ADF Hands-On

Team Project Considerations

Abstract:

This practice demonstrates how you can use features of JDeveloper and ADF to work collaboratively with a team to build applications

2010 / 2011 ADF Internal Enterprise 2.0 Training

Introduction

In this practice, you create a source code repository and use JDeveloper to check in and manage version controlled files. You also install and configure a team productivity center server for use in team development. Additionally, you will configure the team productivity center extension in JDeveloper and use work tasks to track artifacts of applications. Finally, you deploy and consume ADF libraries across applications.

Prerequisite and Setup

1. Copy the tpc_bundle.zip, tpc-connector-task_bundle.zip, and tpcinstaller.jar onto your hard drive.

Work with code repositories

Create a subversion repository

1. In the versioning menu, choose Create Local Repository
2. Enter c:\svn-repos as the path and LocalRepos as the Connection name. This will be a native file system repository.

Check files into a repository

1. Open the HRApplication that you built earlier, or make a copy of the HRApplication folder (the highest level that contains the workspace and project directories) and open the copy in JDeveloper.
2. From the Versioning menu, choose Version Application.
3. Click the folder icon to create a new directory. Name the directory HRAppTrunk.
4. Ensure the source directory points to the workspace directory. Accept the defaults through the last pages of the wizard and click Finish to add the workspace files to the repository and check them out.
5. You may have to refresh the Application view or even close and reopen the application, but it should appear with icons next to every node to delineate unchanged files, and there should be a “2” next to each of the files, to delineate that this working copy is version 2.

Change a file and check in the change

1. Add a new attribute to the EmployeesView view object
    a. Clean up the editor space by right clicking on any open tab and selecting Close All.
    b. In the ADFModel, open the EmployeesView view object in the editor.
    c. Click the attributes node and click the green plus to add a new attribute.

   d. Name the attribute TotalCompensation and click the Expression value Type. Click Edit to create the expression.
   e. Enter Salary * CommissionPct as the Expression, and shuttle those corresponding attributes to the selected dependent attributes.
2. Save the file and notice that the icon in the application navigator changes to show the file is modified.
3. From the View menu, choose Team > Pending Changes. The pending changes window shows changes you've made to files in the outgoing tab, candidates for addition to the repository (typically new files) in the candidates tab, and incoming changes from your peers in the incoming tab. If there are files from the classes directory in the candidates tab, ignore them. Typically, you would exclude all class files from the repository.
4. Select the EmployeesView.xml file in the Outgoing tab and click the Commit icon.
5. Enter a comment such as "Adding TotalCompensation attribute" and click OK. Notice that the version number on EmployeesView increases to 3. In the History tab for the file, you'll see the author and check-in comments for each revision, allowing you to identify who on the team made which changes.

Use Team Productivity Center

Install Team Productivity Center Server

1. In a command line, set the JAVA_HOME variable to the location of your jdk:

   ```
   set JAVA_HOME=C:\Program Files\Java\jdk1.6.0_21
   ```

2. Set the PATH to include this variable:

   ```
   set PATH=%JAVA_HOME%\bin;%PATH%
   ```

3. Launch the Team Productivity Center Installer by issuing:
   `java –jar tpcinstaller.jar`
4. Specify your database connection and the system username and password
5. Click Server to install the server
6. Enter TPC_Admin as both the administrator username and and password
7. Enter the path to the autodeploy directory of JDeveloper's integrated WLS server:
   C:\Documents and Settings\<yourname>\Application Data\JDeveloper\system11.1.1.4.37.58.45\DefaultDomain\autodeploy
8. Click Install
9. Re-launch the Team Productivity Installer.
10. Specify your database connection and the system username and password
11. Click Connectors to install connectors.

12. Choose Install from local file and browse for tpc-connector-task_bundle.zip
13. Exit the installer when finished. Most connectors require an installation on the team productivity center server as well as in each user's JDeveloper installation, but they both use the same extension bundle. For example, if you wanted to create a connection to a JIRA or Bugzilla extension, you would install those connectors as in the previous step, and you would also install them in JDeveloper as shown in the next step.

Install Team Productivity Center and connectors

1. In JDeveloper, choose Help-->Check for Updates.
2. In the Welcome to the Check for Updates Wizard page click Next.
3. In the Source page, select Install From Local File and choose tpc_bundle.zip.
4. Restart JDeveloper when prompted. Click No if prompted to migrate preferences.
5. Repeat these steps for tpc-connector-task_bundle.

Connect repositories to Team Productivity Center

1. Connect to the Team Server from JDeveloper
    a. From the Run menu, choose Start Server Instance
    b. From the View menu, choose Team > Team Navigator
    c. Click Connect to Team Server
    d. Enter localhost or 127.0.0.1 as the Team Server and 7101 as the port.
    e. Deselect Use SSL Connection and select Remember password
    f. Enter TPC_Admin as the username and password and click Connect
2. From the Team Navigator menu choose Team Administration
3. In the Users tab, click New to enter a new user with your first and last name and login TPC_Admin. Provide an email address and select the Administrator permission.
4. In the Teams tab, create a new team named Dev Team. Add yourself to this team as a Group Administrator.
5. In the Repositories tab, add repositories for use by any team
    a. Create a new Versioning repository using the Subversion connector. Name the respository Local SVN. You'll supply a server connection for an individual team.
    b. Create a Work Item repository using the Task connector. Name the repository HRApp Task Repository. Click the green plus to add a new repository server. Accept the default name "New Repository Server" and click OK.

6. In the Teams tab, click the Team Repositories tab to add these repositories to the Dev Team.
    a. Select the Local SVN repository and in the Value field for Server URL, enter file:///C:/svn-repos/HRApp_Trunk
    b. Select HRApp Task Repository
7. Disconnect as TPC_Admin and connect as the new user you created.
    a. From the Team Navigator menu choose Disconnect.
    b. Click Connect to Team Server and connect as the new user.
8. Expand the Work Items and Versioning nodes. This is your team space that contains all the artifacts for the project. Note that you can use versioning without team productivity center, the Versioning navigator will look the same as in the Versioning accordion of the Team navigator.

Work with tasks in Team Productivity Center

1. Now, pretend you are an analyst in QA and you have your own connection to the team server and have checked out the application for testing.
    a. Right click HRAppModule and choose Run to test the application module.
    b. Double click the Departments node. Navigate to department id 80 and double click EmployeesbyDepartment to view the new attribute, TotalCompensation.
    c. Return to the Departments tab and navigate to the next department. You'll get a NullPointerException. As a QA analyst, you should file a bug for this defect.
2. We could have installed and configured a connection to Bugzilla or JIRA, or even the Oracle Bug Database, but these require valid logins for those systems. Instead of filing a bug, create a new task in the task repository.
    a. In the Work Items accordion, right click the HRApp Task Repository and choose New Task.
    b. Name the task Defect: NPE when navigating off of sales department in AM tester.
    c. Assign the task a priority and include a description.
    d. Assuming that you still have the EmployeesView.xml file open in the editor, click the icon in the task tab to Save Context.
    e. Close all the open files in the editor and save the task when prompted.
3. Now, assume you are the developer. Query for new tasks.
    a. Right click the My Queries node of the Task Repository and choose New Query.
    b. Query for Tasks assigned to you

c. Click Save to save the query. This should be a user query and you can name it My HRApp Tasks.

d. Double-click the result to bring up the task. Click Make Active. This might give an error that you can safely ignore.

e. Click the icon in the task tab to Restore Context. This brings up the EmployeesView.xml object. This is a great way to remind yourself which files you were working on when you last looked at this task.

f. You realize that you haven't made the groovy expression check for null values.

   i. Double click the TotalCompensation attribute to bring up the editor

   ii. Edit the groovy expression to read (Salary!=null?Salary:0) + (Salary * (CommissionPct!=null?CommissionPct:0))

   iii. This time, check your work before checking in the change. Run the application module and verify the NPE is avoided.

g. In the Pending Changes window, click Commit to commit the modified EmployeesView file. Add checkin comments and click Associate with Workitems to associate the checkin with the active work item.

h. Open the task and notice that the Changes node now contains an entry for the change to the EmployeesView file. Change the status to Done and save the task.

Use Team Productivity Center

To Do: Create a reusable component

Create an ADF Library

1. Open EmployeesView in the editor. In the Property Inspector, set Library Private to true and save the file.
2. Double click ADFModel to bring up the Project Properties. Click the Deployment node and choose New.
3. Select ADF Library JAR file as the archive type and change the name to BaseHRApp.
4. Right click ADFModel and choose Deploy > BaseHRApp. Click Finish to deploy the JAR file.
5. In the Resource Palette, expand the IDE connections accordion and choose New Connection > File System from the folder icon dropdown.
6. Enter BaseApp as the connection name and set the directory path to the ADFModel\deploy directory. Test the connection to verify.

7. Close the HRApplication workspace.
8. Now, pretend you are a member of the UI team and you will build a web application based on the model objects created by your counterparts in the service team. Create a new Generic Application named TestApp with a default project named Project1.
9. In the Resource Palette, expand BaseApp. Right click BaseHRApp.jar and choose Add to Project. Click Add Library to confirm.
10. In Project1, create a new JSF page named untitled1.jspx.
11. Expand the Data Controls accordion and notice that the ADF data controls are available from the ADF library.
12. Expand the BaseHRApp.jar, Business Components, and adfmodel.queries nodes in the Resource Palette. Note that the EmployeesView object is not visible because it was marked as private.

Conclusion

Team Productivity Center allows teams to integrate with industry-preferred ALM vendors and enable developers to work collaboratively. Additionally, the built-in source control support allows teams to manage all their application versioning needs from within JDeveloper, without requiring a separate client. Finally, ADF libraries enable reuse across teams and projects.

RELATED DOCOMENTATION

⊠	TPC on OTN: http://www.oracle.com/technetwork/developer-tools/tpc/index.html
⊠	
⊠	