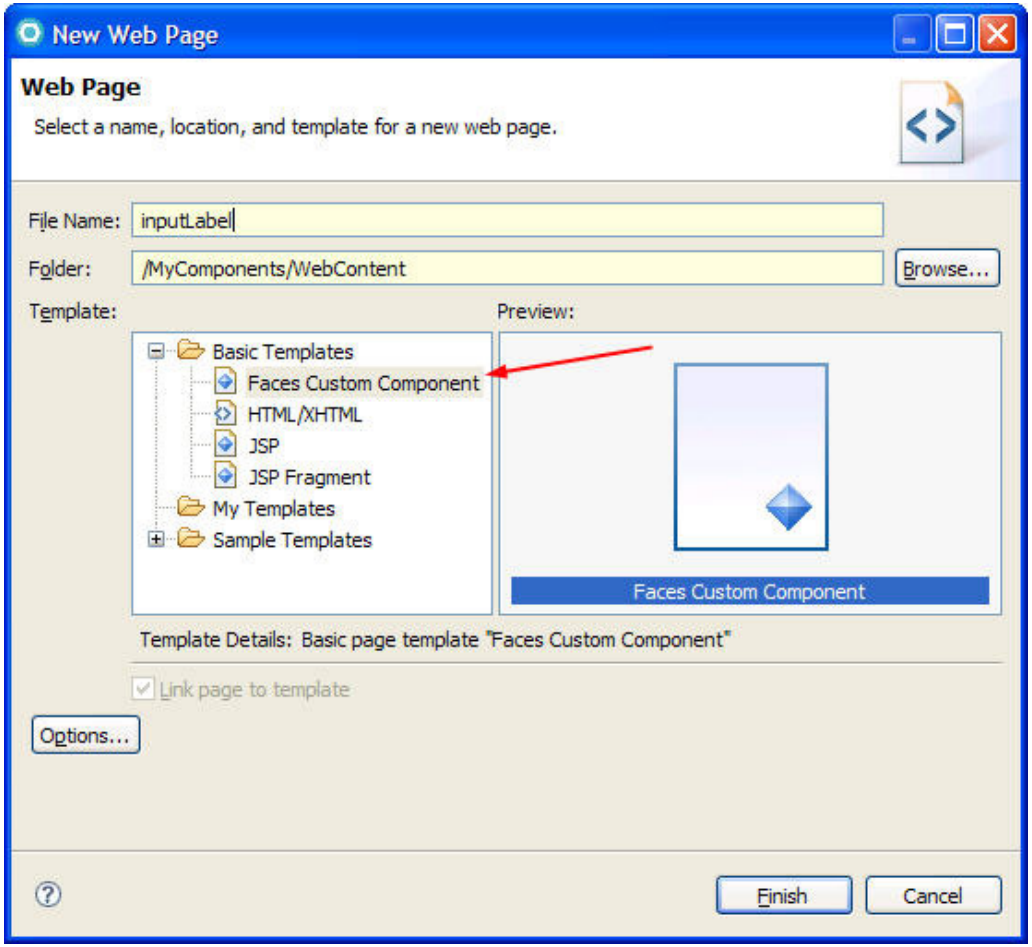
URI for generated tag library: http://ibm.com/jsf/MyComponents

Prefix of generated tags: m

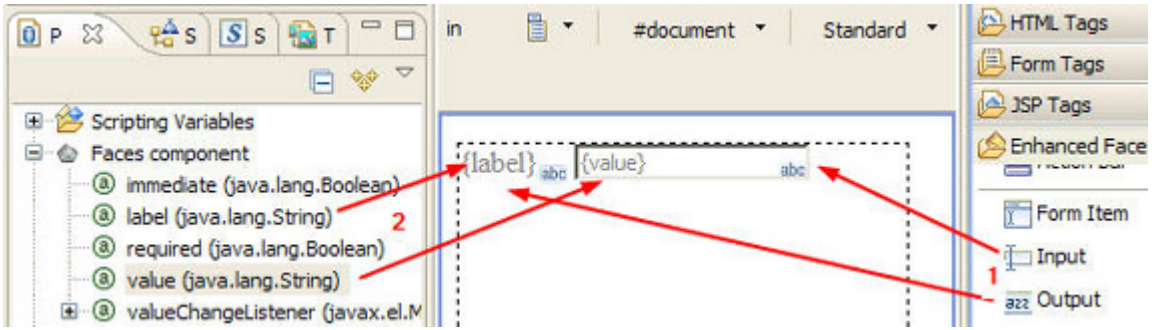
Java package prefix for generated classes: com.ibm.faces.mycomponents

< Back Next > Finish Cancel

New custom component

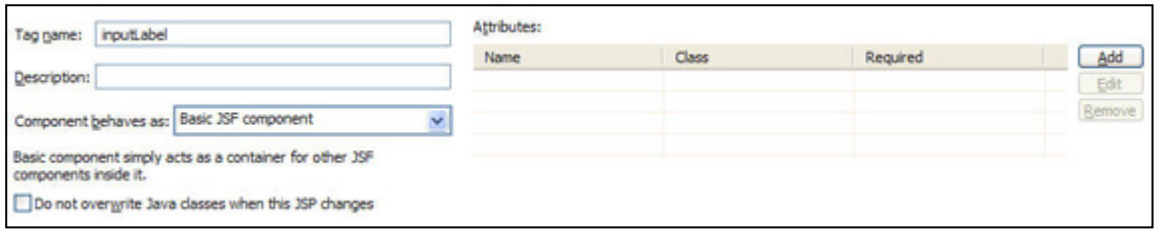


Creating component content

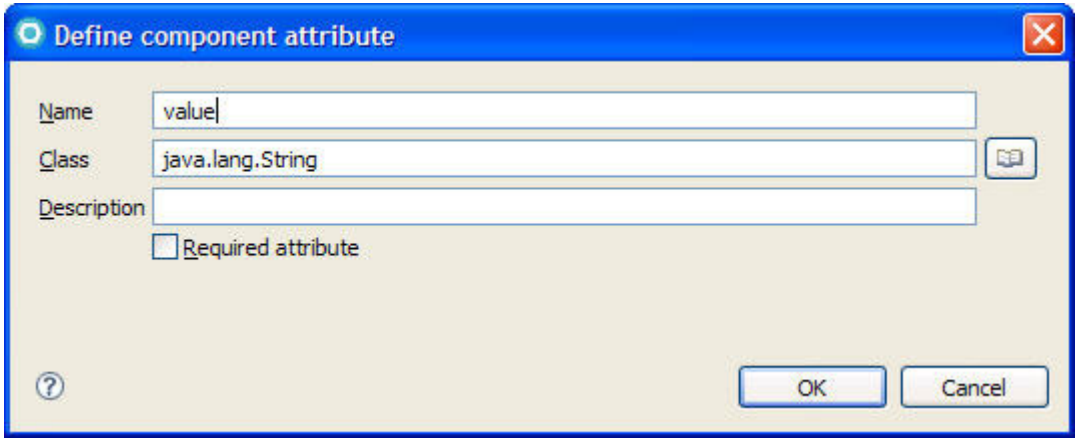


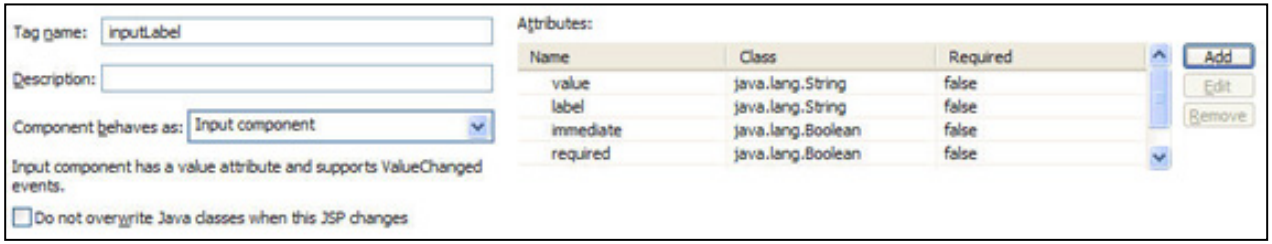
Component source

```
<jsfc:component extends="javax.faces.component.UIInput">
    <h:outputText styleClass="outputText" id="text1"
        value="#{component.label}"></h:outputText>
    <h:inputText styleClass="inputText" id="text2"
        value="#{component.value}"></h:inputText>
</jsfc:component>
```



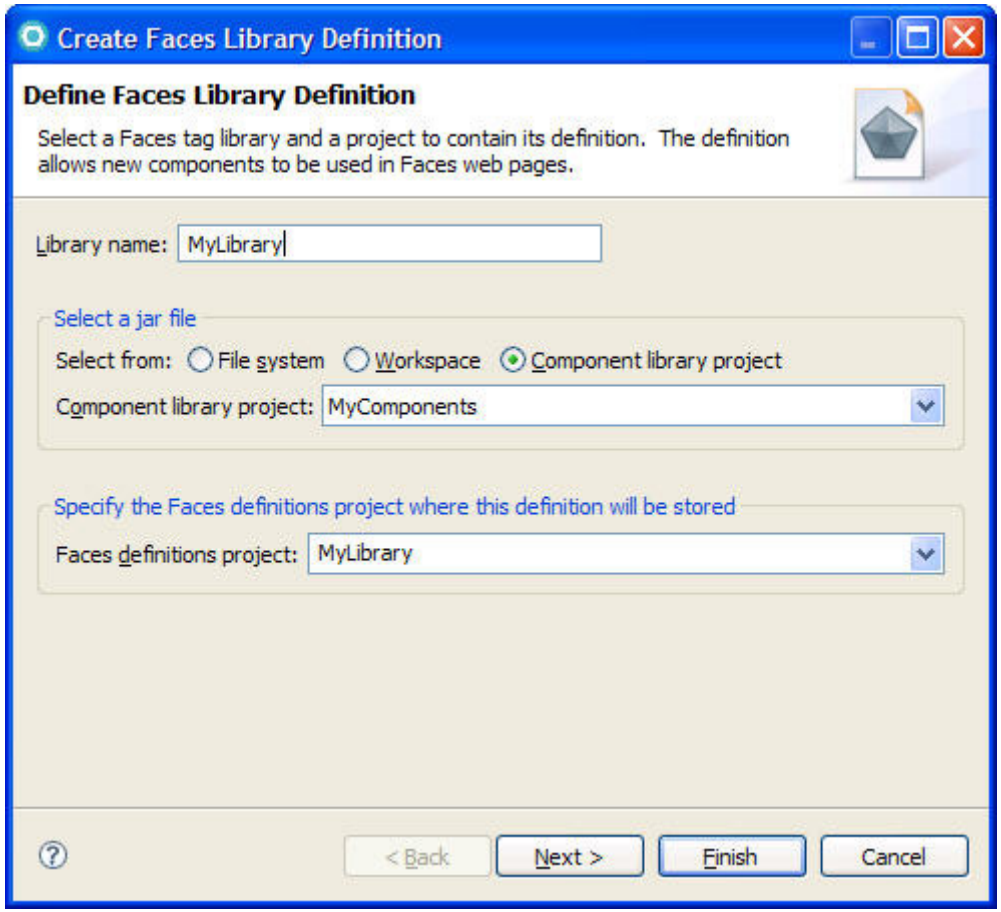
<jsfc:component> Component properties



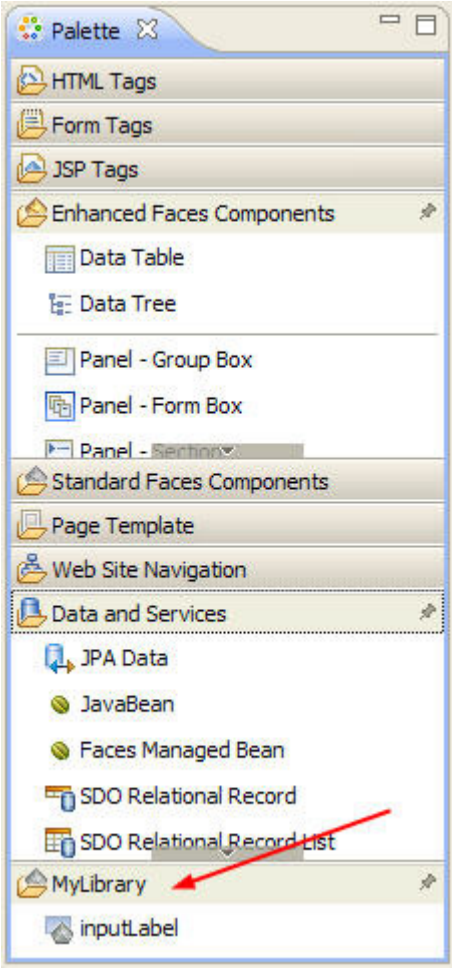


Configured component

Library definition



Added to RAD Palette



Building Visual Custom Tags

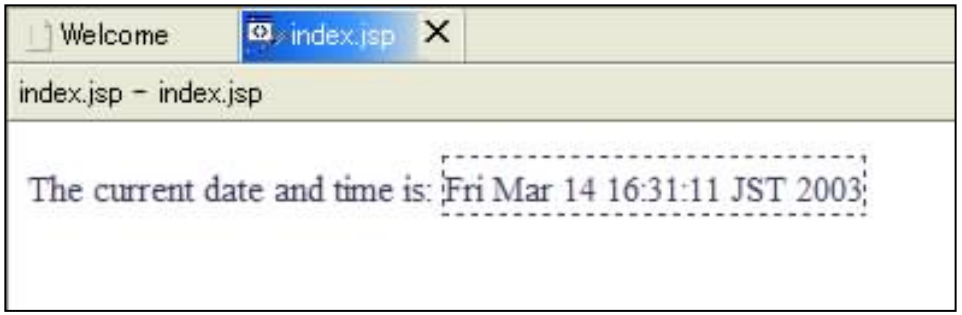
- **Visualizing Custom Tags in the design view**
  - Building custom plug-in to visualize my tag
  - Extend CustomTagVisualizer
  - Provide the visualization information in doStart or doEnd Methods
  - Building and importing new plug-in
  - Add custom properties view for the custom tag

Sample plugin.xml extract

```
<requires>
  <samp><import plugin="org.apache.xerces"/>
  <samp><import plugin="com.ibm.etools.webedit.core"/>
</requires>
<extension point="com.ibm.etools.webedit.core.visualCustomTag">
  <samp><vtaglib uri="/WEB-INF/lib/sample.jar">
    <samp><vtag  name="date"
      class="com.ibm.etools.webedit.vct.sample.DateTimeTagVisualizer"
      description="Dynamically outputs current date and time"/>
  <samp></vtaglib>
</extension>
```

Example Custom tag visualized

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<HTML> <HEAD>

<%@ taglib uri="/WEB-INF/lib/sample.jar" prefix="vct" %>

<%@ page language="java"  contentType="text/html; charset=ISO-8859-1"  pageEncoding="ISO-8859-1" %>
<TITLE>index.jsp</TITLE> </HEAD>
<BODY>
        The current date and time is:  <vct:date/>
</BODY>
</HTML>
```




```
import com.ibm.etools.webedit.vct.*;

public class DateTimeTagVisualizer extends CustomTagVisualizer {
  public VisualizerReturnCode doEnd(Context context) {
            Date now = new Date();
            context.putVisual(now.toString());
            return VisualizerReturnCode.OK;
  }
}
```

Resources

- **RAD**
  - http://publib.boulder.ibm.com/infocenter/radhelp/v7r5/index.jsp
- **JDT**
  - http://help.eclipse.org/ganymede/index.jsp?topic=/org.eclipse.jdt.doc.isv/guide/jdt_int_model.htm
  - http://publib.boulder.ibm.com/infocenter/rtnlhelp/v6r0m0/index.jsp?topic=/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/AST.html
  - http://www.jdg2e.com/ch27.jdt/doc/index.html
  - http://help.eclipse.org/ganymede/index.jsp?topic=/org.eclipse.jdt.doc.isv/guide/jdt_int_model.htm
- **BIRT**
  - http://www.eclipse.org/birt/phoenix/
  - http://wiki.eclipse.org/Integration_Examples_%28BIRT%29
  - http://www.vogella.de/articles/EclipseBIRT/article.html
  - http://download.eclipse.org/birt/downloads/examples/scripting/scripteddatasource/scripteddatasource.html
  - https://www6.software.ibm.com/developerworks/education/dw-r-umlbirtreport/index.html (UML Model reports in RSA)
- **Crystal Reports**
  - http://publib.boulder.ibm.com/infocenter/radhelp/v6r0m1/index.jsp?topic=/com.businessobjects.integration.eclipse.doc.devtools/developer/ArchitectureOverview2.html
- **Plug-in development**
  - http://help.eclipse.org/ganymede/index.jsp?topic=/org.eclipse.platform.doc.isv/guide/firstplugin.htm
- **Web Tools Customization**
  - http://www.ibm.com/developerworks/websphere/library/techarticles/0304_hosokawa/hosokawa.html
  - http://www.ibm.com/developerworks/rational/library/09/0106_kats/

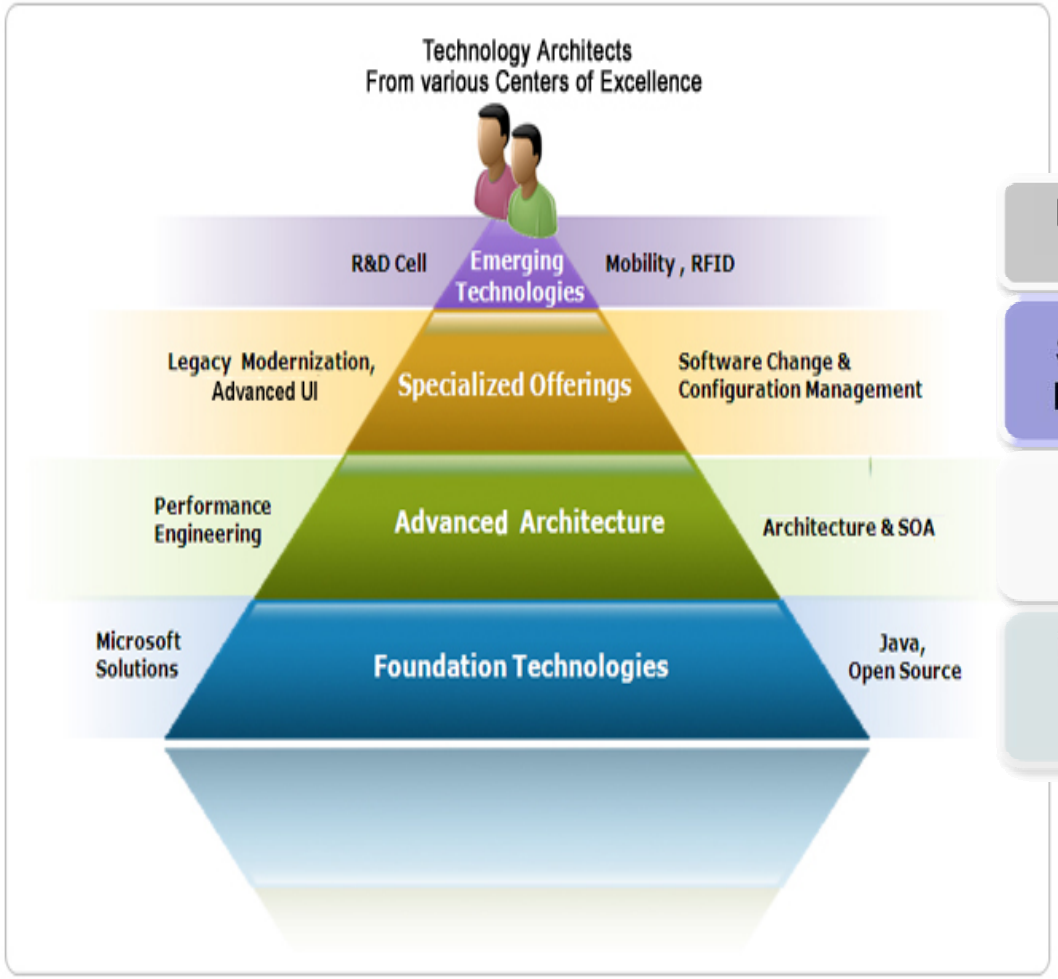
A Case Study @ CTS

- **Extracting Quality metrics from Source code**
  - Using available CTS project metric tools
- **Packaging the custom plug-ins and Integrating to existing QA systems**
  - Modes of running the customized plug-ins
- **Usage tracking**
- **Productivity tracking**

Extending RAD @ CTS…

GTO structure

Cognizant Global Technology Office
Passion for making a difference



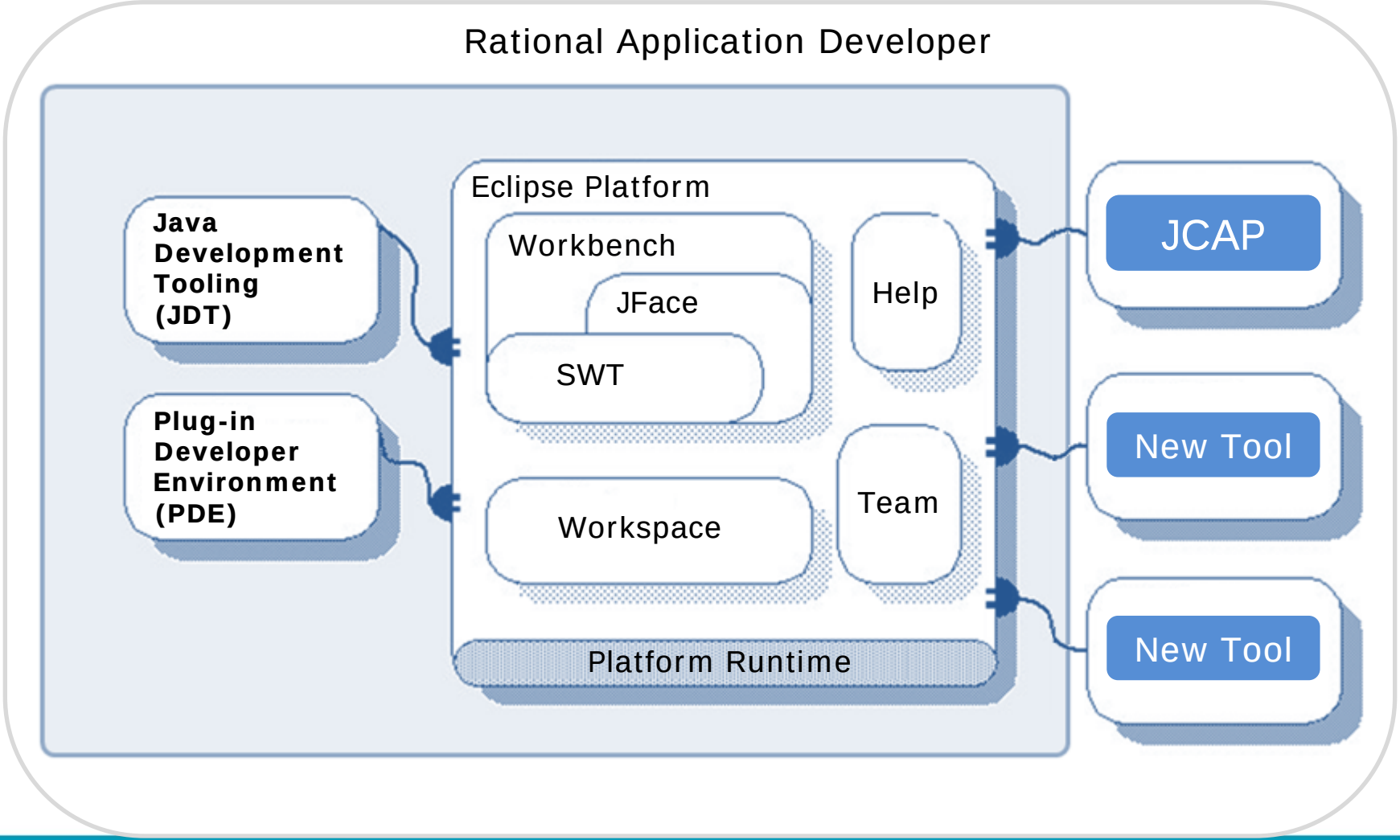
Innovative and differentiating solutions deploying emerging technologies

Solutions focused towards reuse, user productivity and software management

Practice & methodology with focus on building agile, scalable and high performance enterprise architecture

In-depth & end to end coverage of the core technologies with focus on the development, productivity & quality

RAD Extension – JCAP Plug-in



JCAP – Java Code Assessment Platform

Cognizant
Passion for building stronger businesses

Objective

- To build extensions to Rational Products to:
  - » Collect
  - » Analyze
  - » Integrate Code quality Metrics with Cognizant Delivery Platform

Project Quality Management - Need of the Hour

- Project Code Quality Metrics at development milestones, On demand
  - » Overall Project health to be known at build time (entire project code base)
  - » Also Get to know the application health from time to time On demand
- Individual developer's work quality - required for mentoring new recruits/trainees
- Monitor in a continuous basis – decreasing trend or increasing trend

JCAP – Features

Cognizant
Passion for building stronger businesses

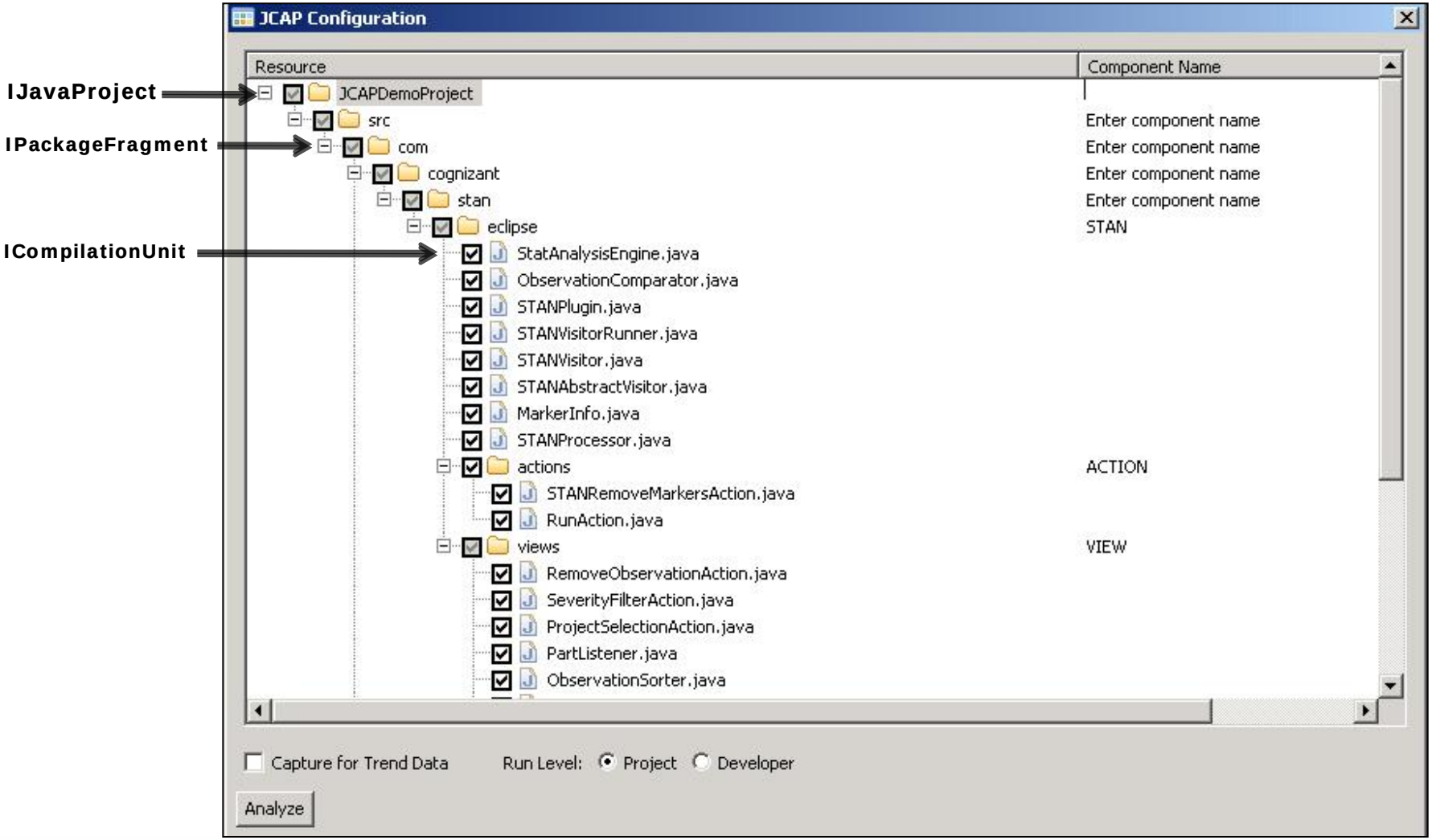
- **J**ava **C**ode **A**ssessment **P**latform (henceforth called as **JCAP**) – implemented as a RAD Extension RAD extensions provided as eclipse plug-ins; extensions to the Menu, Toolbar, Project Explorer and View

- Data Acquisition
  - » Source Analysis & Metrics Capture
  - » Violations against Coding standard rules
  - » Size metrics [Lines of Code and Documentation Lines of Code]
  - » Cyclomatic Complexity
  - » Code duplication
  - » Class coupling

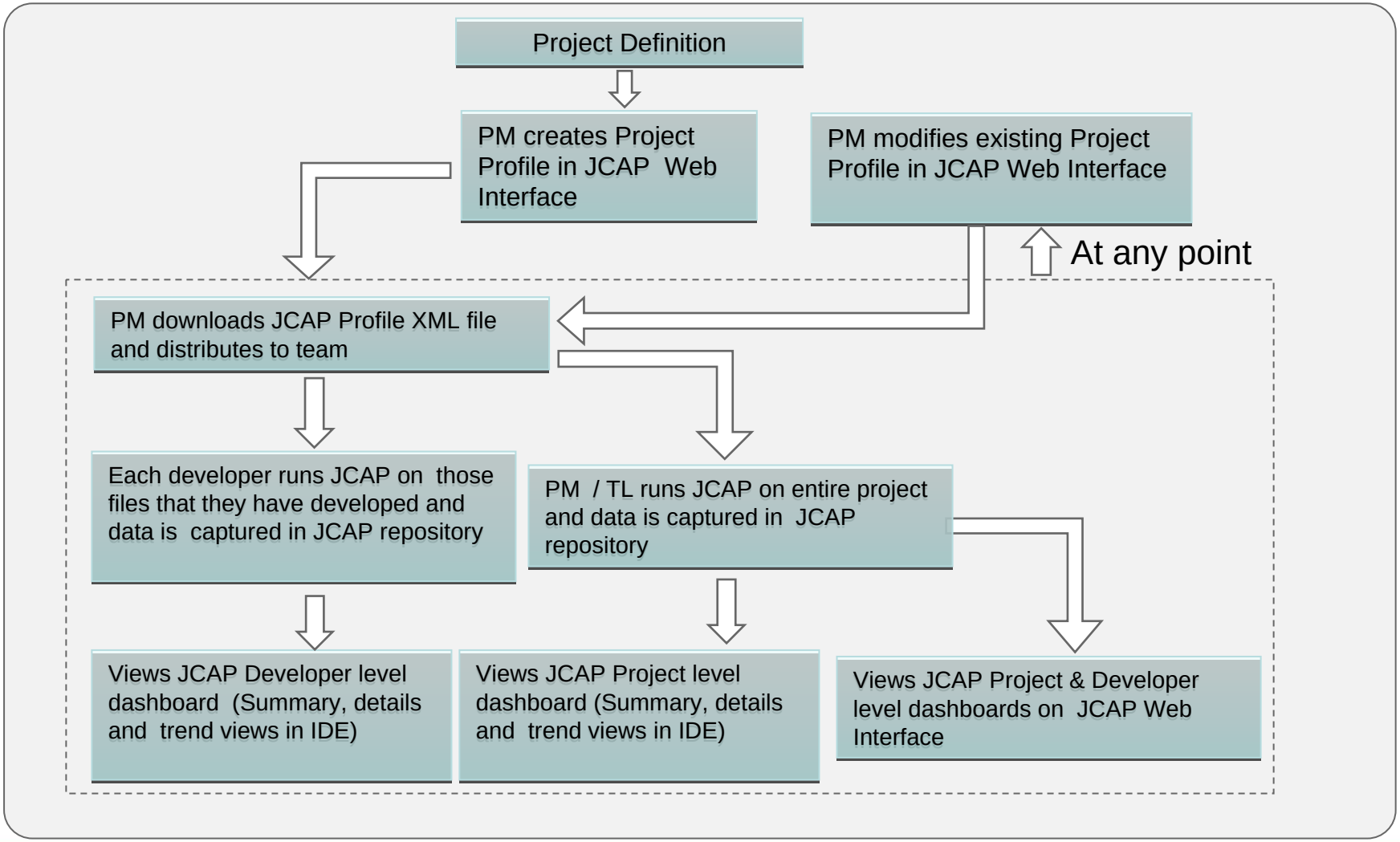
- Data Integration and Reporting
  - » IDE level dashboards
  - » Web dashboards integrated with the Organization Governance dashboards

Cognizant
Passion for building stronger businesses

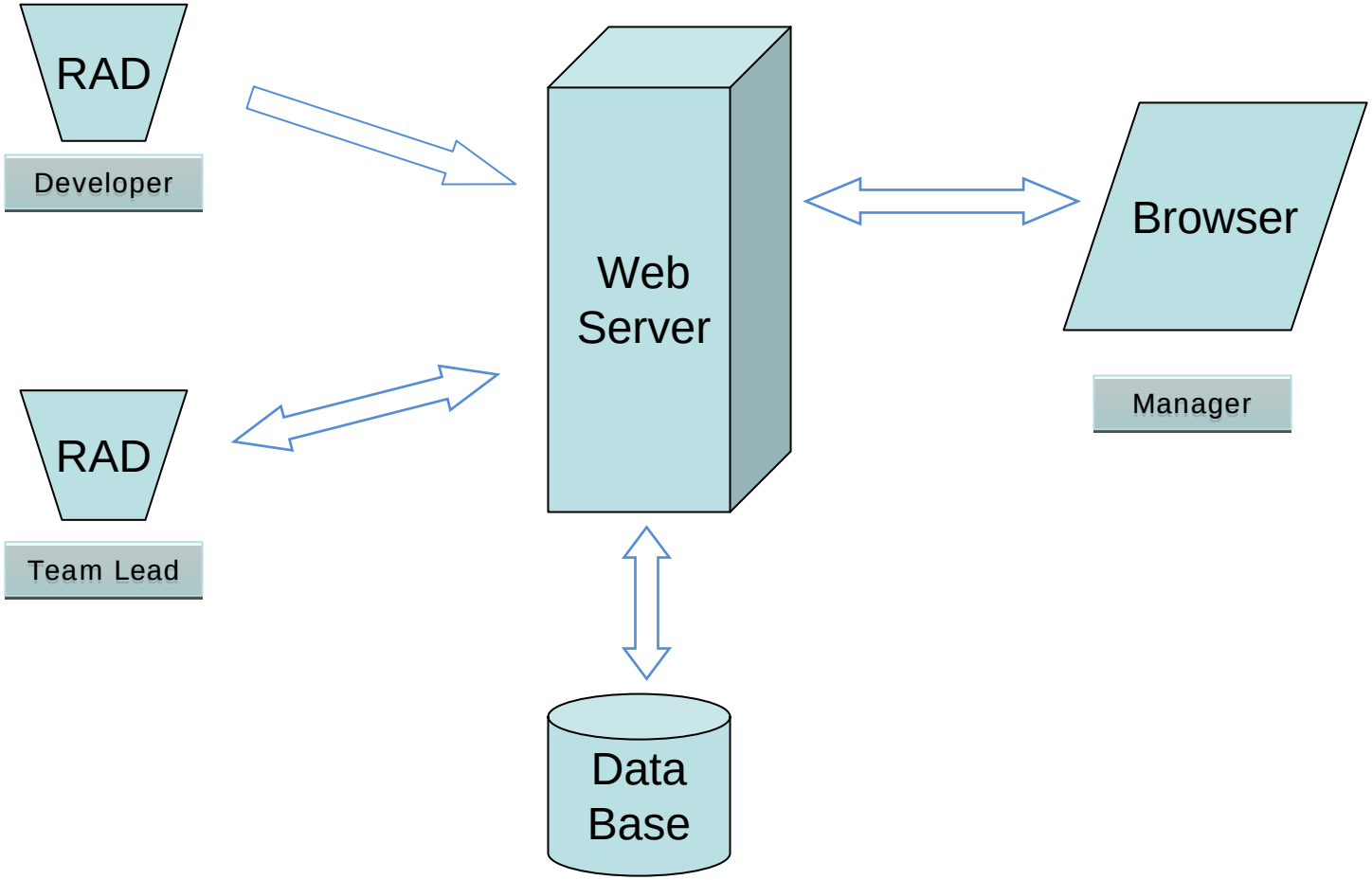
JCAP Plug-in



JCAP – Functional view



JCAP – Architectural View



Data Reporting - Project Quality Dashboard

Cognizant
Passion for building stronger businesses

Code Assessment Platform - Microsoft Internet Explorer provided by Cognizant Technology Solutions

File Edit View Favorites Tools Help

Back Search Favorites

Address http://localhost:8040/CAPWeb1.3/jsp/doLogin.action Go Links

JCAP

Java Code Assessment Platform

Cognizant

Logout

Profile Management | Code Quality

Project ID - 22211 DashBoard Data Collection - Wed, 5 Aug 2009

Select Project 22211 Select Developer Project

Scoring Metrics as on 2009-08-05

Metric Name	Value(per 100 loc)
Inline-doc	99.05
CyclomaticComplexity	11.06
Code Duplication	5.15

PMD Summary as on

#File	LoC	Lines/File	Violation
58	3935	67.84	77

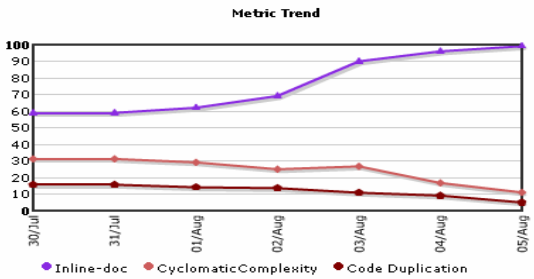
Detail Trend

CPD Summary as on

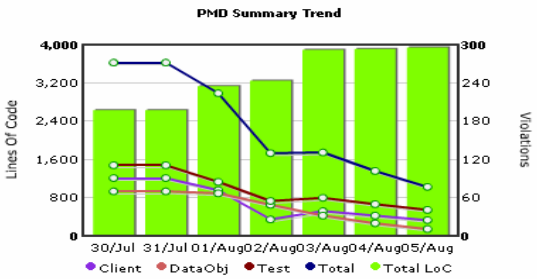
#File	LoC	Lines/File	Duplication
35	2111	60.31	21

Detail

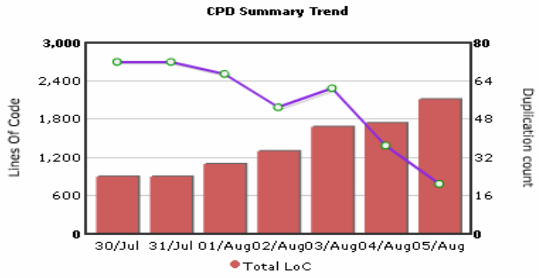
Scoring Metric Trend



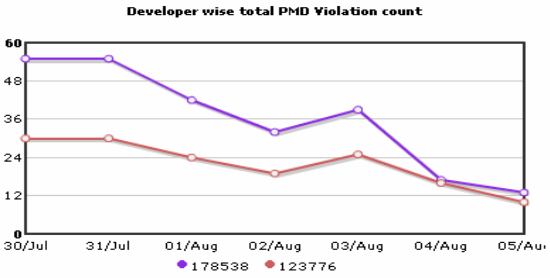
Componentwise Trend | Componentwise Distribution



Code Duplication (Copy-Paste) Count



Developer Violation Trend



Class : C1 - Highly Critical

Done Local intranet


DEMO


Questions