Collaboration using Rational Rhapsody and Rational Team Concert

Rohan De Noronha

November 28, 2011

Introduction

The purpose of this white paper is to take you through the process of integrating Rational Rhapsody with Rational Team Concert. This white paper will demonstrate the collaboration aspects between the two tools in terms of parallel and agile development.

This white paper brings to you the capabilities of the two products. It demonstrates how to use the complementing capabilities from these products to establish an overall collaborative development environment.

In a development environment, having to work with a source control tool is as important as developing the software itself. You never know when you might be forced to revert to a previous version. While Rhapsody provides for development of the software, Rational Team Concert and its Eclipse Client can be used to get source control capabilities. Additionally, the various functionality provided by Rational Team Concert makes collaboration amongst developers easy.

This white paper takes a generic scenario and provides you with all the guidelines on setting up and configuring the integration, using work items and Rhapsody perspectives within Rational Team Concert.

The Project areas and Team areas of Rational Team Concert provide a way of teaming people together by managing, sharing, and collaborating on projects. At the same time, you can authenticate and authorize the right users on the right actions using the Project management concept. In this white paper, you will see how it is done using a process template.

If you are new to Rational Team Concert, see the *Key Rational Team Concert Definitions and Terminology* section whenever you encounter a concept with which you are not familiar.

Key Tool Capabilities to the Collaboration

Rational Team Concert is a collaborative development environment that connects teams to simplify, automate, and govern software delivery in real time. It supports the work of developers, architects, and project managers by providing a fully integrated change, configuration, and build management software delivery environment. It also provides integrated agile project planning and dashboards with metrics and reporting.

Rational Team Concert is the first offering of IBM Rational based on the Jazz platform that enables real-time collaboration. Such collaboration enables software teams to be more collaborative, transparent, and productive. Highlighted below are some of the Source Control features:

- Change sets created to resemble contributions to a project.
- Streams created to follow a synchronized pattern of work between members in an extended team.
- Parallel development with respect to Repository workspaces.

Rational Rhapsody is a model-driven development solution that improves productivity, improves quality, increases communication, and helps validate development projects through model-based design, simulation and automated testing.

The Rational Rhapsody Platform integration with Rational Team Concert provides a highly integrated, collaborative development environment. This is an integration with defect tracking, integration with builds, and process-centric automation allowing software developers to work on a Rational Rhapsody project within the Eclipse platform.

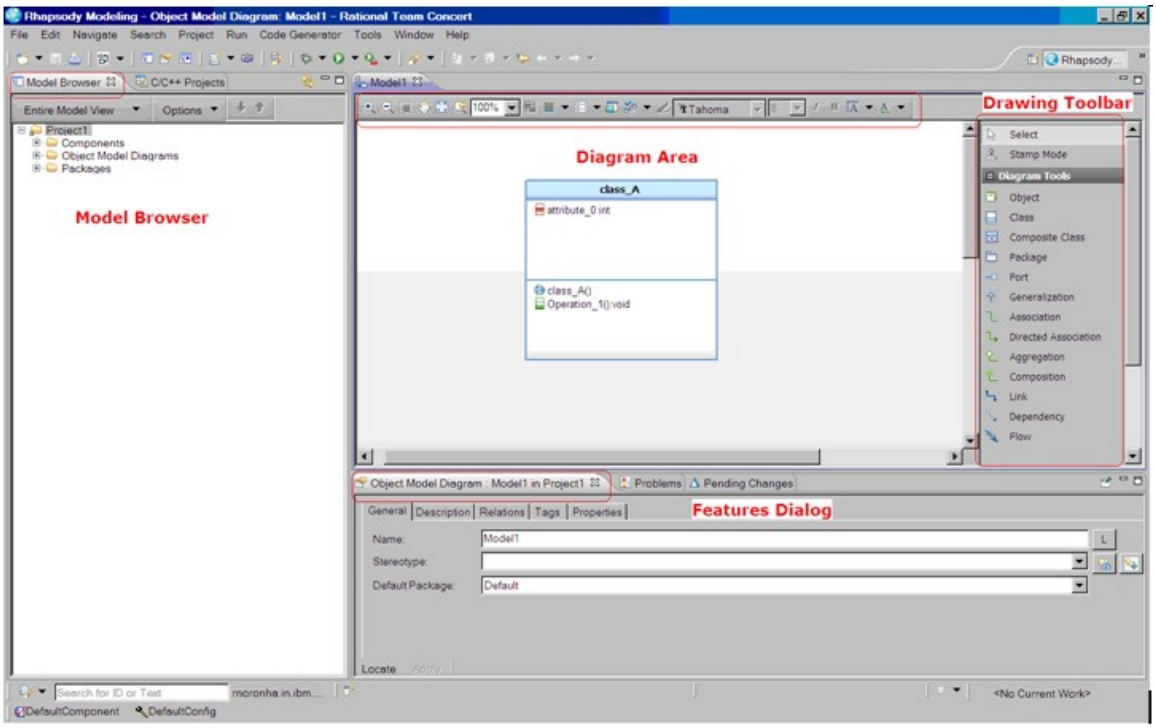
Introducing the workbench perspectives

Perspectives control the menu and toolbar items appearance on the screen. They define visible action sets, which you can change to customize a perspective. In this collaboration you require work with the Rhapsody modeling perspective and Work-Items perspective

You can switch between the perspectives using the **Open Perspective** option in the **Window** menu toolbar.

The Rhapsody Modeling perspective in the Team Concert client

This perspective view includes the model browser, diagram area, drawing toolbar, and the model element features dialog.



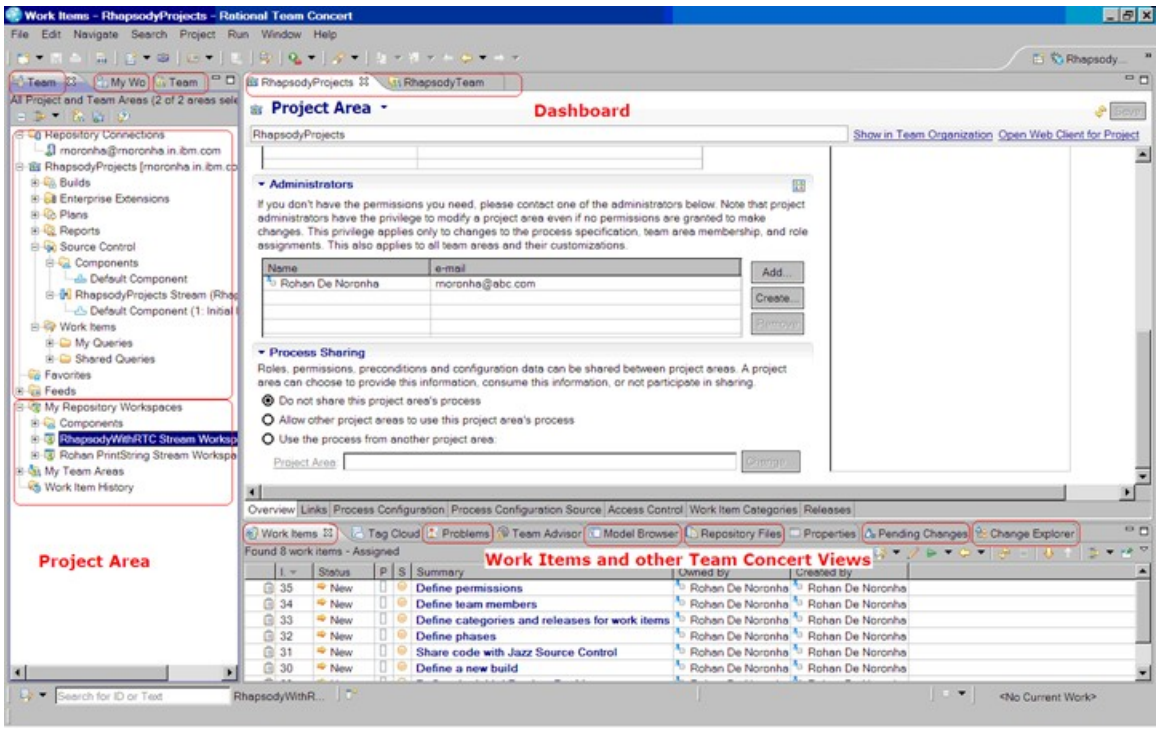
Rhapsody Modeling perspective

This perspective allows you to work in Eclipse as if you were working directly on Rhapsody itself:

- Make Changes to Model in the browser and Diagram.
- Refresh browser to view incoming changes.
- Check-in and Deliver changes done to the Model.
- Accept Changes done to the model.
- Invoke the Diff-merge utility to view change set contents.

Work Items perspective in the Team Concert client

The Work Items perspective includes these views: Work Items, Team Artifacts, Team Dashboard, My Work, and Tag Cloud. All these views provide tools to help you create, triage, and work on work items.



Work-Items perspective

A Work-Items perspective provides Project Management capabilities in terms of:

- Creating and displaying work items and associating with users.
- A Team Artifact view for creating/displaying repository connections, Project Areas, Repository workspaces,
- A Team Dashboard view tracking events in the connected Project Area, displaying descriptive information

Presenting the Scenario

Here is a scenario used throughout the paper to describe various setup configurations and capabilities of the collaboration.

Software developers spread across geographically distributed teams need to collaborate with each other towards the development of a project. They need to track and understand the updates and changes taking place in the project while simultaneously contributing to its development.

A project is initially created in Rhapsody and stored in a central repository called the Jazz Source Control by the Project Administrator or Project Manager. Users within the team load the project into their local sandbox using the Team Concert Eclipse client and integrate with Rhapsody to perform these tasks:

- Create modules and share by delivering the changes to the project main stream.
- Modify existing modules and deliver to the project main stream.
- Synchronize with the project main stream to fetch changes done to the project courtesy of other team members.
- Accept changes from the project main stream and incorporate them in the local workspace whenever applicable.

All of the activities performed in this collaboration correspond to these versions of the tools:

- IBM Rational Rhapsody V7.6
- IBM Rational Team Concert Eclipse Client V3.0

Configuration of the Environment

Topics touched upon in this section include the integration of Rational Team Concert and Rational Rhapsody through the installation of eclipse plug-ins. Topics also include an overview of the Rhapsody modeling perspective, the Work-Items perspective in the Team Concert workbench, and Jazz project administration. Jazz product administration includes defining Jazz repository connection, project area, team area, users, permissions, client licenses, and roles.

Finally, this section covers the topic of creating and sharing a Rhapsody project to a central Jazz repository so it is accessible to authorized users of the Project Area.

Integrating Rational Rhapsody and Rational Team concert

Pre-requisites

- Rhapsody pre-installed with an active license.
- Jazz Team Server pre-installed.
- Rational Team Concert Eclipse client pre-installed.

Establishing the Integration

A Rhapsody Platform Integration with Rational Team Concert is established by the installation of Rhapsody Eclipse plug-ins into the Team Concert Eclipse client.

1. Open Rational Team Concert.
2. Open the Software Updates and Add-ons window. Select **Software Updates** in the Help menu.
3. Under the **Available Software** tab, click the **Add Site** button.
4. In the Add Site window, click the **Local** button.
5. In the **Browse For Folder** window, navigate to **<Rational Rhapsody installation path>\Eclipse**. Click **OK**.
6. In the Add Site window, click **OK**.
7. In the Software Updates and Add-ons window:
    a. Expand **<Rational Rhapsody installation path>\Eclipse**.
    b. Expand the **Model Driven Development** category.
    c. Select the applicable integration check boxes. For example, check IBM Rational Rhapsody Platform Integration.
    d. Click the **Install** button. Complete the installation process by responding to the last few prompts.
8. Restart Rational Team Concert if needed.

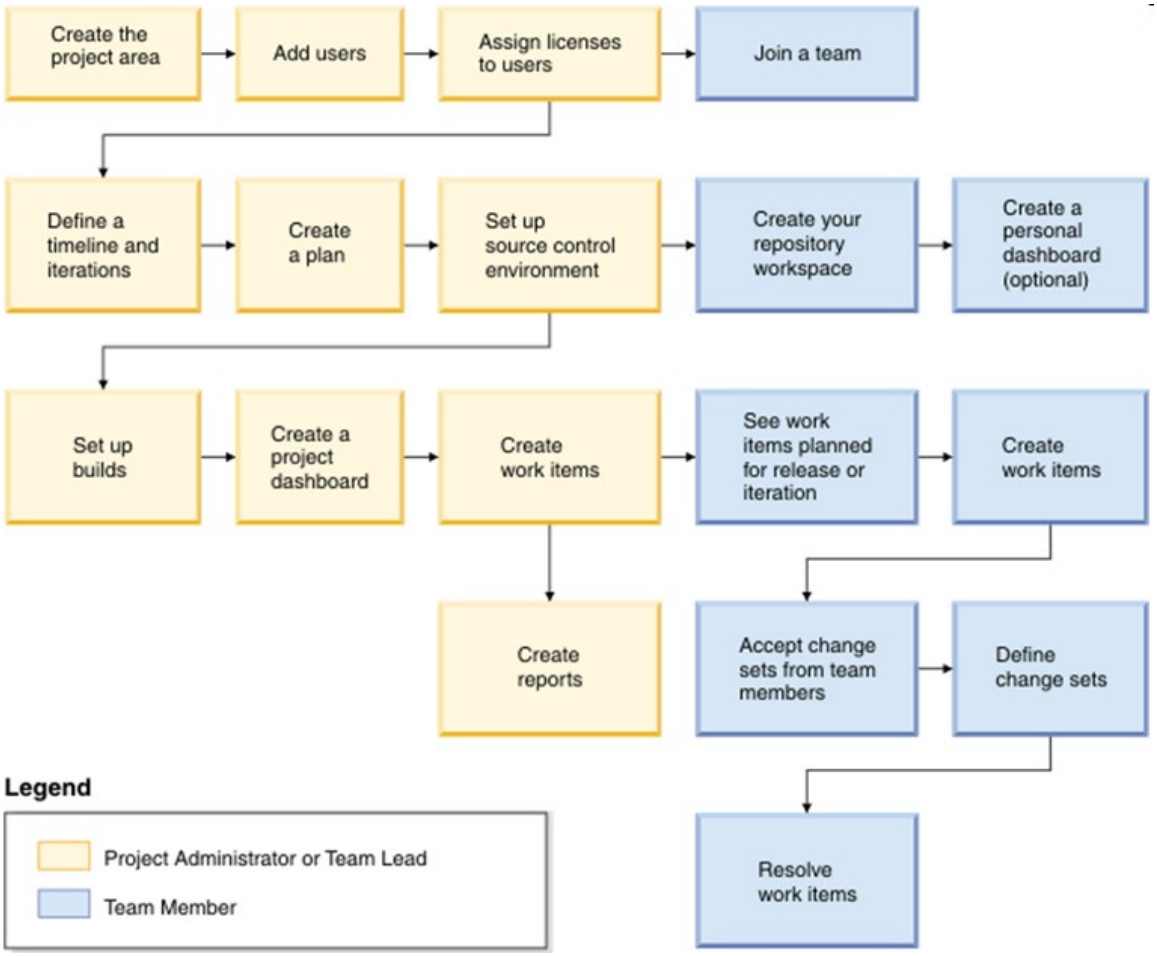
To confirm you installed the Rational Rhapsody plug-ins, open the **About Rational Team Concert** window. The Rational Rhapsody logo is on the window if the plug-ins are installed.

Jazz Project Area Administration

This involves basic administrative tasks. Using a Jazz wizard the administrator will connect to the server and create a project area, which defines the project to Jazz.

A team area is used to identify the team members in the project and everything else you need to use Jazz. Team members in Jazz are known as "users". A process is defined for the project area using a pre-defined process template supplied by Jazz. The process you choose will initialize your Jazz configuration with everything you need to try out other Jazz components like work items and Jazz source control.

The below image gives a complete overview of the tasks from the administrator perspective and user perspective.



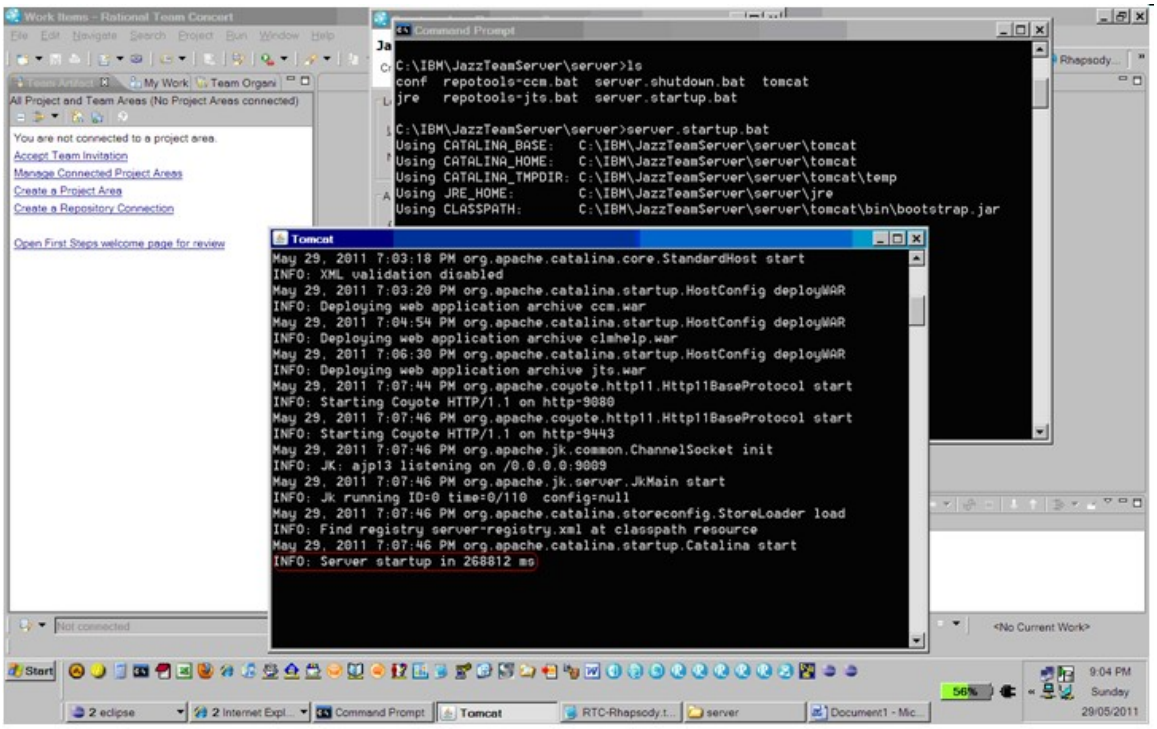
Project administration overview

There is a brief description of the administration key tasks below. Once the administrator completes the initial Jazz Team Server setup through the web interface and defined a public URI, the next tasks can be performed. See the *Post-installation tasks* topic in the Rational Team Concert Information Center for initial setup guidance.

Start the Jazz team server

To start the server from the Windows Start menu: Click **Start > All Programs > Jazz Team Server > Start Jazz Team Server**.

To start the server from the command line: Navigate to the **<JazzInstallDir>/server** directory. Run the command **server.startup.bat**

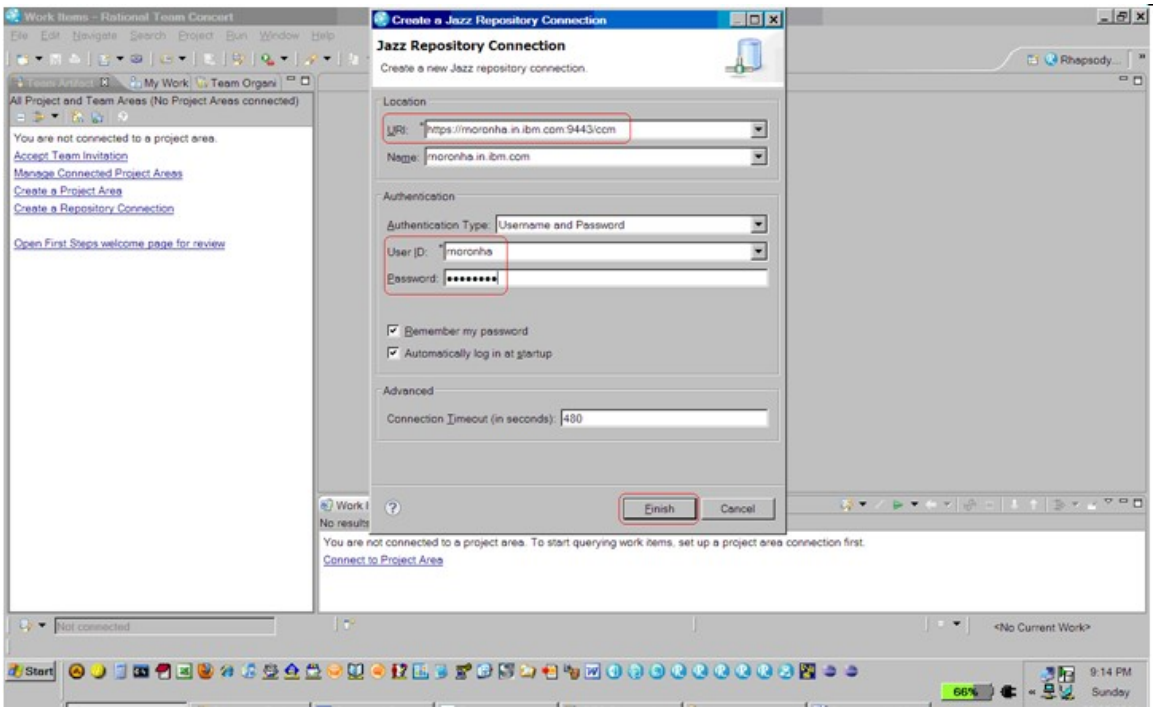


Start the Jazz Team Server

Create a Jazz repository connection

Users must create a connection from the client to a repository for access to project areas. The Jazz administrator must supply information to enable your access to the repository:

1. Click the **Create Repository Connection** link in the Team Artifacts view. Otherwise, right click inside the Team Artifacts view. Select **Jazz Repository Connect** in the **New** menu.
2. In the Create a Jazz Repository Connection window, enter a Location URI. For example: **https://hostname.exampleuri.com:9443/ccm**
3. In the Authentication section, the default Authentication Type is **Username and Password**.
4. Click **Finish**. The repository connection is available in the **Team Artifacts Repository Connections** folder.

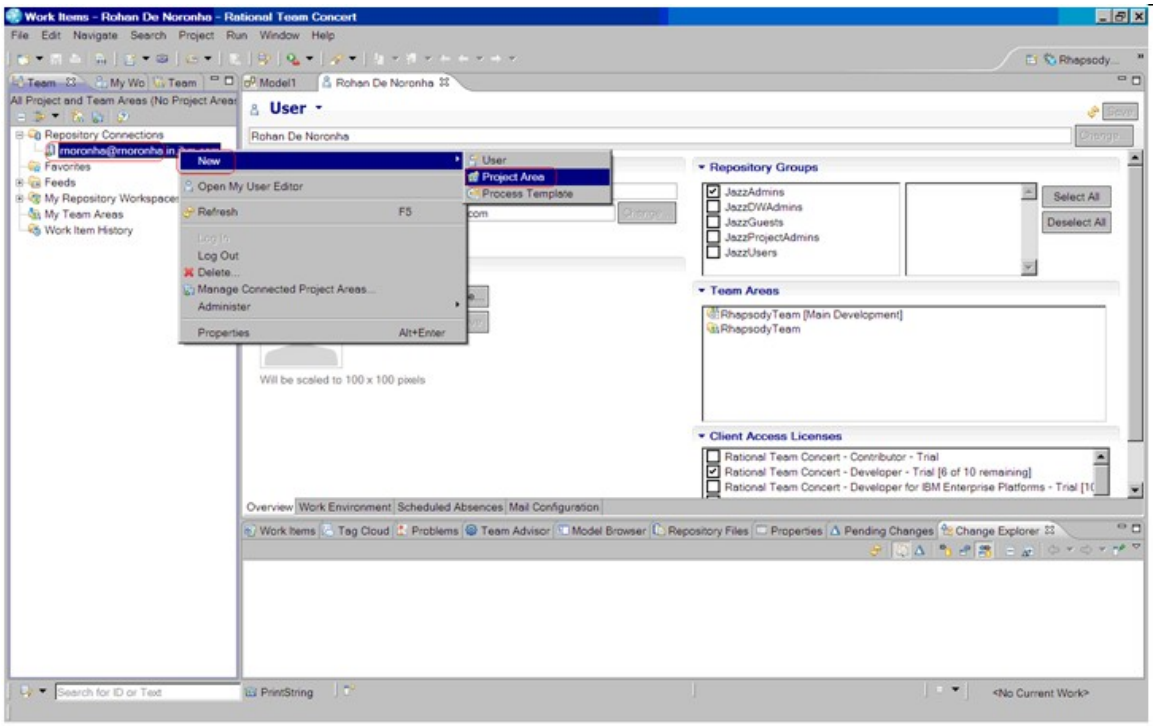


Create a Jazz repository connection

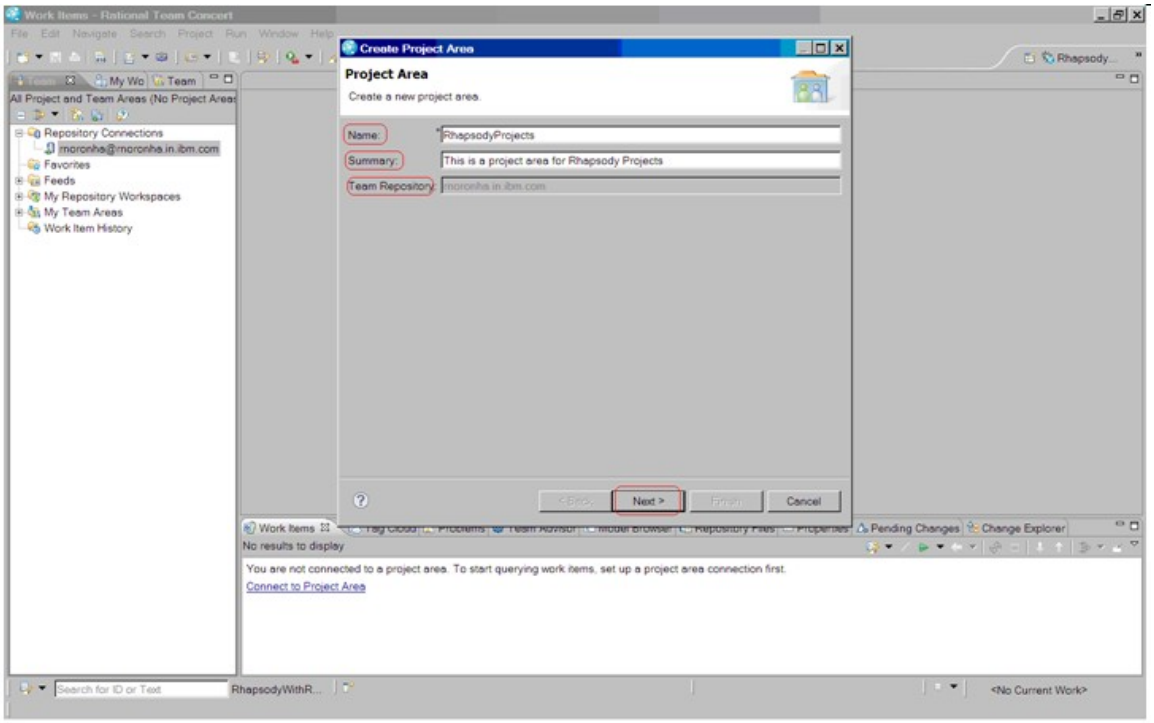
Create a Project area and deploy a process template

A project area is created in a repository for managing the project deliverables, team structure, process, and schedule. A process template provides a new project area with an initial process configuration and iteration structure. You can use one of the predefined process templates, use a process template that your team has created, or use a template that has been imported into the repository.

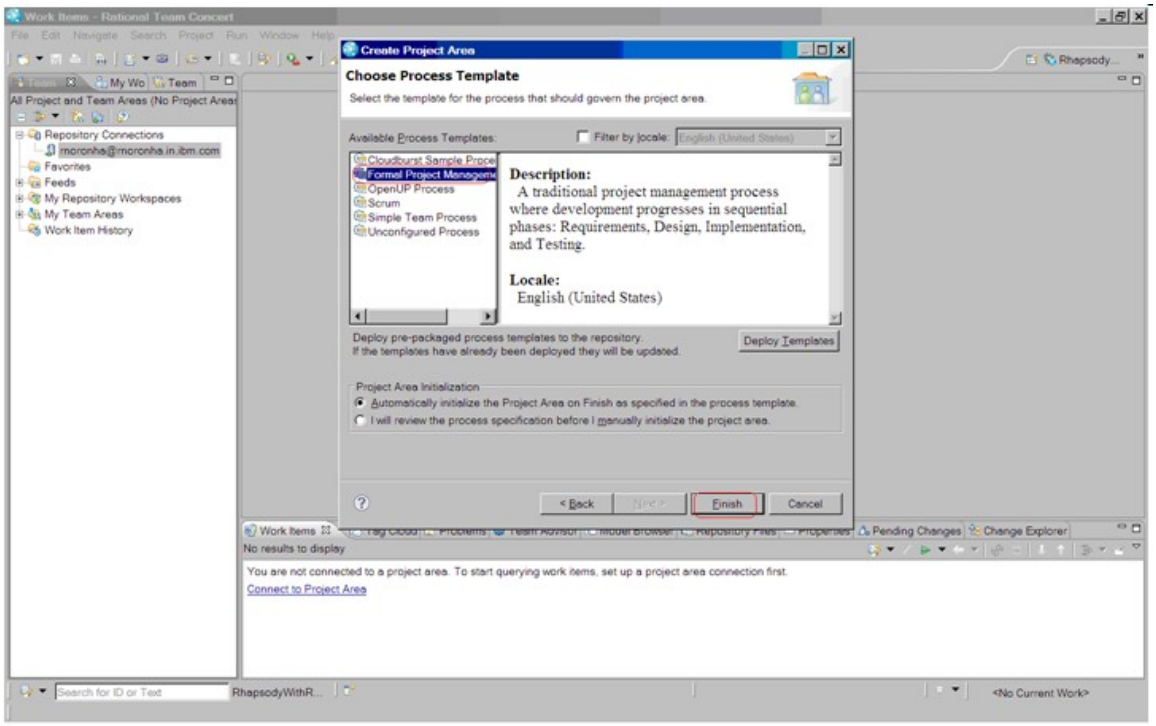
1. Right click a repository connection in the Team Artifacts view. Select **Project Area** in the **New** menu.
2. In the Create Project Area window, enter a name and optional summary for the project area. Click **Next**.
3. Deploy a process template. Click **Finish**.



Create a new Project Area



Enter Project Area information

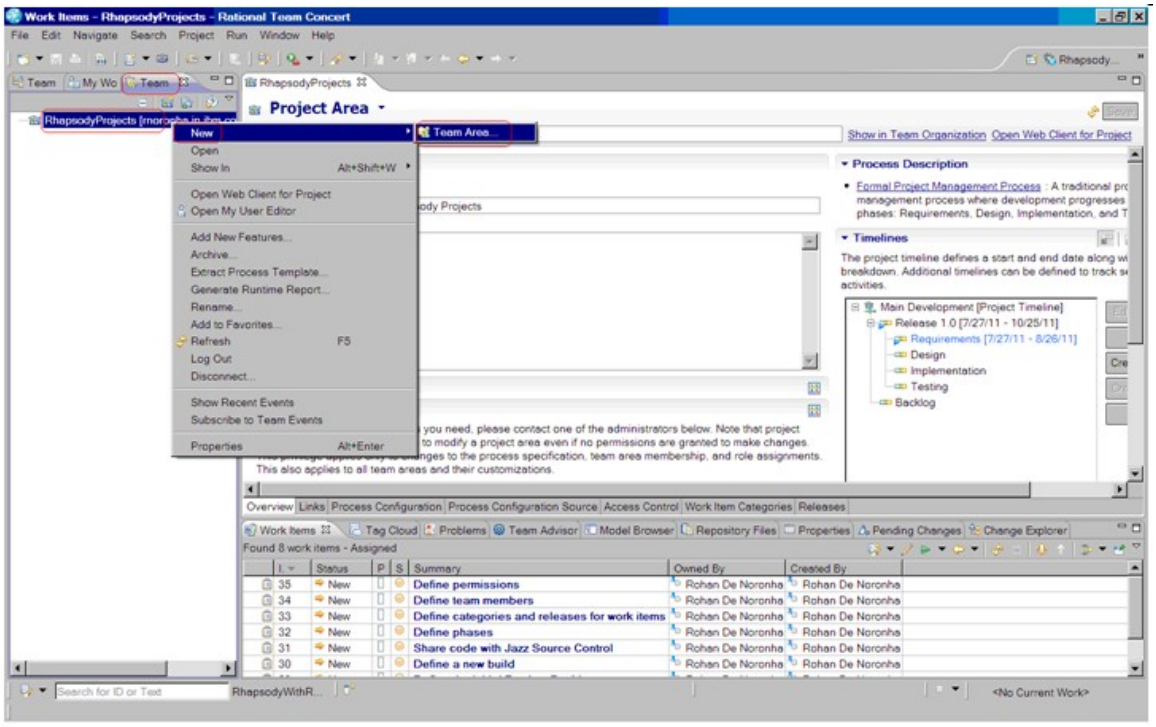


Choose a Process Template

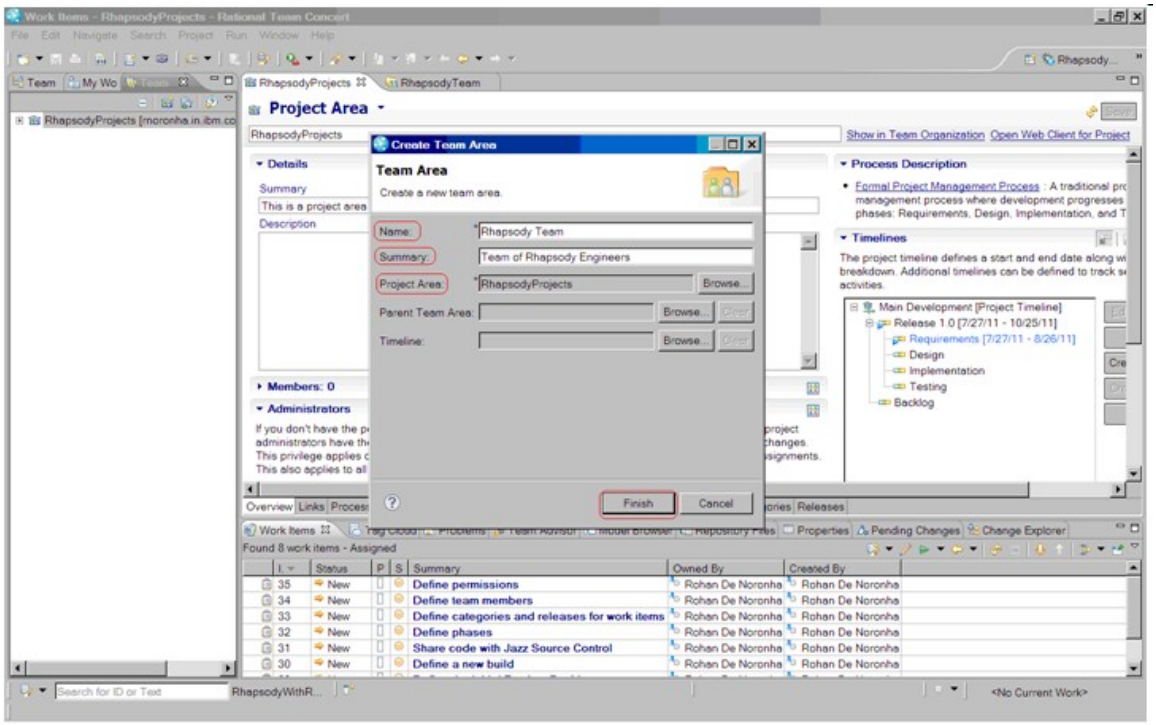
Create a Team area

A team area is created to assign users in particular roles for work on a timeline or a particular set of deliverables.

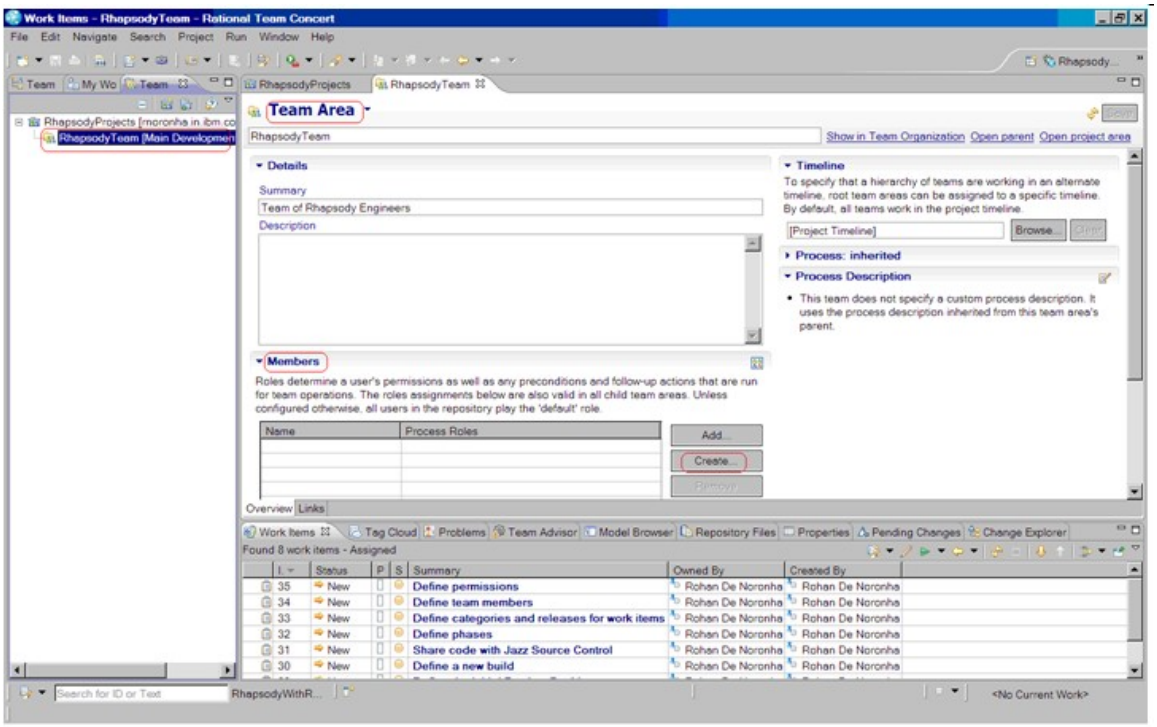
1. In the Team Artifacts view or the Team Organization view, right click a project area or an existing team area. Select **Team Area** in the **New** menu.
2. In the Create Team Area window, type a name and an optional summary for the team area. Click Browse to modify these values:
    a. Select a project area in the repository.
    b. Select a team area as a parent for the new team. Alternative, click **Clear** to move the team area to the top-level in the hierarchy.
    c. Assign the team to a timeline defined by the project area.
    d. Click **Finish**. The team area opens in the editor view.
3. Click **Save**.



Create a Team Area



Enter Team Area information

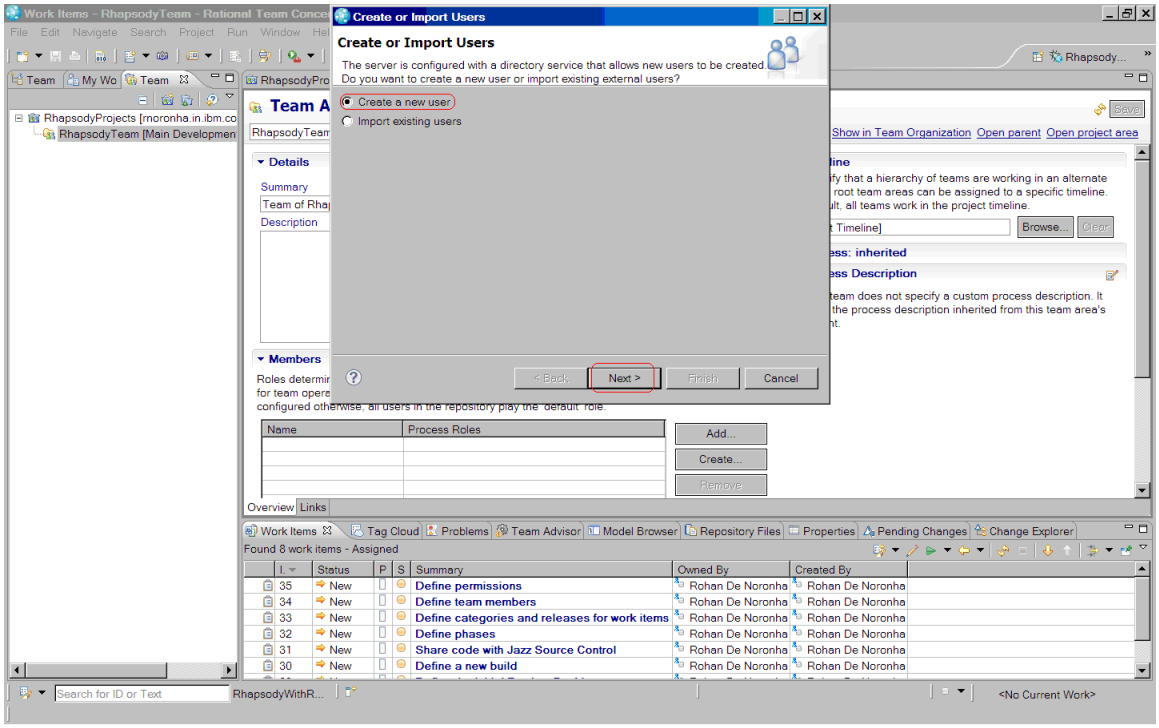


Team Area view

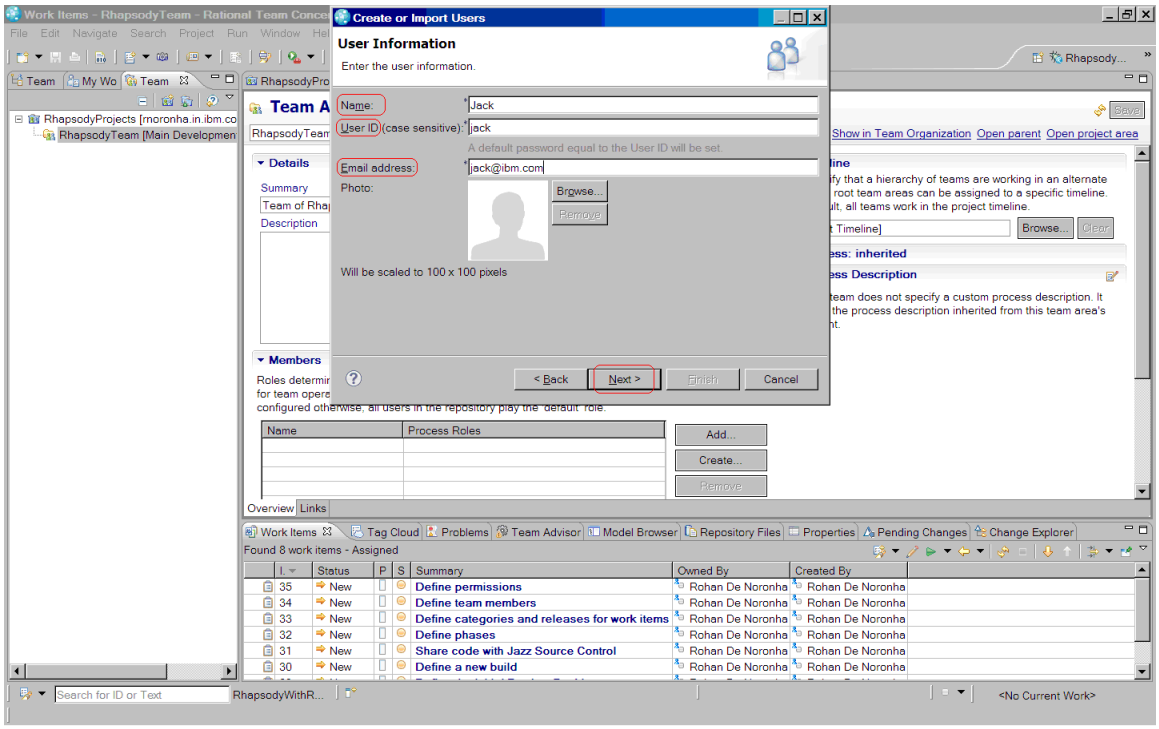
Create and define users and groups

You must create users, define repository groups, assign licenses, and add users to the team. Users are created and added as members in a project area or a team area. When creating a user, you assign a name, user ID, email address, repository groups, and client access licenses.

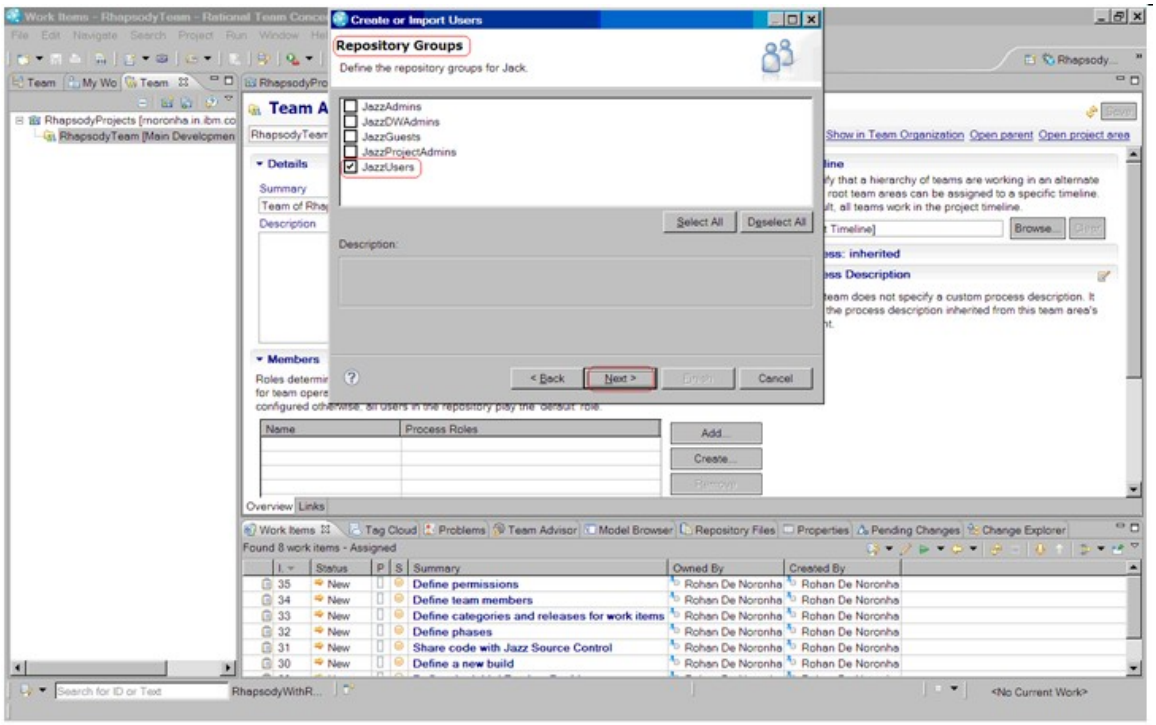
1. Open the Overview page in the project area editor or the team area editor:
   - For a project area, right click the project area in the Team Artifacts view. Click **Open**.
   - For a team area, expand a project in the Team Organization view. Right click a team area. Click **Open**.
2. In the Members list, click **Create**.
3. In the Create or Import Users window, click **Create a new user**. Click **Next**.
4. Type a name, user ID, and email address for the new user. Optionally, navigate to a photograph of the user and add it. Click **Next**.
5. Assign one or more of these repository groups to the user:
   - JazzAdmins: Administrators of a Jazz repository with full read-write access.
   - JazzDWAdmins: Administrators of a Jazz repository with specific permissions to control the data warehouse on a Jazz Server.
   - JazzGuests: Users with read-only access to the Jazz repository.
   - JazzProjectAdmins: Administrators of a Jazz Repository with specific permissions to create and modify project areas, team areas, and process templates.
   - JazzUsers: Users with regular read-write access to the Jazz repository.
6. Click **Next**. Assign one or more client access licenses.
7. Click **Finish**. Click **Save** in the project area or the team area editor



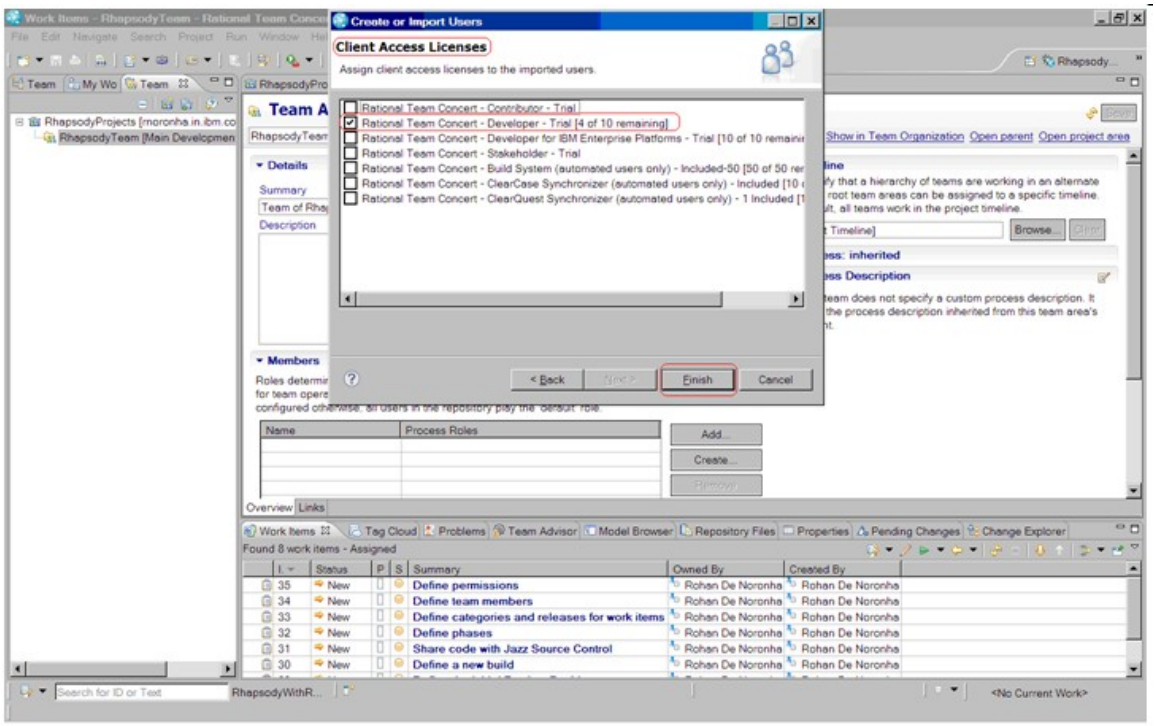
Create Users



Enter user information

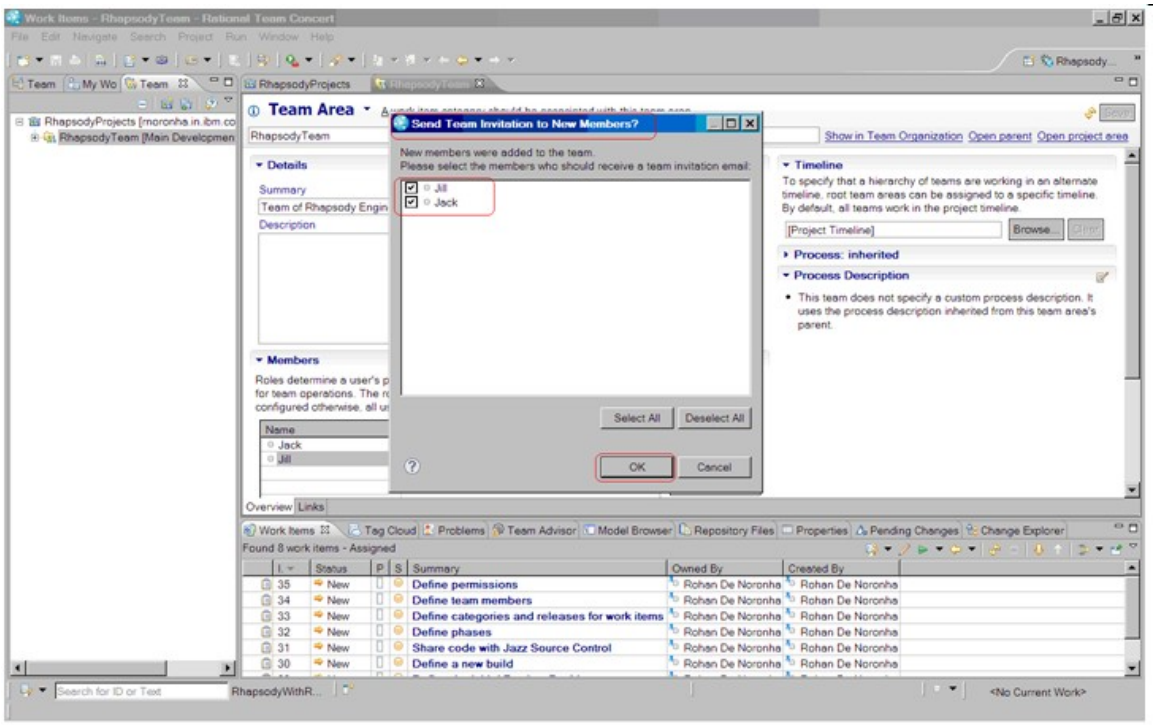


Define Repository Groups for the user



Allocate Client License to user

After adding users to a team, you are prompted to send an invitation to the new members to join the project area or team area. An email invitation is sent and can include the repository name, user ID, project area, and team area. Depending on your team process, after accepting the invitation, new users receive new work items to guide them through common team tasks. These tasks include setting up instant messaging, finding work items, and creating a repository workspace.

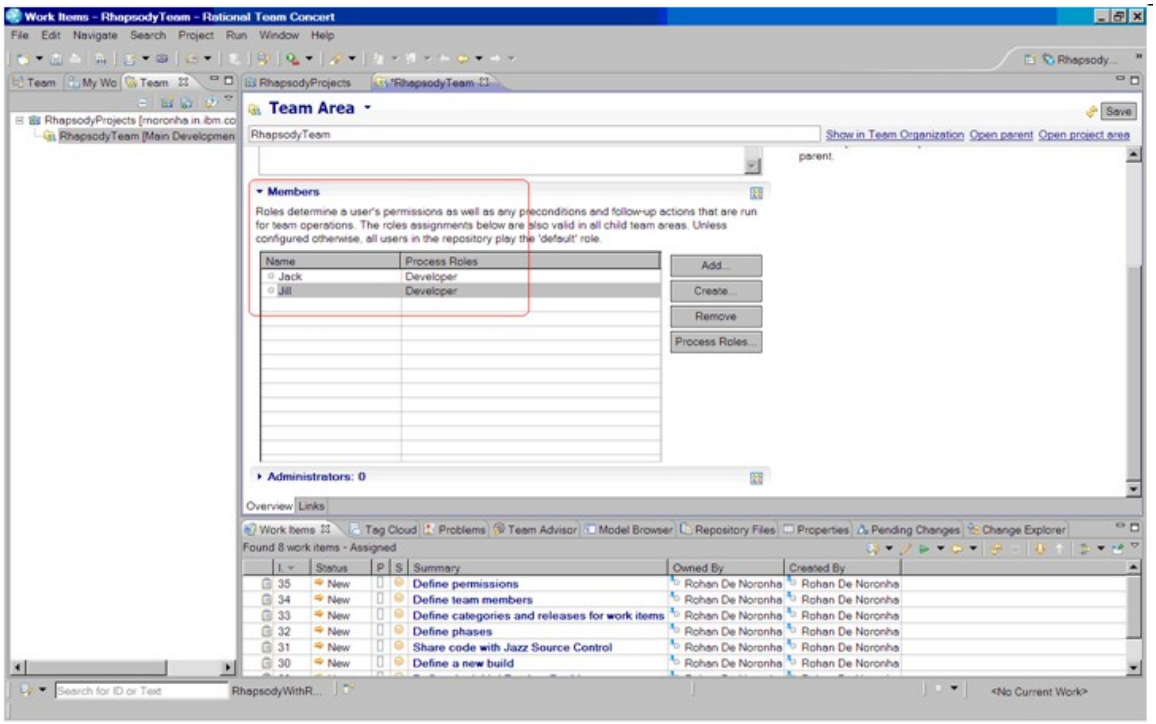


Send Team Invitation to client

Assign roles to users

A user account is assigned one or more roles based on their involvement in the project

1. Open the Overview page in the project area editor or the team area editor:
   - For a project area, right click the project area in the Team Artifacts view. Click **Open**.
   - For a team area, expand a project in the Team Organization view. Right click a team area. Click **Open**.
2. Select the user name in the Members list and click Process Roles.
3. In the Edit Process Roles window, select a role in the Available Roles list. Review the role description. Click **Add** to assign the role to the user name. Click **Finish**.
4. Click **Save** in the project area or the team area editor.



Assign Roles to users

In case you have encountered permission errors during any of the tasks, see the Team Advisor view for detailed error information. Redefine appropriate permissions to users.

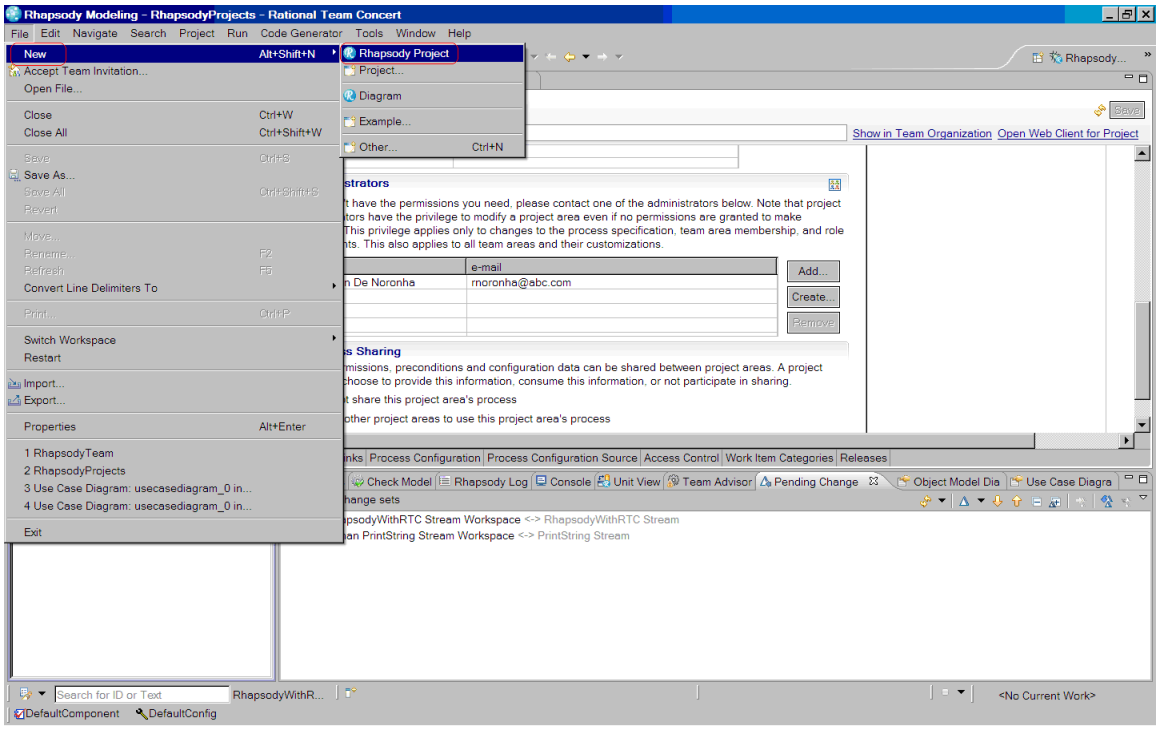
With the project administration complete, each user is now capable of using the Rational Team Concert Eclipse client to create a Jazz repository connection and connect to an existing project area.

Rhapsody Project Administration

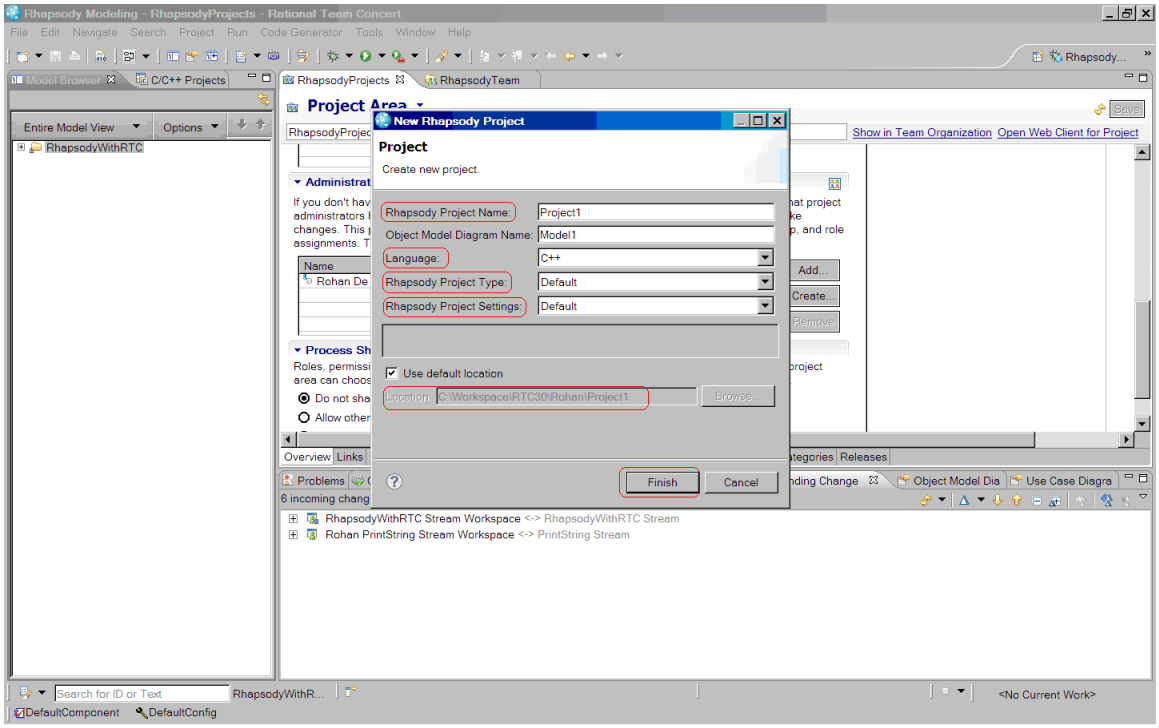
A Project Manager or Administrator will create a Rhapsody project in the local workspace and share the project to the Jazz source control

Create a Rhapsody model to the local workspace

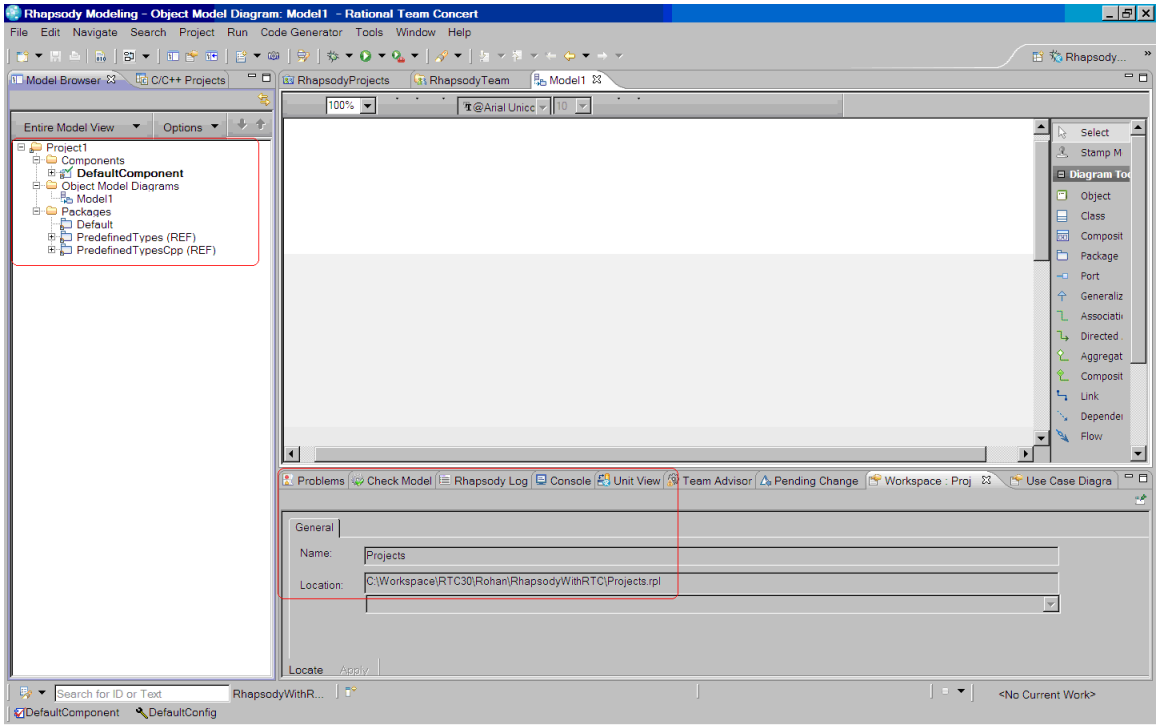
1. Switch to the Rhapsody Modeling perspective in the Team Concert client.
2. From the **File** menu, create a new Rhapsody Project.
3. Enter project information in the Wizard. Click **Finish**.



Create a new project in the local workspace

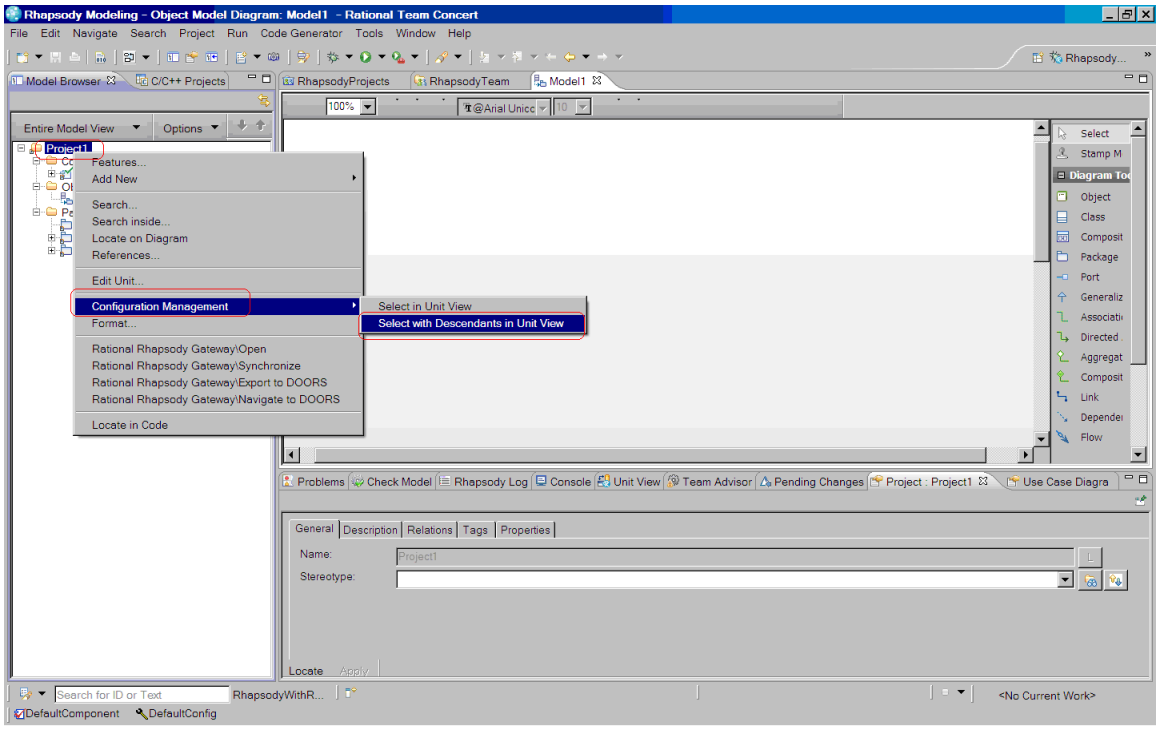


Enter new project information



New Project view

In the Rhapsody perspective change the model to **Unit View**.

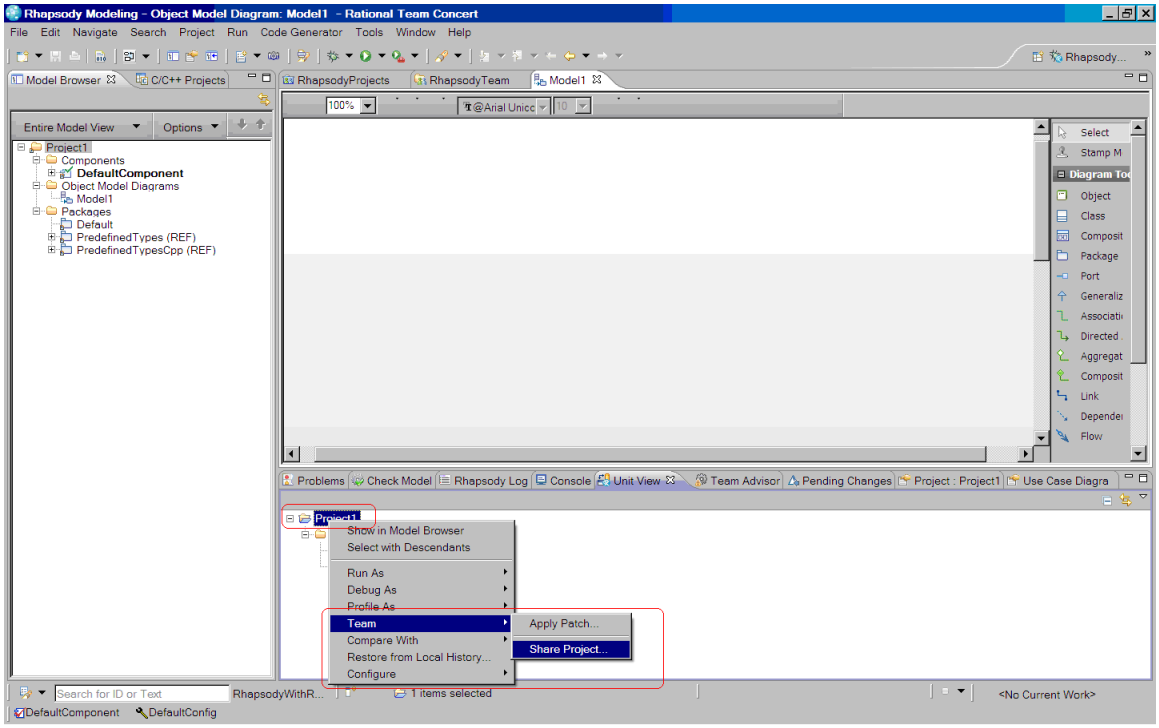


Change model view

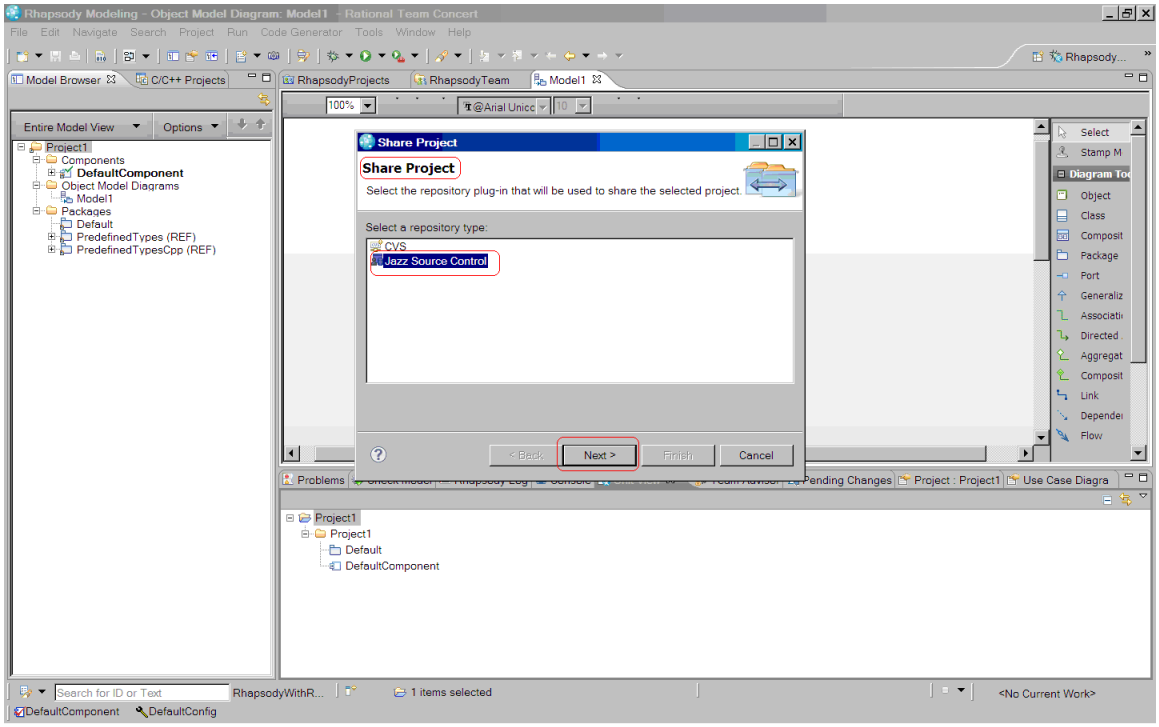
Share the project to the Jazz Source Control repository

To add a project in your Eclipse workspace to the Rational Team Concert source control, use the Share Project wizard to create a repository workspace and add the project to it. This copies the project's files to the repository and links the Eclipse workspace with the repository workspace so that you can back up your work by checking in changes that you make

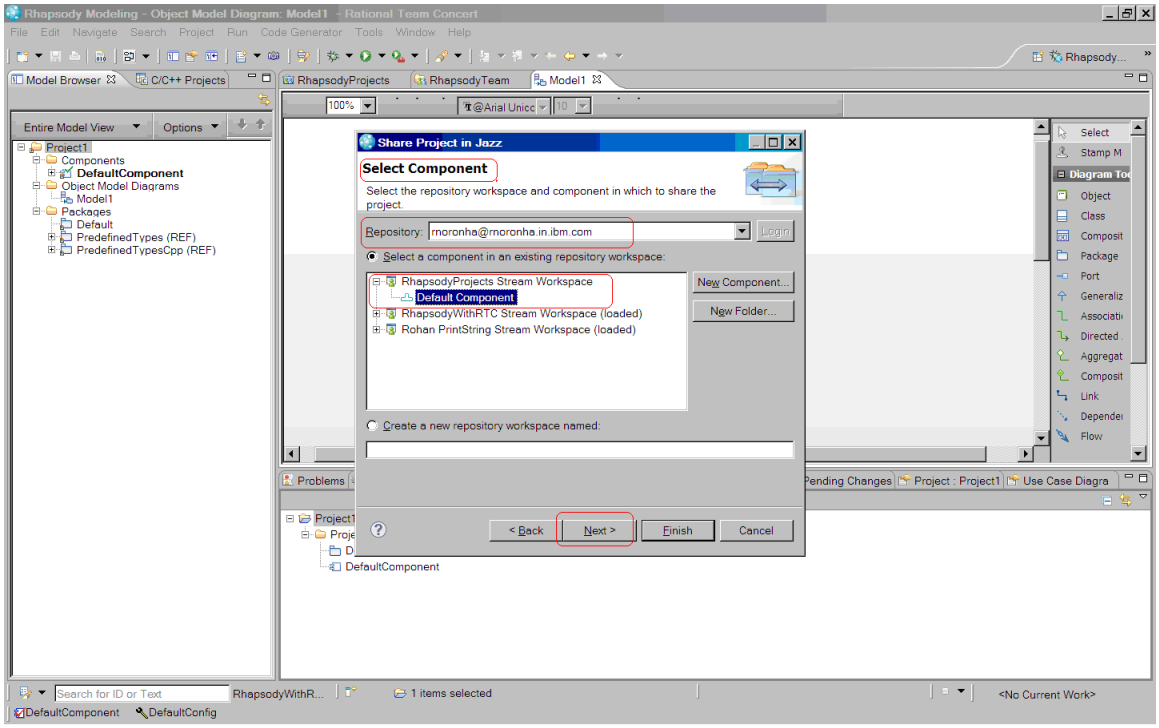
1. Right click the project. Select **Share Project** in the **Team** menu.
2. In the Share Project wizard, select **Jazz Source Control** as the repository type. Click **Next**.
3. Specify the repository workspace to use.
    - If you have an existing repository workspace and component in which you want to share the project, leave **Select a component** in an existing repository workspace selected. Expand the workspace and select a component.
    - If you want to place the project in a new component in an existing repository workspace, select the workspace. Click **New Component**. Enter a name for the new component. Click **OK**. Select the new component in the list. Click **Next**.
4. Select the projects to share in the Jazz Source Control.
5. Select Resources to exclude from Rational Team Concert Source Control.
6. Right click the Outgoing projects in the Pending Changes view and **Deliver**.



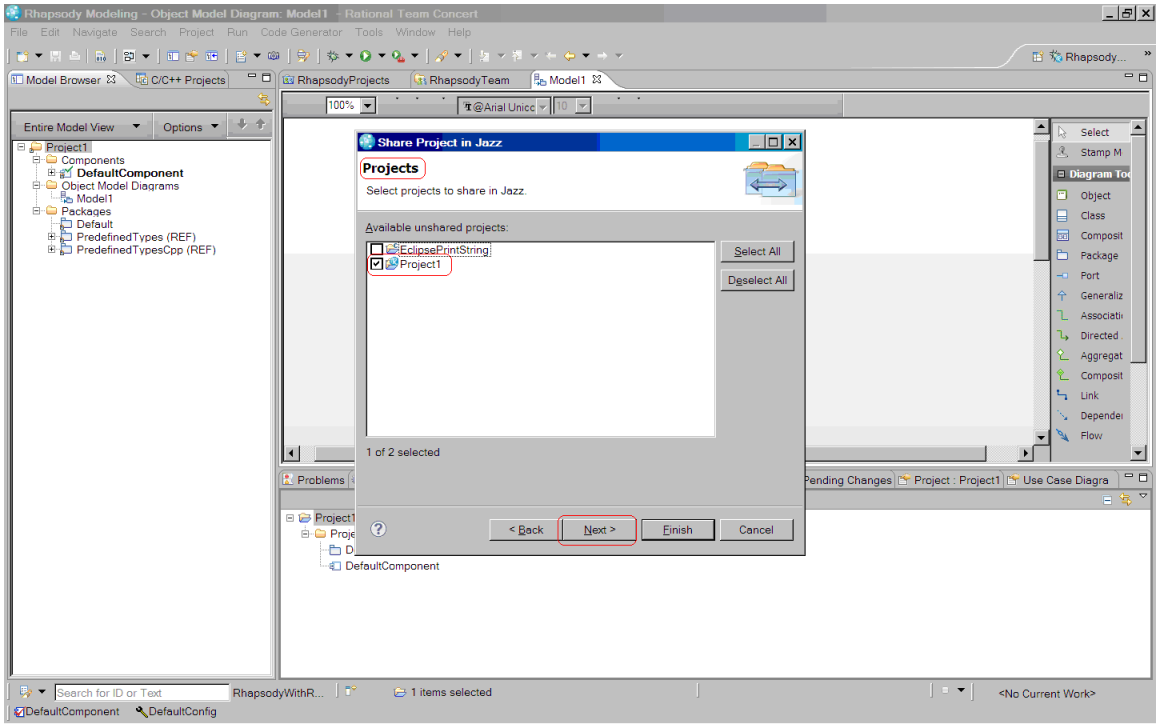
Share project to Jazz source control



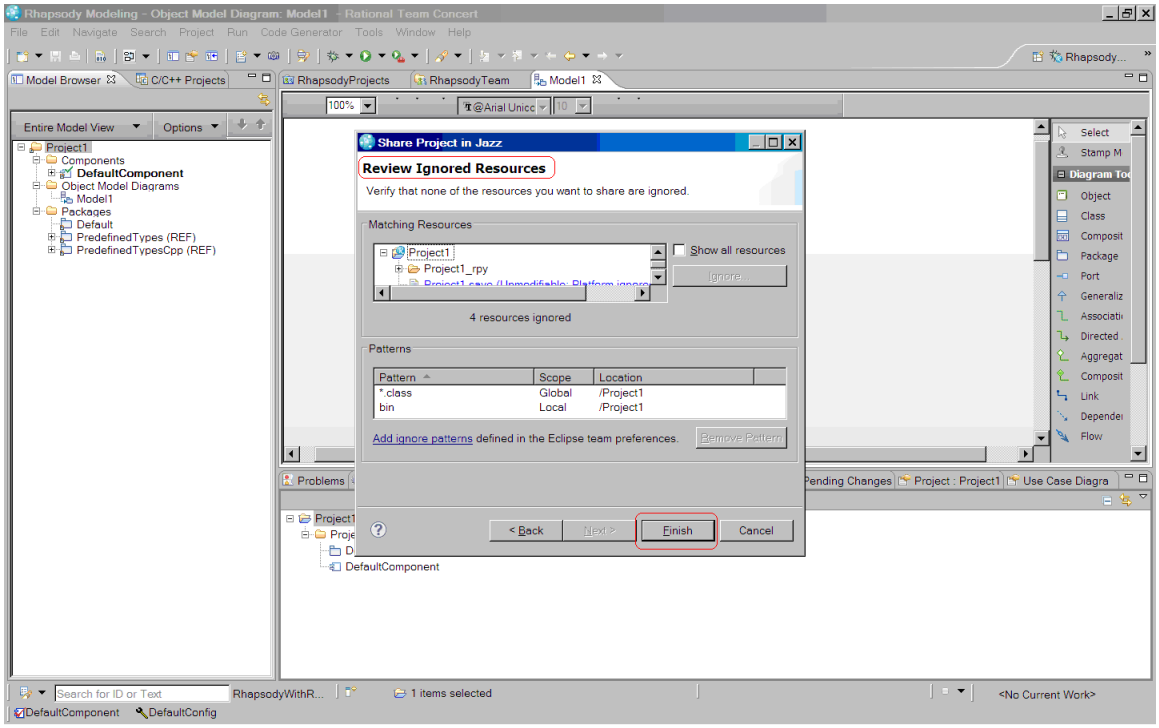
Choose the Repository



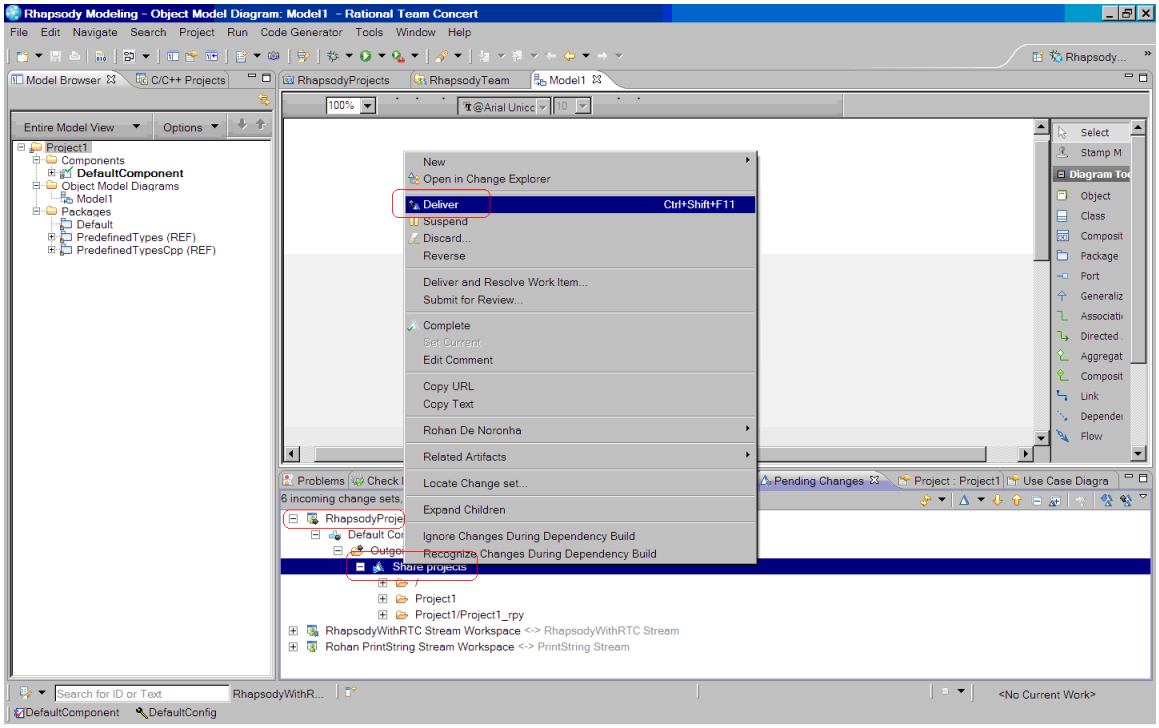
Select the component to load the project



Choosing the projects to share



Ignore resources not to be shared



Deliver the project to the main stream

With this, the task of sharing a Rhapsody project to the Rational Team Concert Source Control is accomplished.

Collaboration

Use the Source control component in Rational Team Concert to manage the Rhapsody models and model artifacts that are placed under version control and shared with a team.

Rational Team Concert places model artifacts in the Jazz repository. You create a repository workspace to hold your private copies of the project files with which you will work. You then load the contents of your repository workspace into a sandbox (a directory in your workstation file system) so that the project files are accessible to tools like editors, compilers, and integrated development environments.

As you make changes to the contents of your sandbox, you periodically check them in, which copy them to the repository workspace so that the two workspaces contain the same versions of the files. In the repository workspace, related changes are collected as change sets, which enable changes in multiple project files to be committed in a single operation.

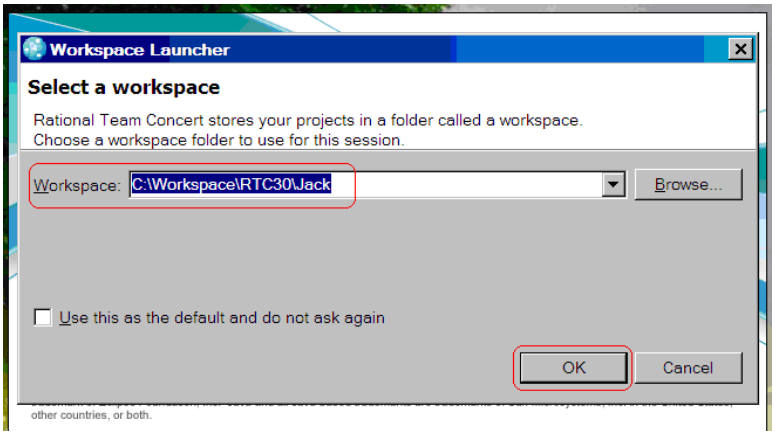
A software development team typically works with a large base of files that comprise the source code for a software product or system. As a team member, you can add new features or fix existing ones within this base of source code. After you build and test the code to verify that your changes are correct, you can share the changes with the rest of the team.

Fetching private copies of the project from the Jazz Repository

You will create a local Rational Team Concert eclipse workspace, create a repository connection, and connect to the project area. You will create a repository workspace to flow with the project main stream, load the project artifacts into the sandbox, and create Rhapsody projects in the rhapsody modeling perspective.

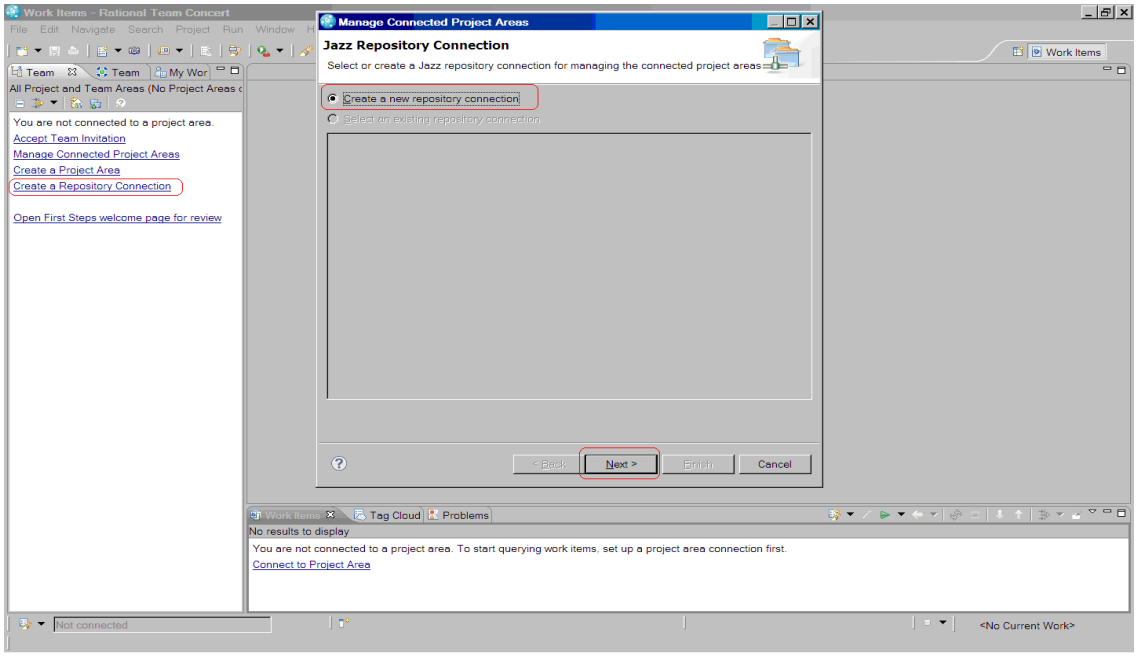
Create a local workspace

In Rational Team Concert, create a workspace for project information and artifacts.

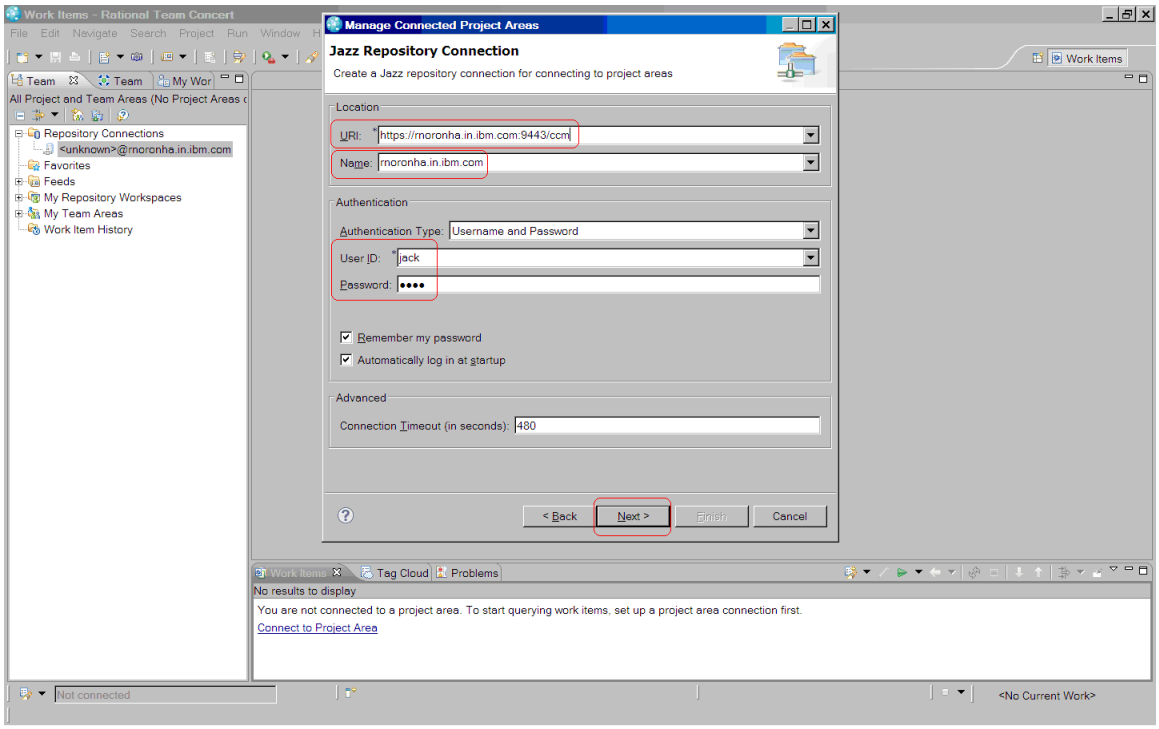


Connect to a the Project Area

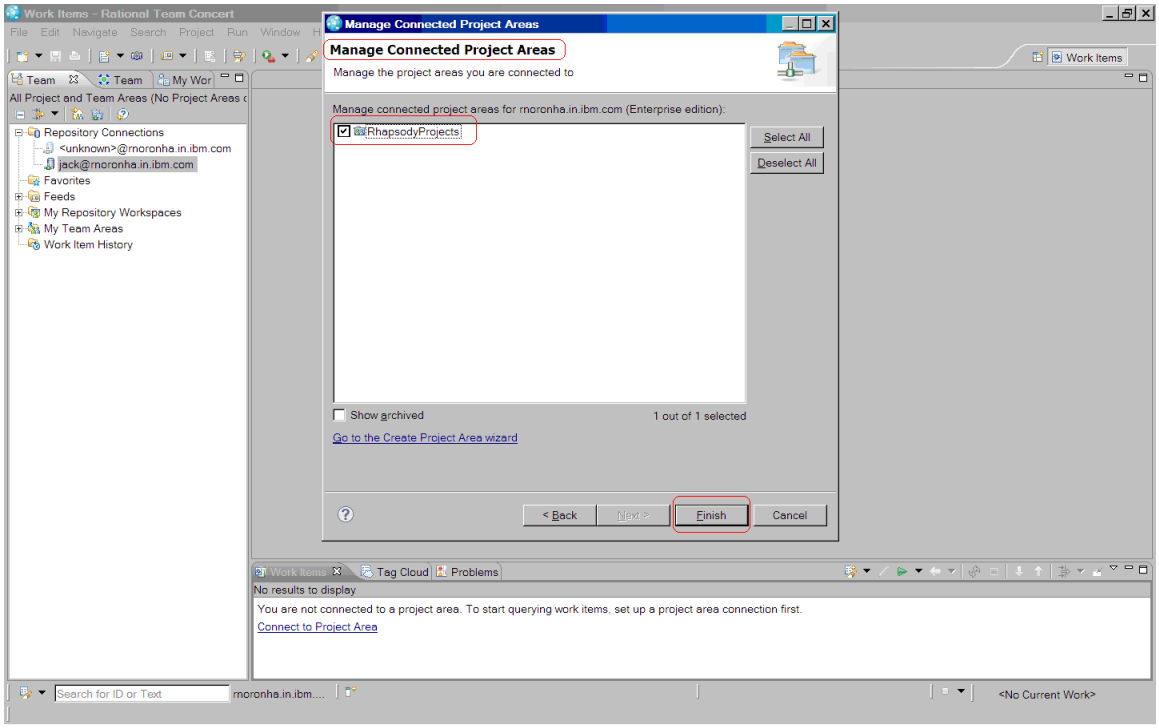
Click the hyperlink **Create a Repository Connection** or **Manage Connected Project Areas** to create a repository connection.



Enter the **Name** and **URI** of the repository connection. Also provide login details.

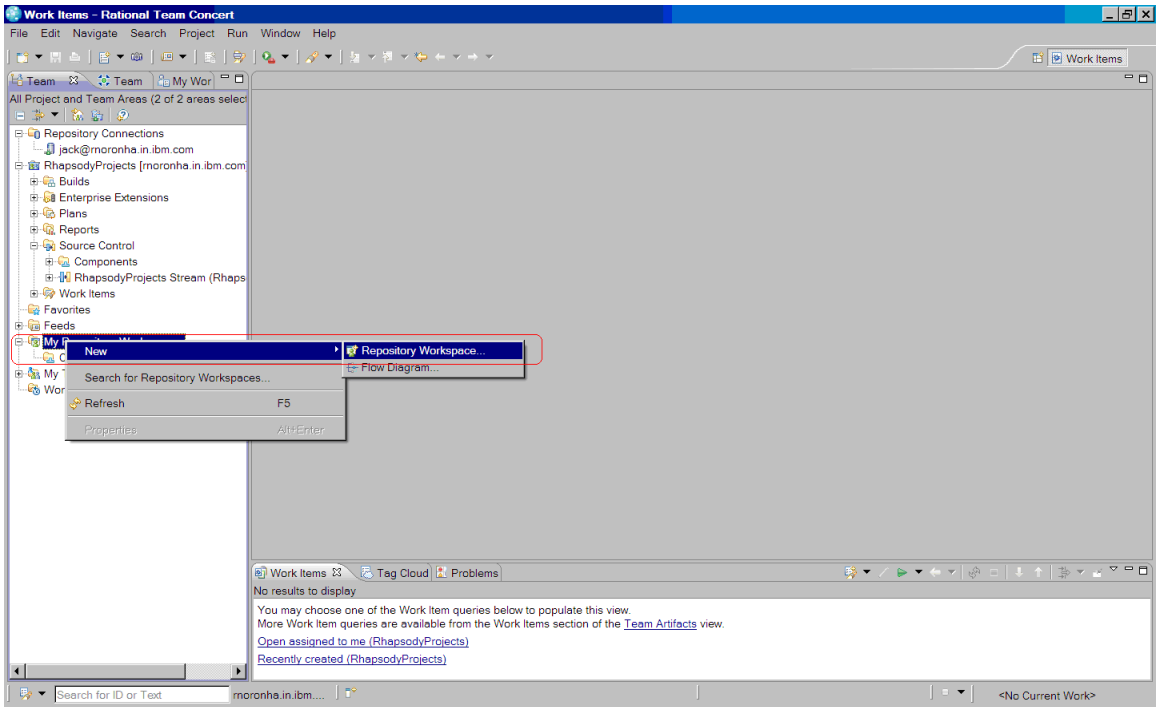


Choose the Project Area to which you will connect. Click **Finish**.

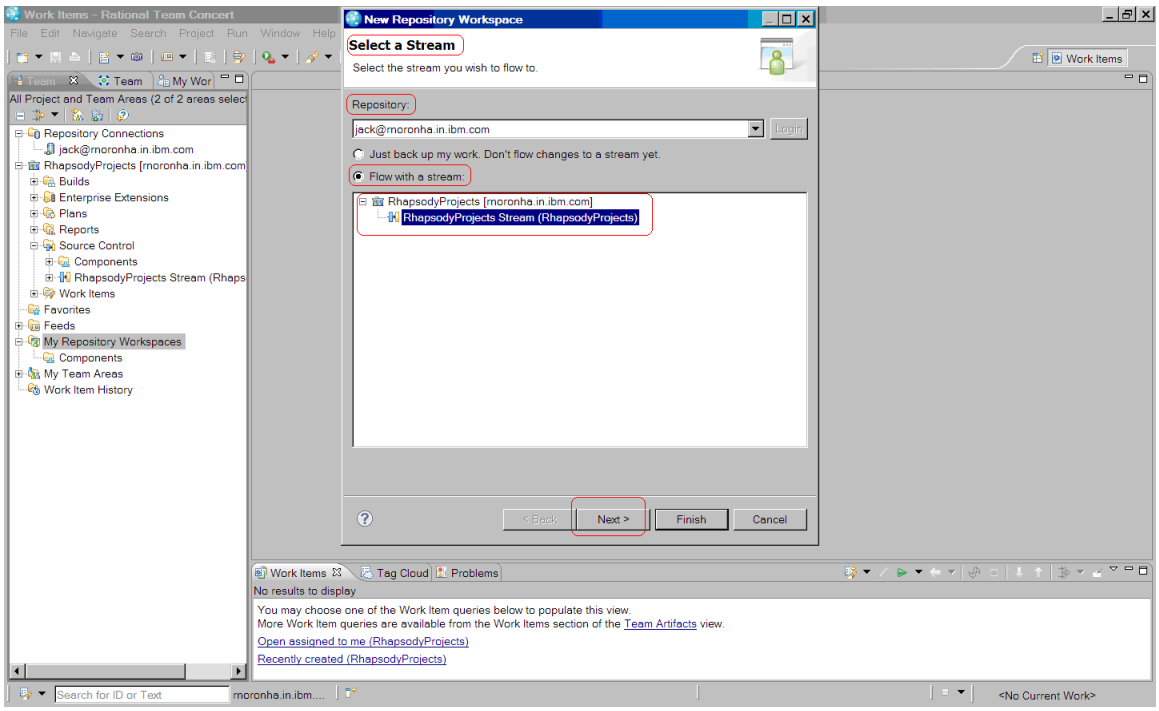


Create a synchronized Local Repository Workspace

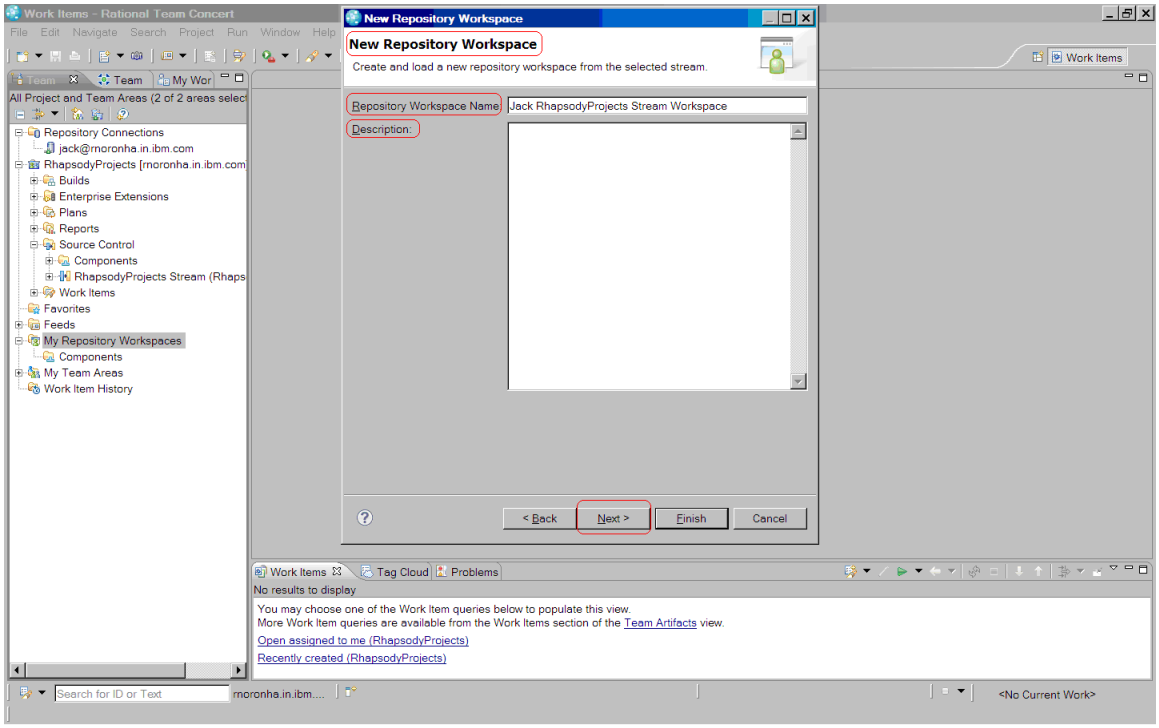
Create a synchronized Local Repository Workspace and synchronize to the Project Main stream. From the pop-up menu of **My Repository Workspaces**, select **Repository Workspace** in the **New** menu to create a local repository workspace.



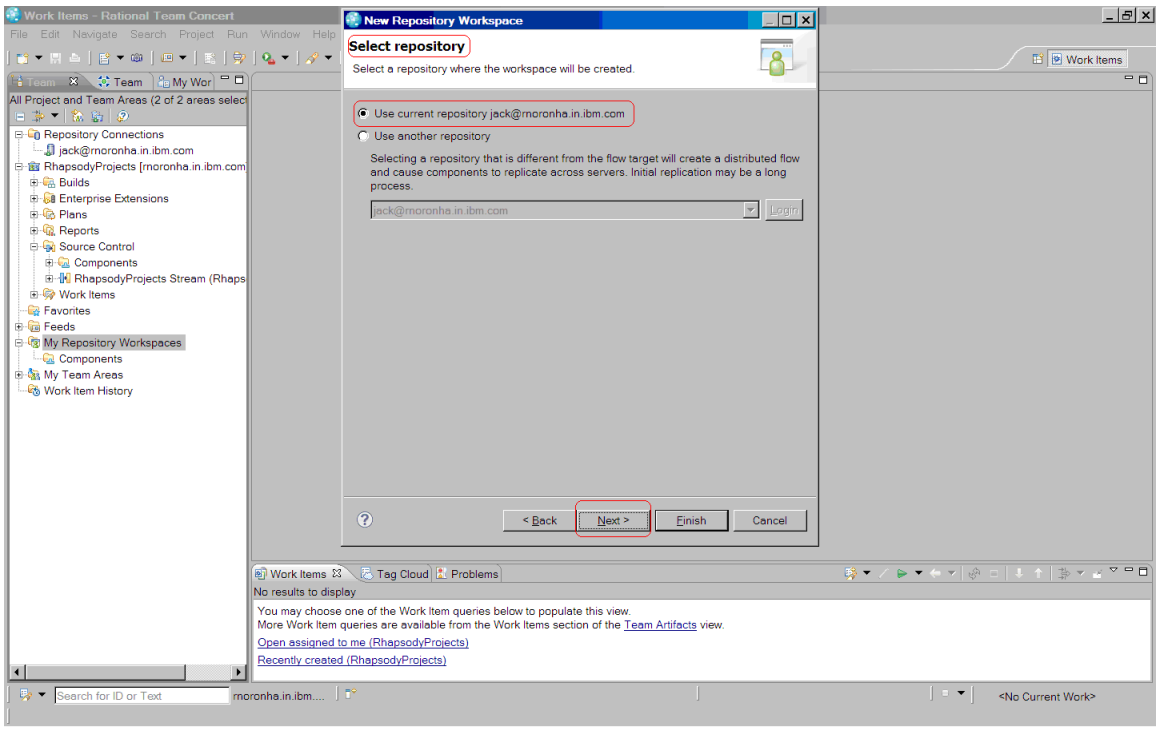
Select a project area stream to flow with the local repository workspace.



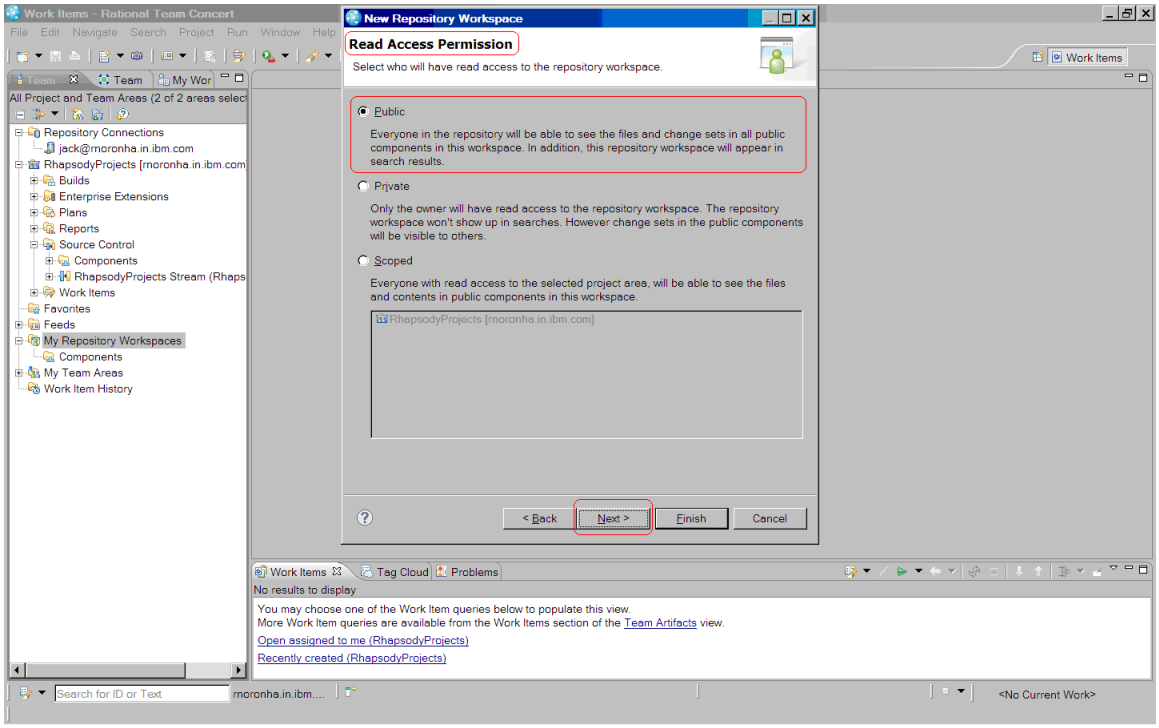
Specify a name and description for the local workspace.



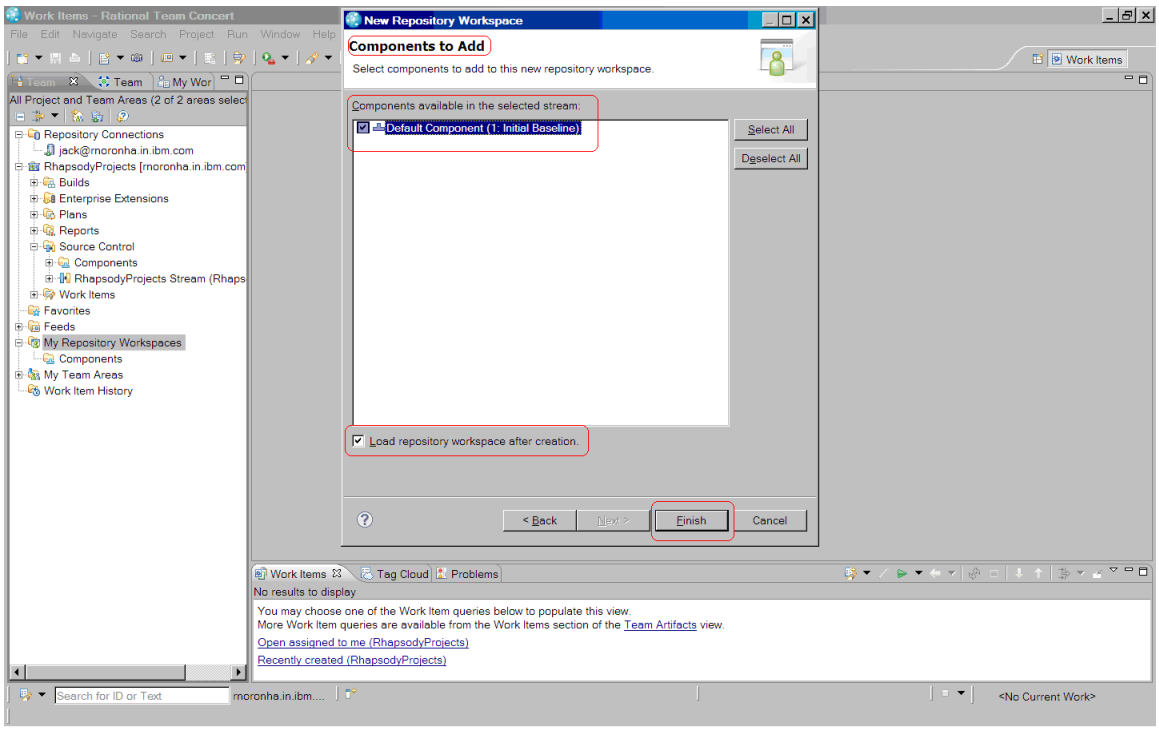
Select the repository name to link the local workspace.



Provide read access permissions to the workspace.

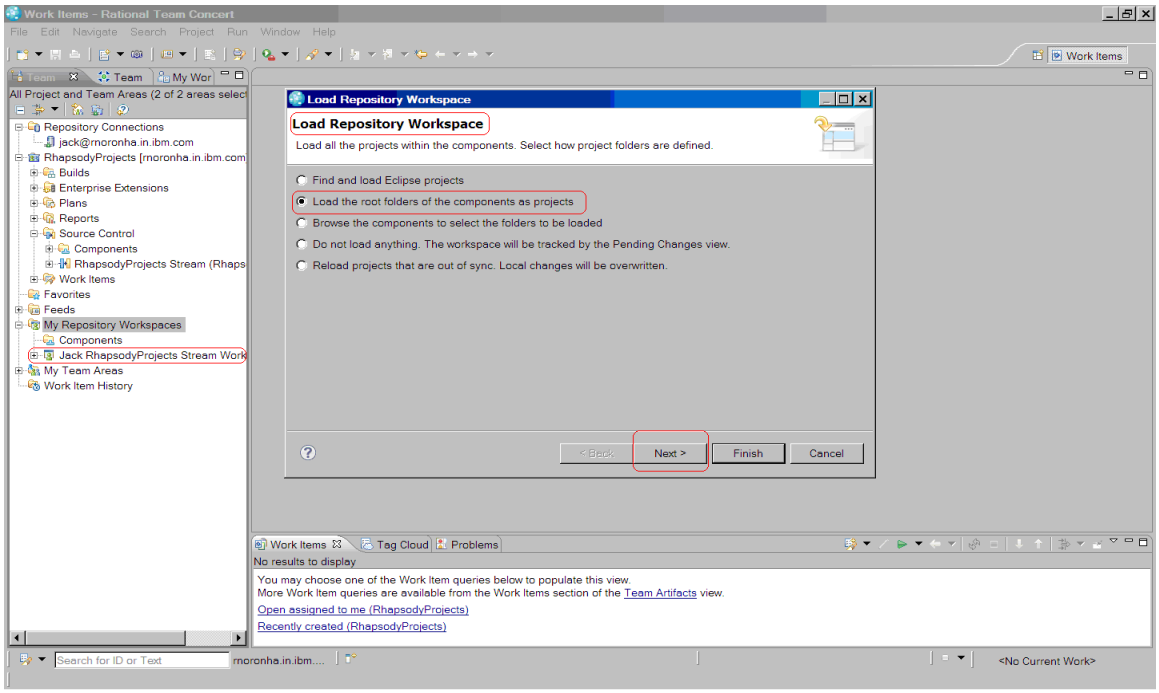


If required, create a component. Otherwise, choose the available component. Click **Finish**. Select the option **Load repository workspace after creation**.

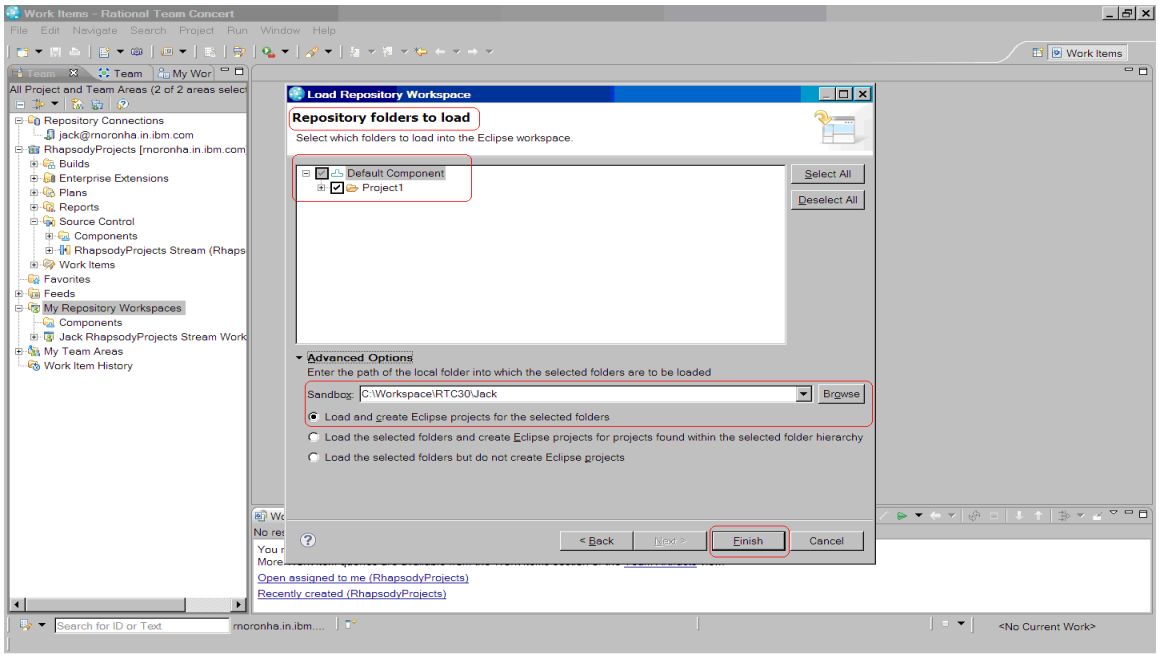


Load the Rhapsody Project to the local workspace

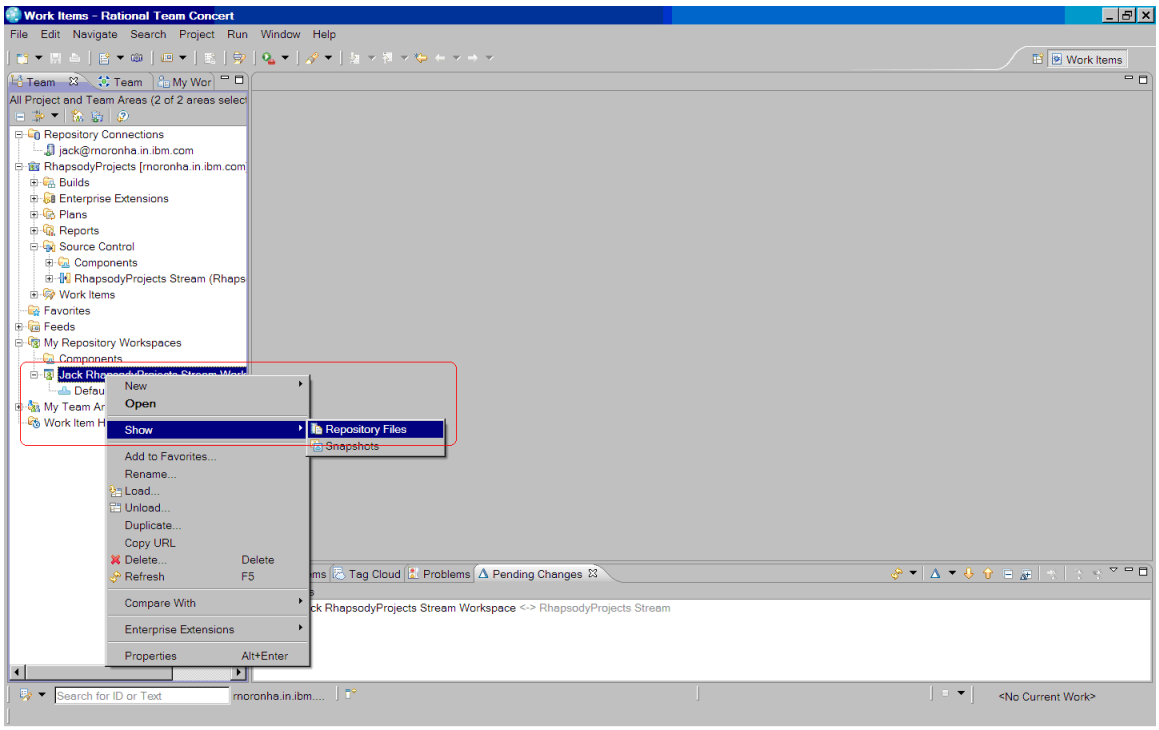
From the pop-up menu of the local repository workspace select **Load Repository Workspace**. Choose the option **Load the root folders of the components as projects**, which contain the project artifacts or resources.



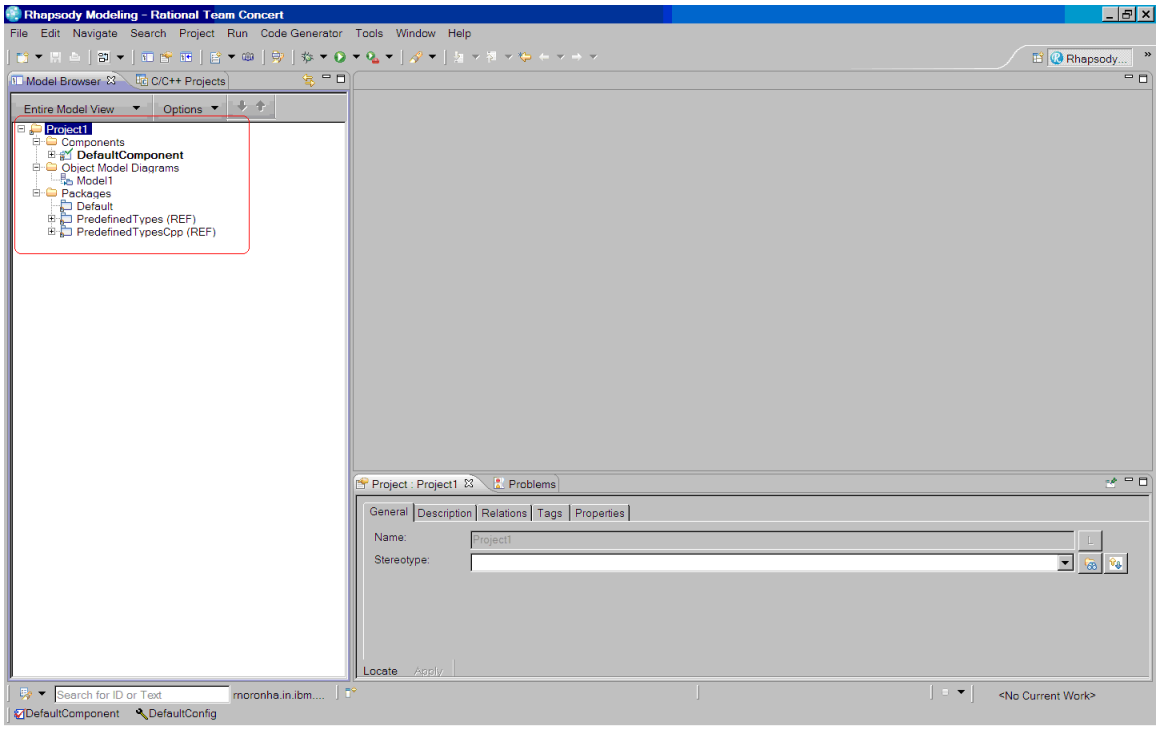
Select the root folder of the project. Provide the sandbox location for the copied the project. If required, create an eclipse project to represent the project. Click **Finish**.



After loading, you can verify the files in the local repository workspace and switch to the Rhapsody modeling perspective to browse the created project.



View repository files in the Rhapsody modeling perspective.



Exploring the source control operational workflow

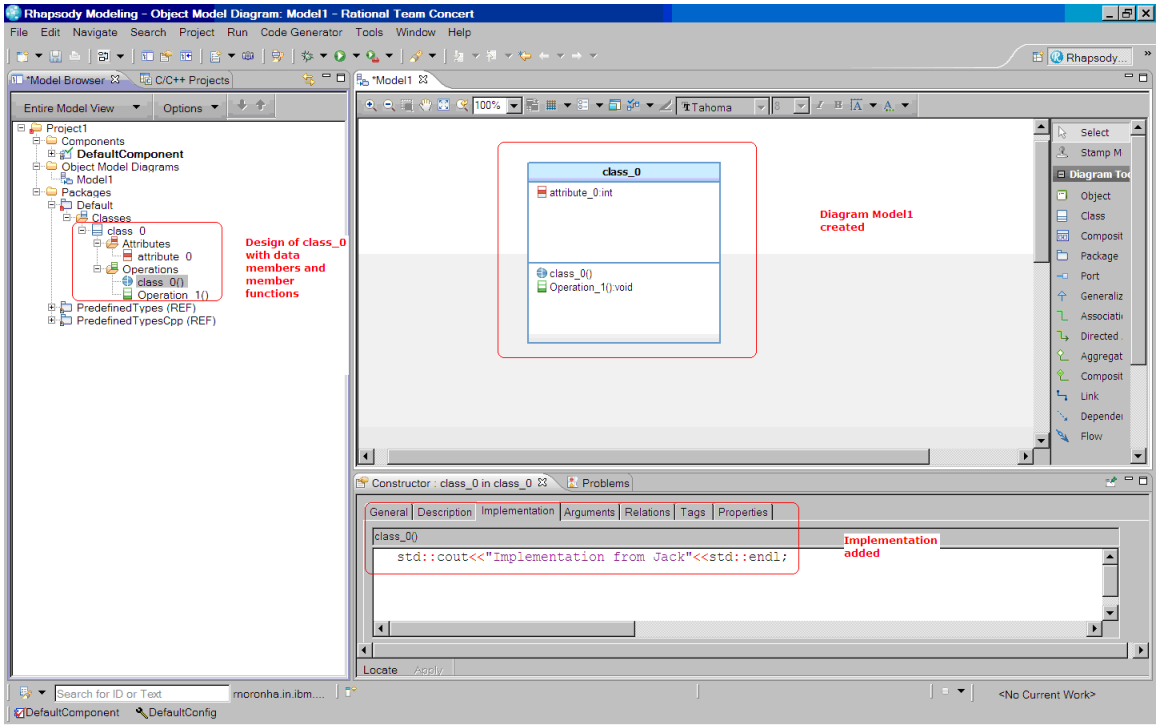
This topic covers the source control features of Rational Concert used by Rhapsody developers to collaborate on a project.

In this scenario, Jack and Jill are two developers working on the single project version controlled by the Jazz source control component. The project artifacts are loaded into their local workspace and Eclipse projects created for the same. Working with the Rhapsody Modeling perspective in the Team Concert Eclipse client, they collaborate with each other while contributing to the development of the project.

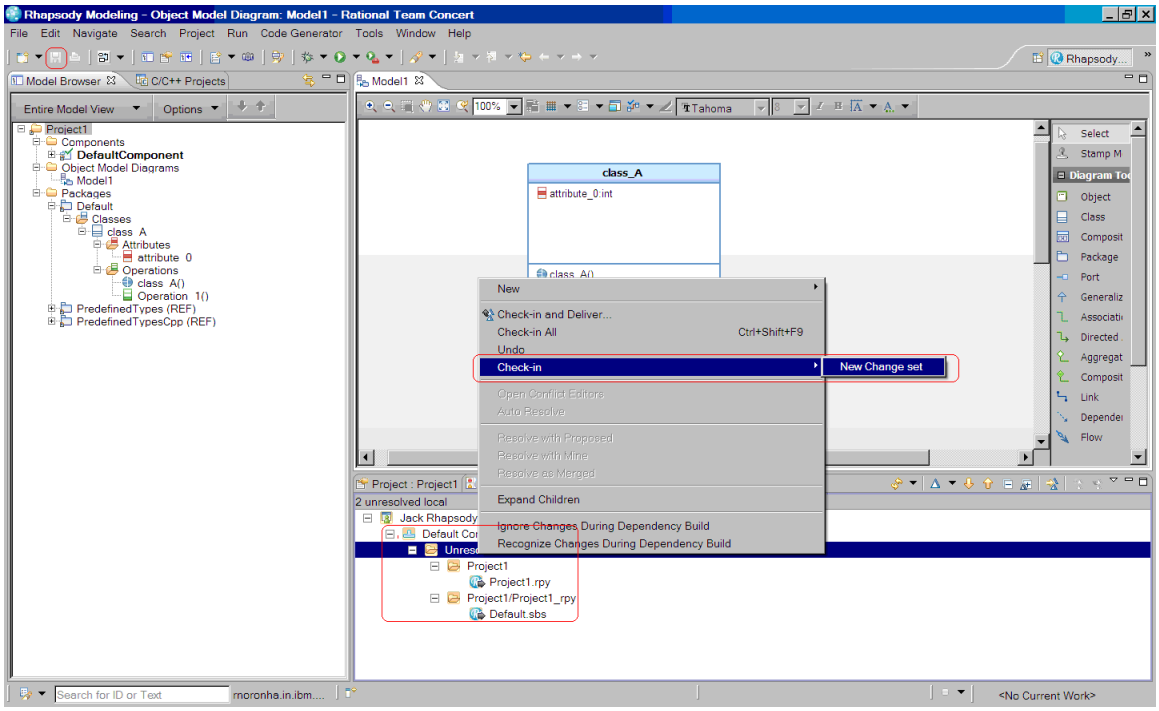
Making a change and delivering to the project main stream

Jack makes a change to the project by designing a class artifact with data members and member functions. A change set is created to represent the changes and delivered to the project main stream. This change set is available to all users synchronized to the project main stream.

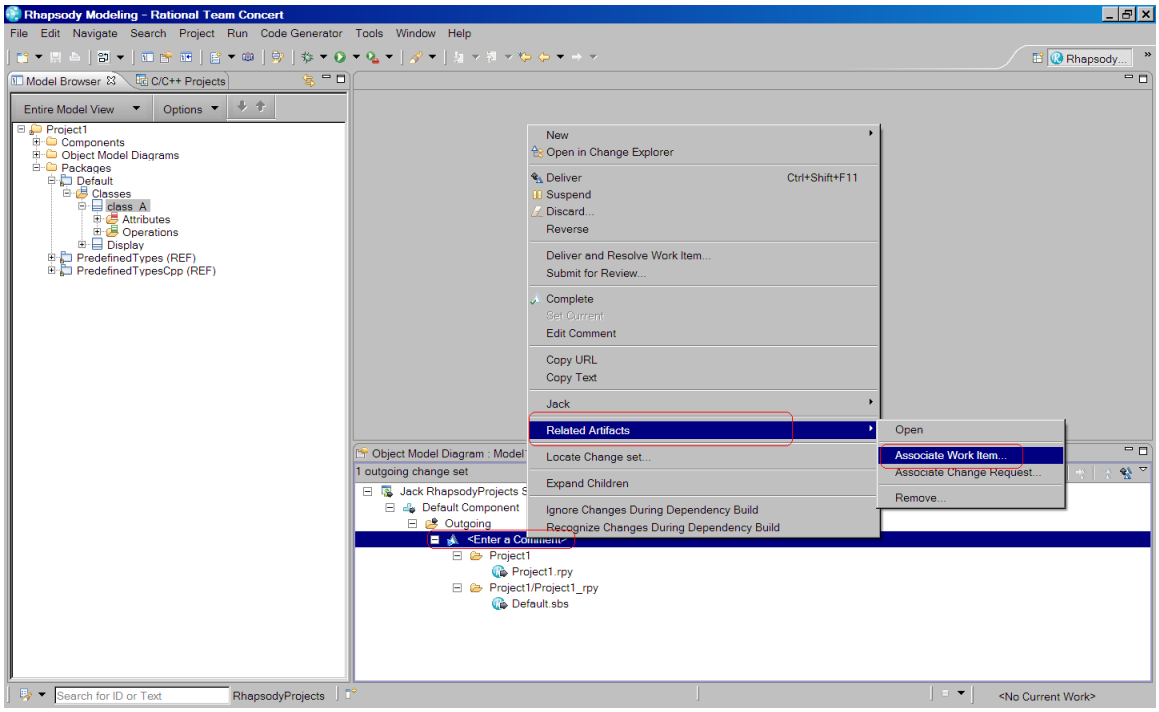
Make changes to the project in the model using the diagram or browser.



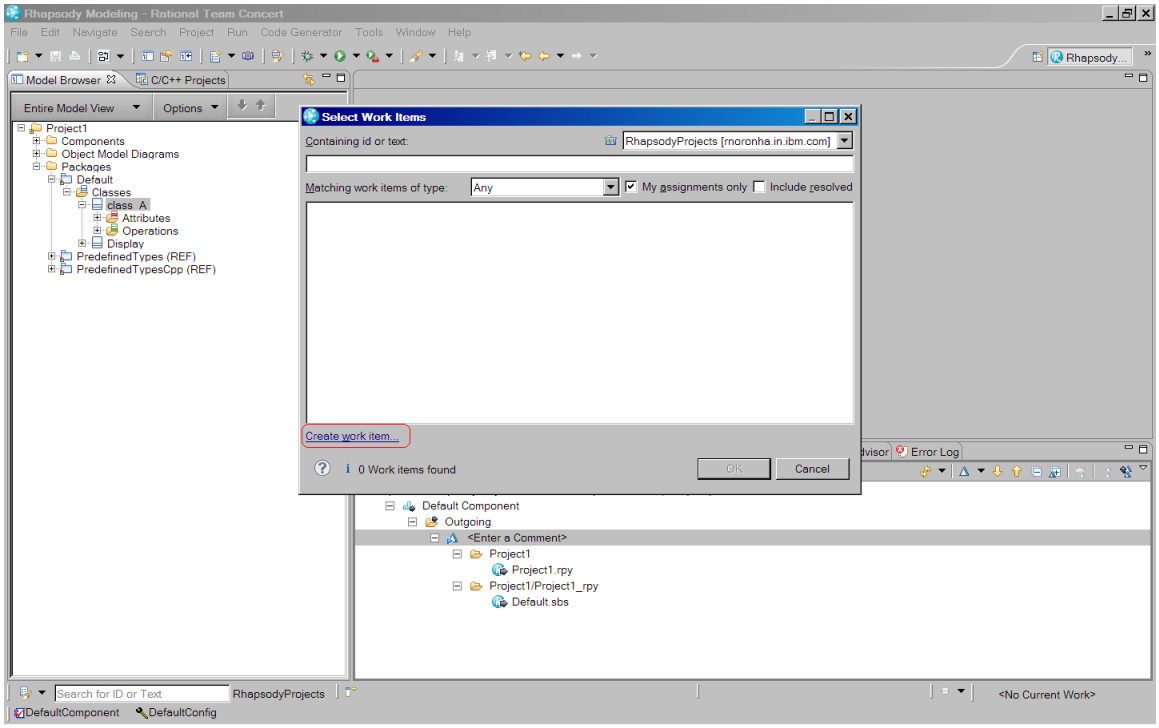
Save the model. View Pending changes to see the unresolved node of changes. Check in the change set. From the pop-up menu of the unresolved node, select **New Change set** in the **Check-in** menu.



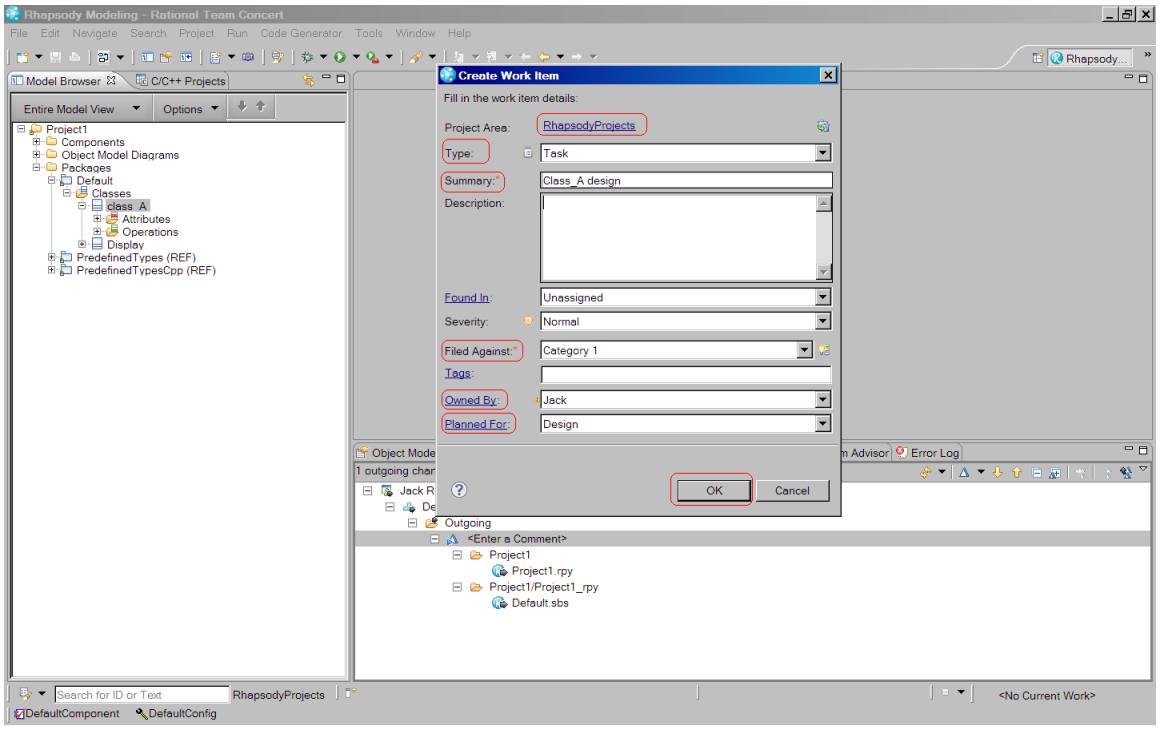
Associate a work item to the change set. From the pop-up menu of the change set, select **Associate Work-item** in the **Related Artifacts** menu. Work items are a source of tracking and managing developments in a project.



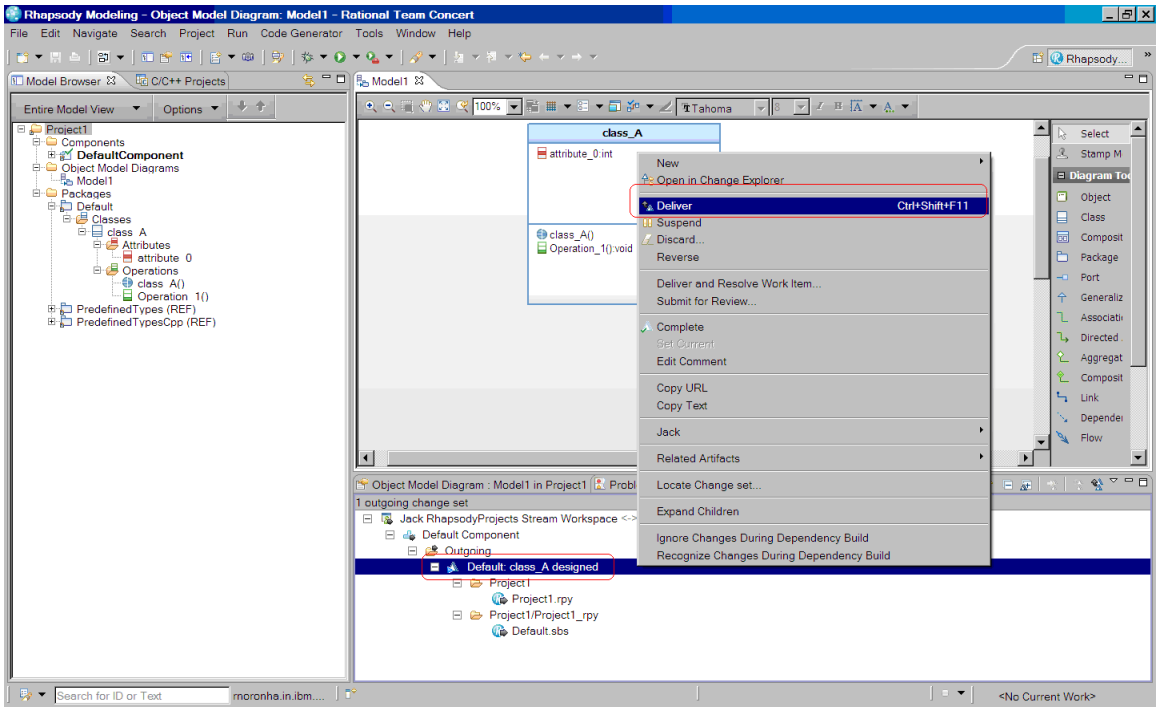
Create a work item or select a related, existing work item.



Enter work item information. Click **OK**.

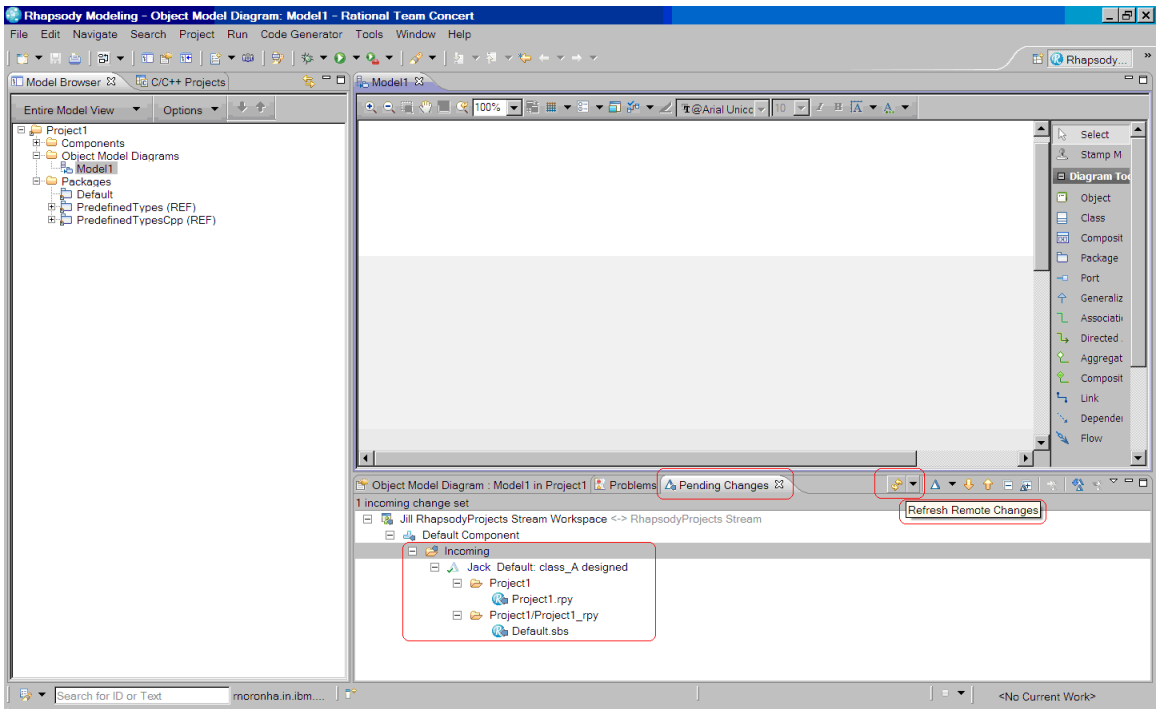


Deliver the change set to the project main stream. From the pop-up menu of the change set, select **Deliver**.

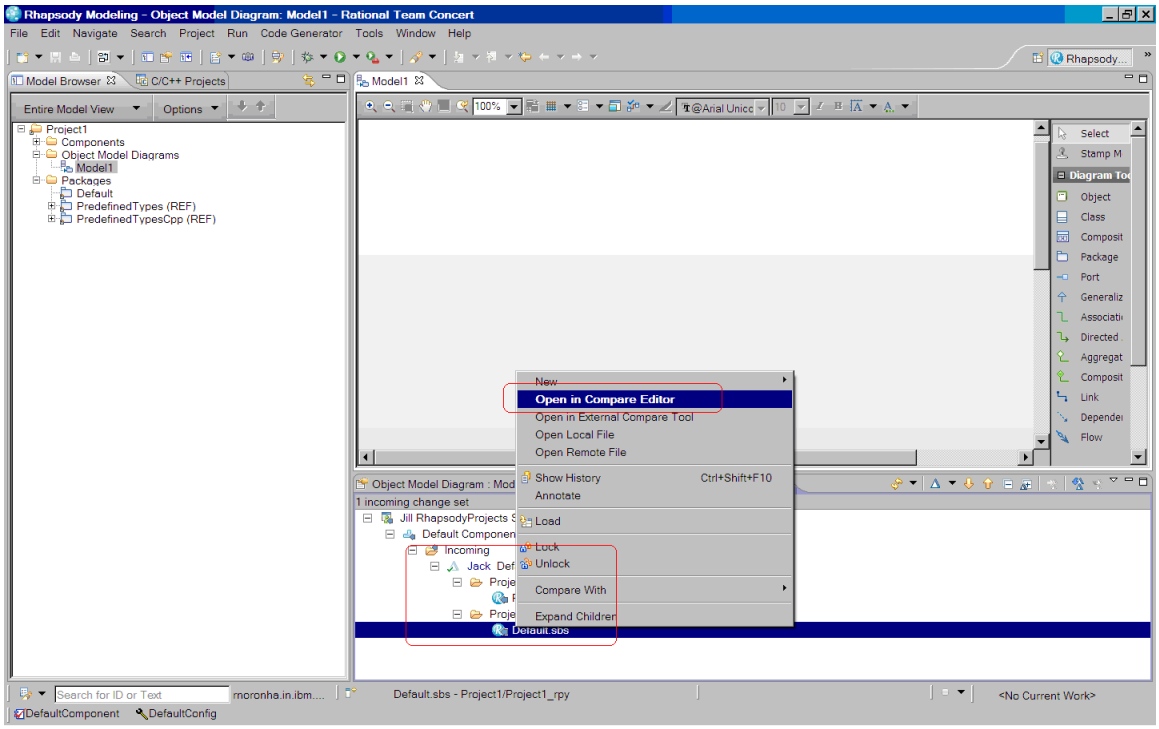


Viewing and accepting changes from the Project main stream

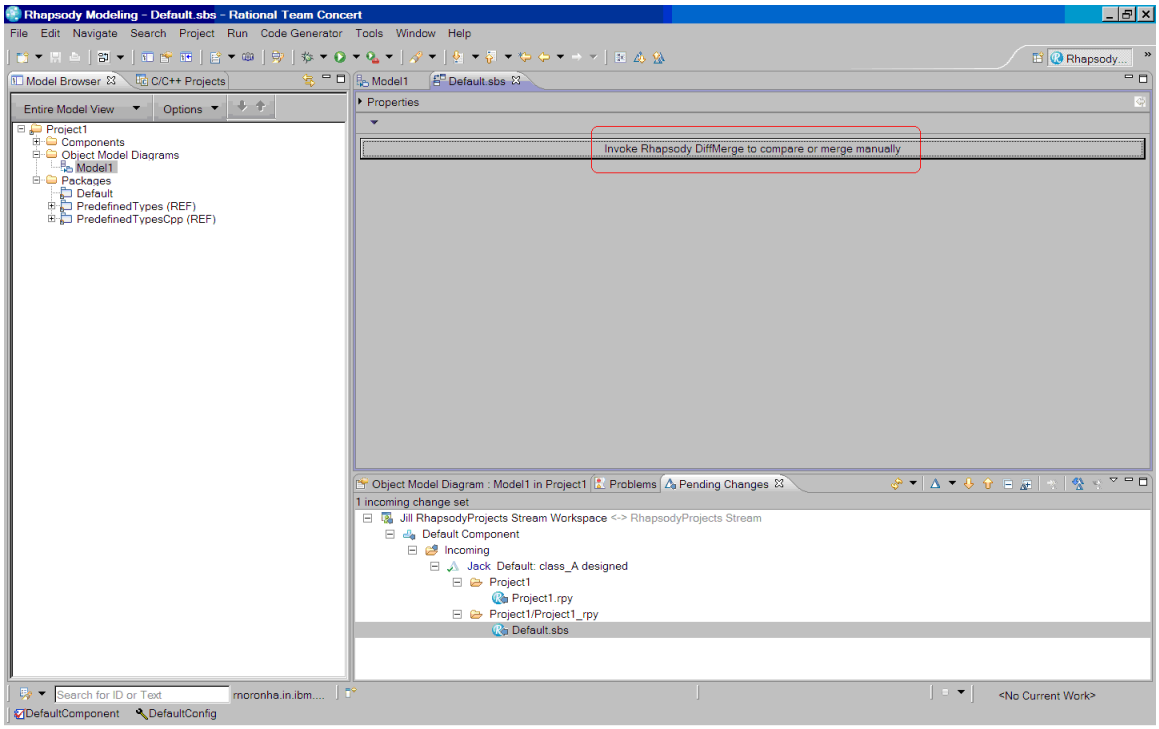
Jill refreshes the workspace to check incoming change sets using the Refresh button on the toolbar.

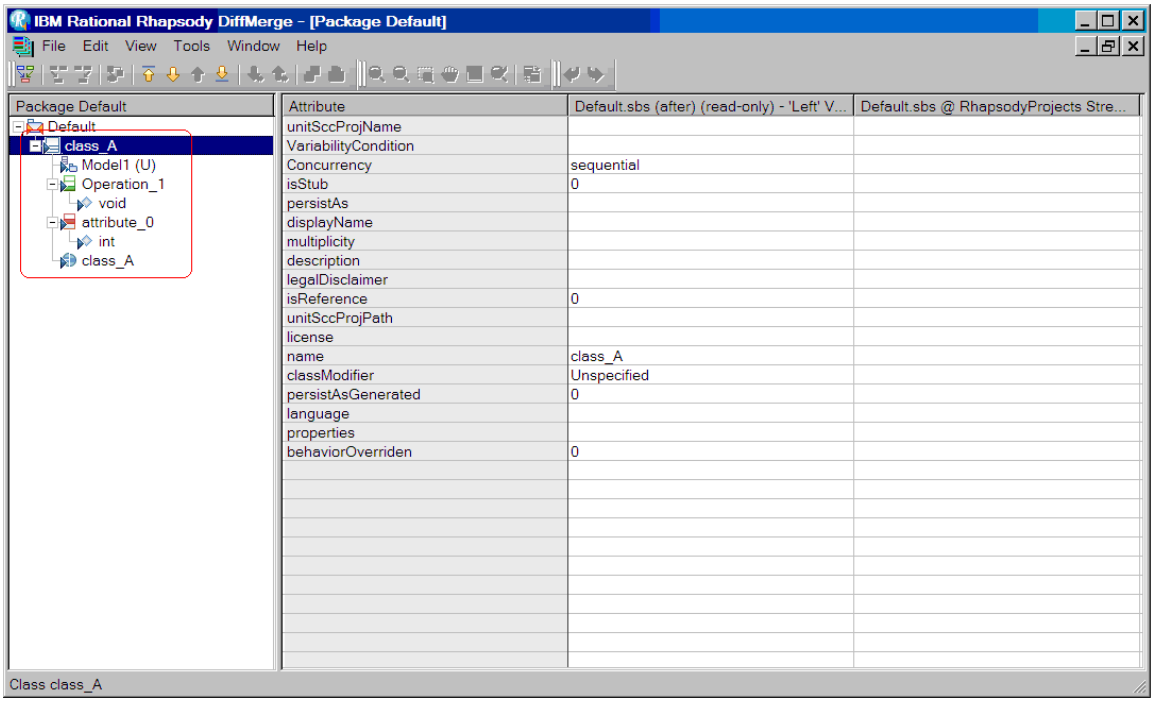


Open the change set in the compare editor to view the changes. Select the unit. In its pop-up menu, select **Open in Compare Editor**.

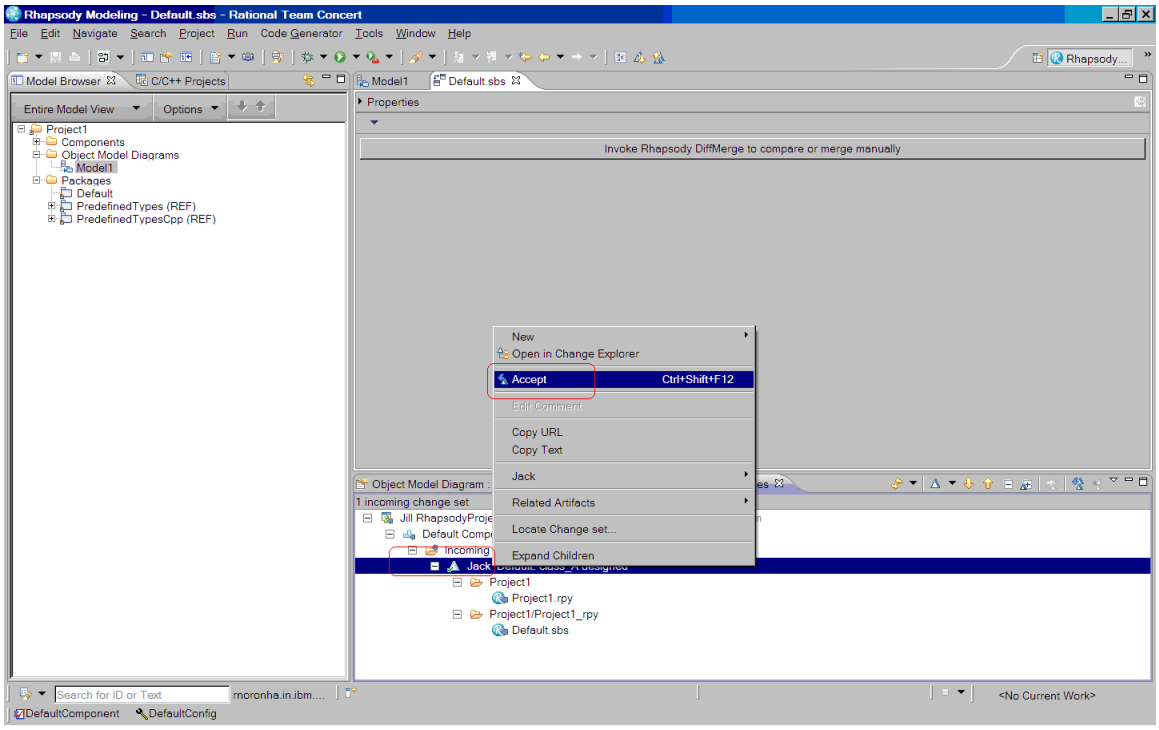


Using the built-in Rhapsody Diff-merge tool, you can view the changes in the Rhapsody unit. Invoke the Rhapsody Diff-merge utility to view the change set.

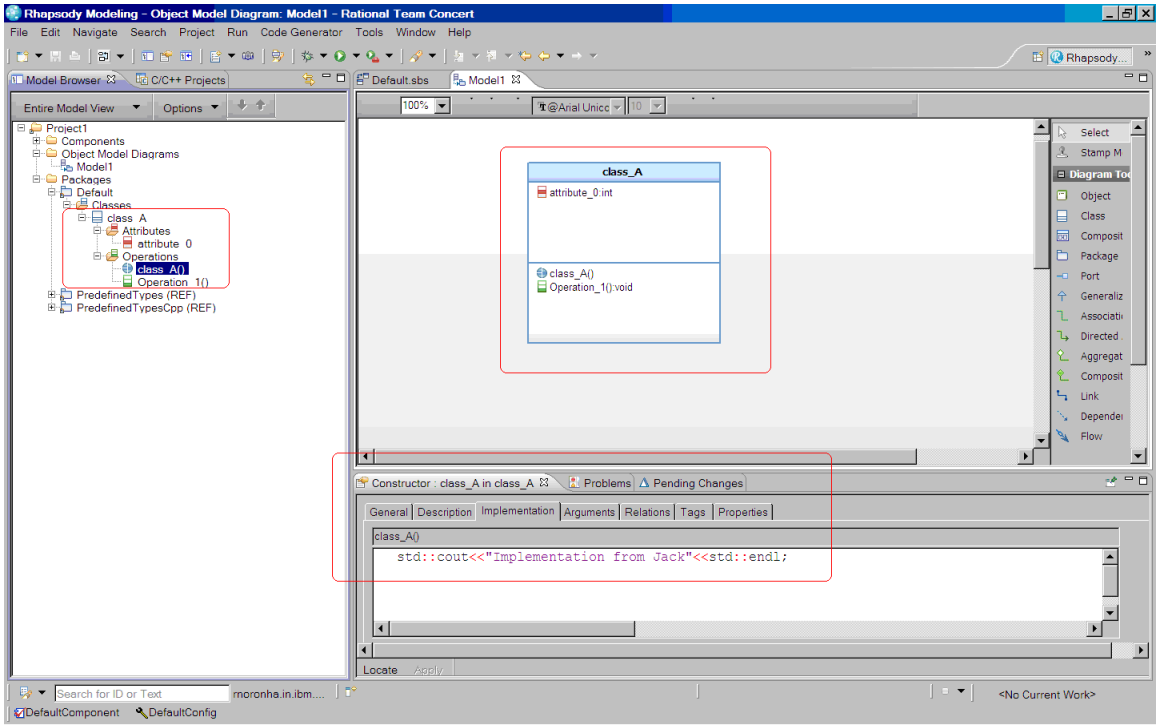




Accept the changes in the change set after verifying its contents.



The Rhapsody project browser and diagram area update to reflect the changes.



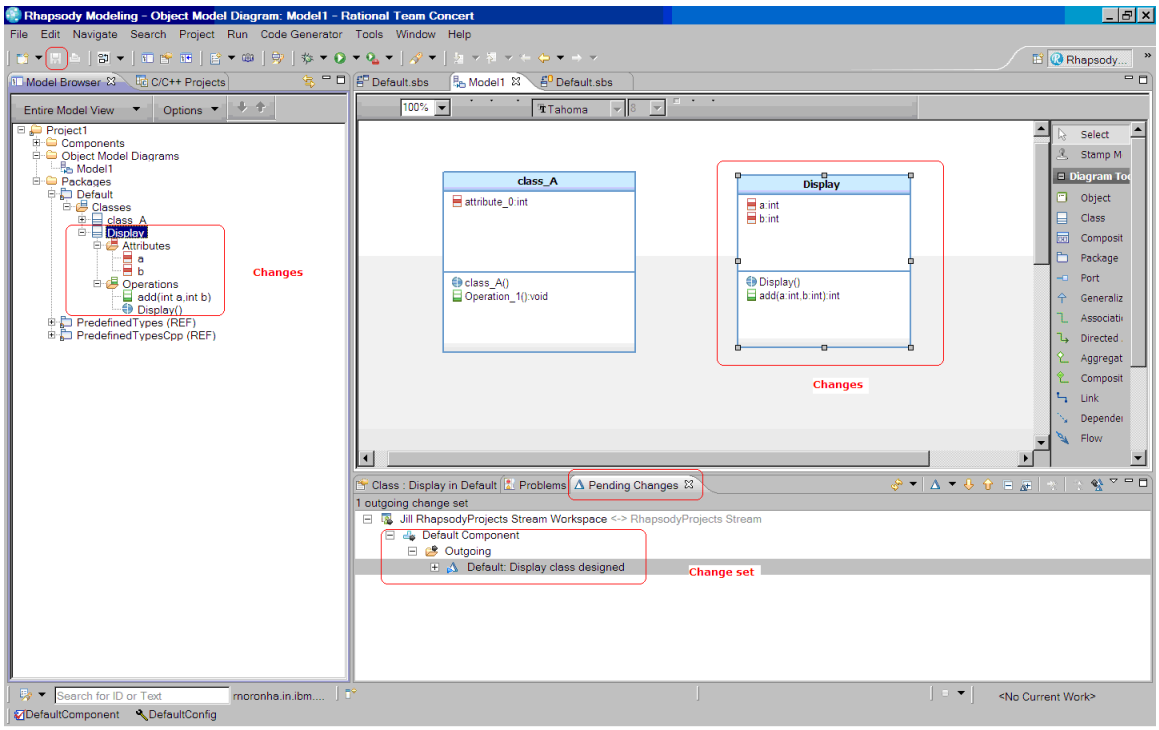
Arise of conflicts due to Parallel changes

Jill and Jack design a class artifact "display" in the same unit file and check-in (No Deliver) their respective change sets. The implementation of their respective class artifacts remain different. However, Jill delivers the change set to the project main stream before Jack. This causes Jack to deal to with the consequence of incoming change set against outgoing change set after refreshing the workspace.

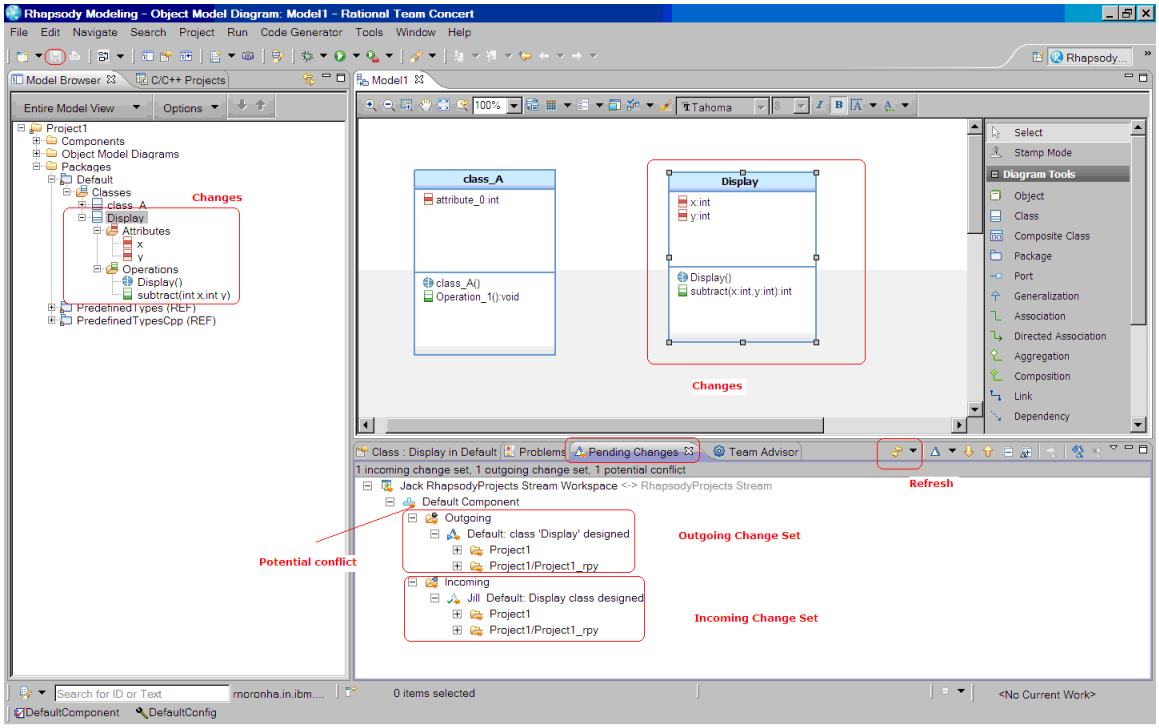
Since both the change sets are concerning the same unit file and contain almost similar information, Rational Team concert poses a potential conflict between the incoming and outgoing change set. Thus they need to resolve the conflict.

Jack needs to resolve this conflict before delivering the outgoing change set to keep his local workspace in sync with the project main stream.

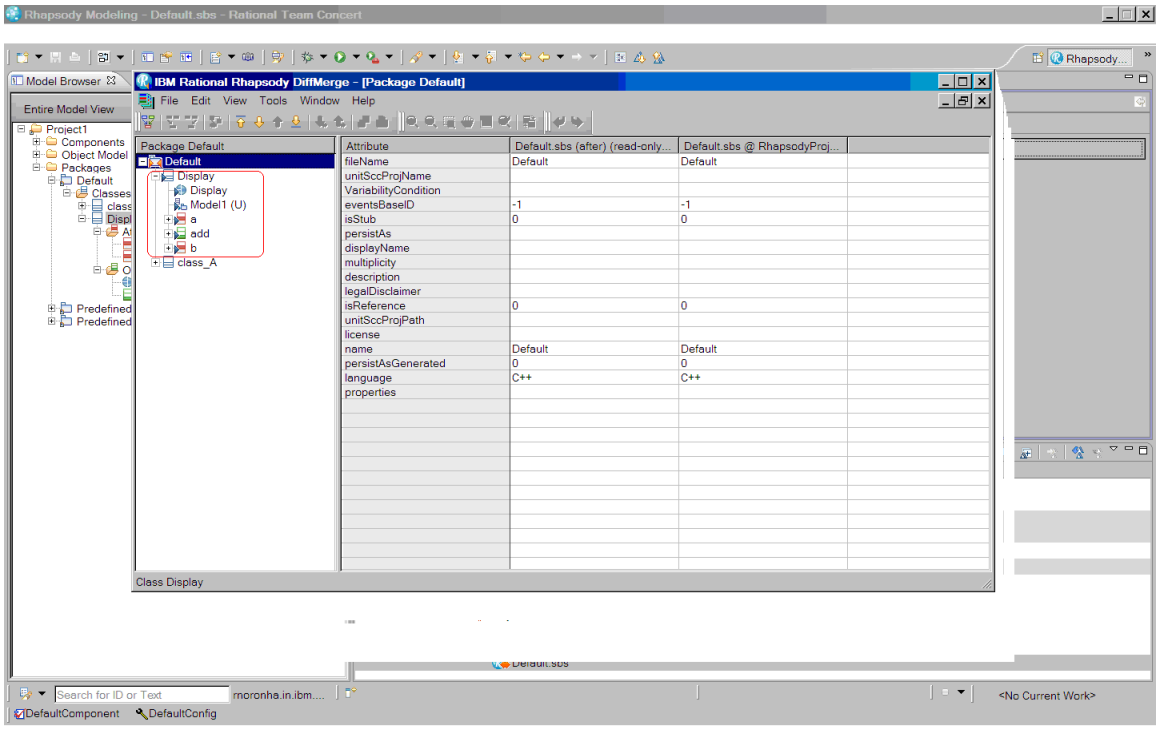
Here are the changes checked in and delivered by Jill.

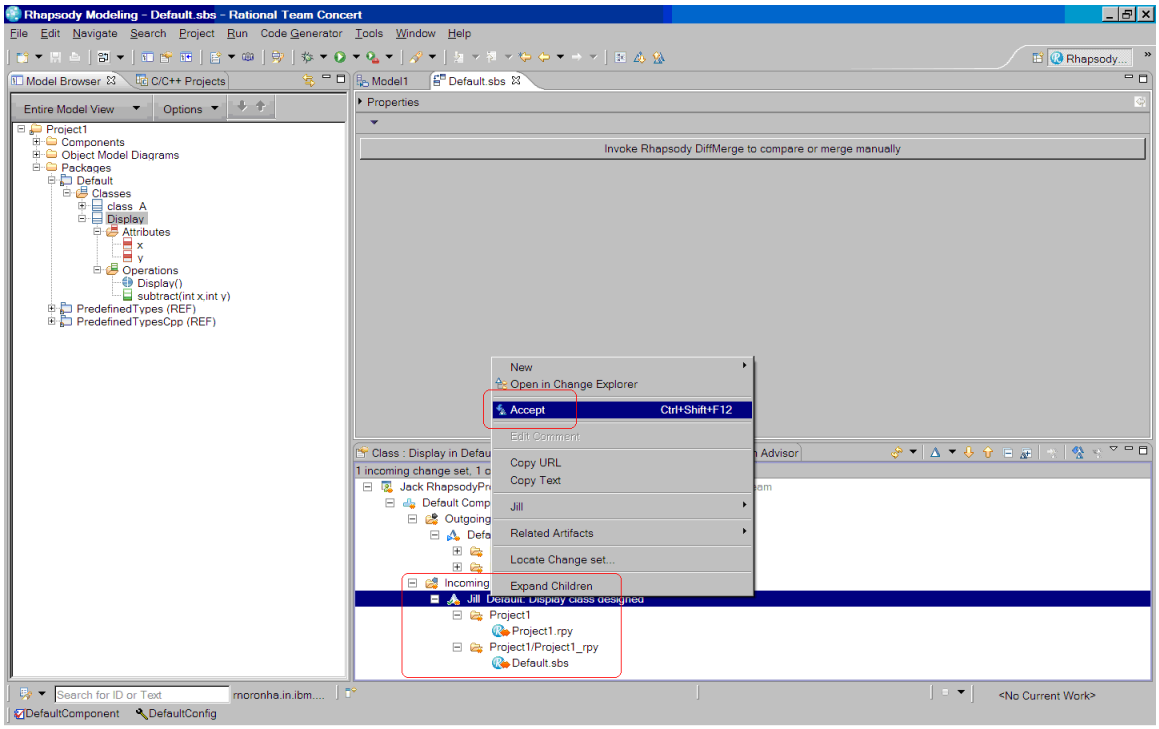


Here are the changes checked in by Jack. With the refreshed workspace, observe the incoming change set as a potential conflict

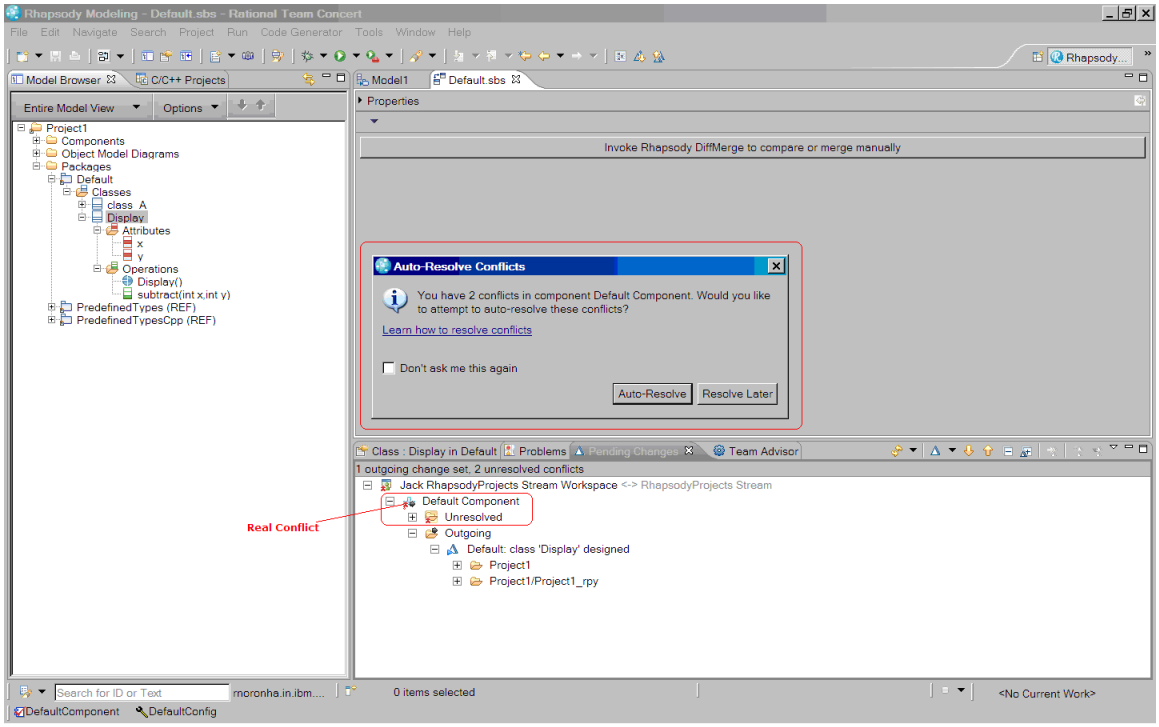


View the incoming conflicting change set in the Rhapsody Diff-merge tool and accept the change set.





Arise of a conflict in the unit due to different versions of the same file

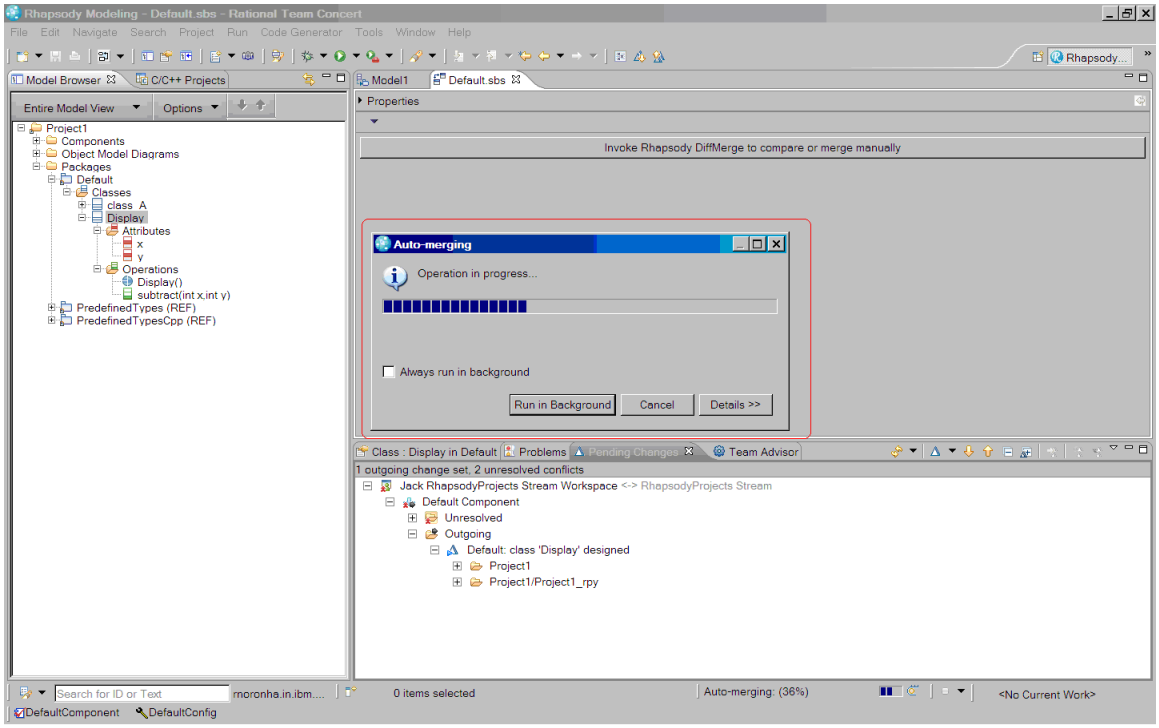


Resolving Conflicts

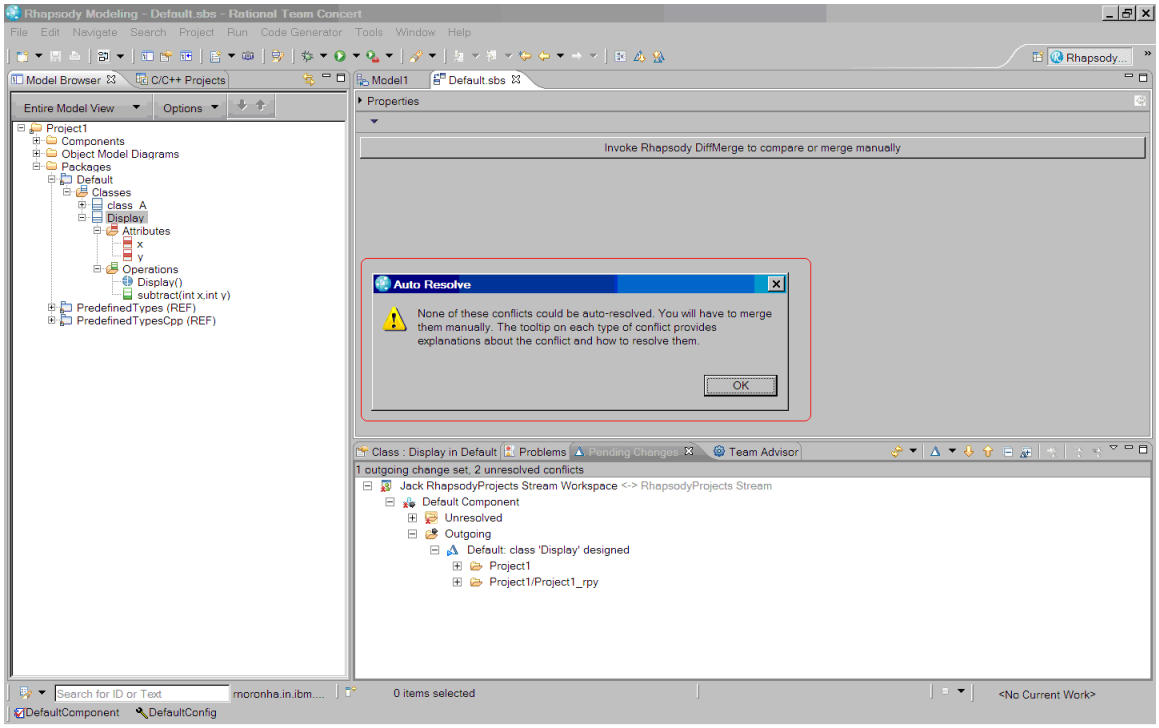
Before accepting the change set, Jack opens the potentially conflicting unit from the incoming change set in the Rhapsody Diff-merge tool. On deciding to incorporate the changes into the local workspace, he resolves the conflict and updates the project main stream with the merged unit.

Starting by accepting the incoming change set, the potential conflict turns into a real conflict resulting in different versions of the same unit.

Possible way of dealing with the conflict is to allow Team concert to auto-resolve the conflict or use the Rhapsody Diff-merge tool to manually compare the two units and merge the file with the necessary artifacts.
After resolving the conflict with the Rhapsody Diff-merge tool, Jack can reload the merged unit by selecting resolve as merged and deliver to the project main stream.

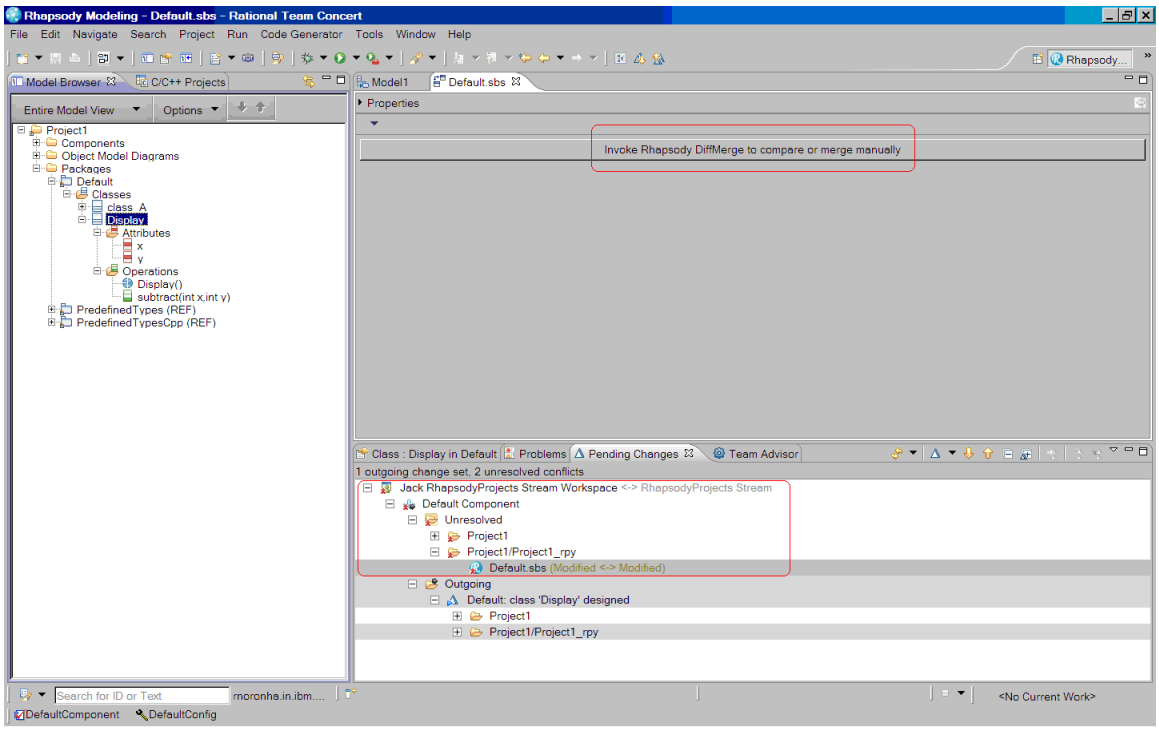


Attempt to auto-merge the conflicts that can be resolved



Failure to auto-resolve the conflicts

View the real conflicted unit. Use the Rhapsody Diff-merge tool to manually merge the unit.



You can resolve the conflicted unit in three ways:

Resolve with proposed

The project will overwrite the current versions in the repository workspace with the proposed changes in the accepted change set.

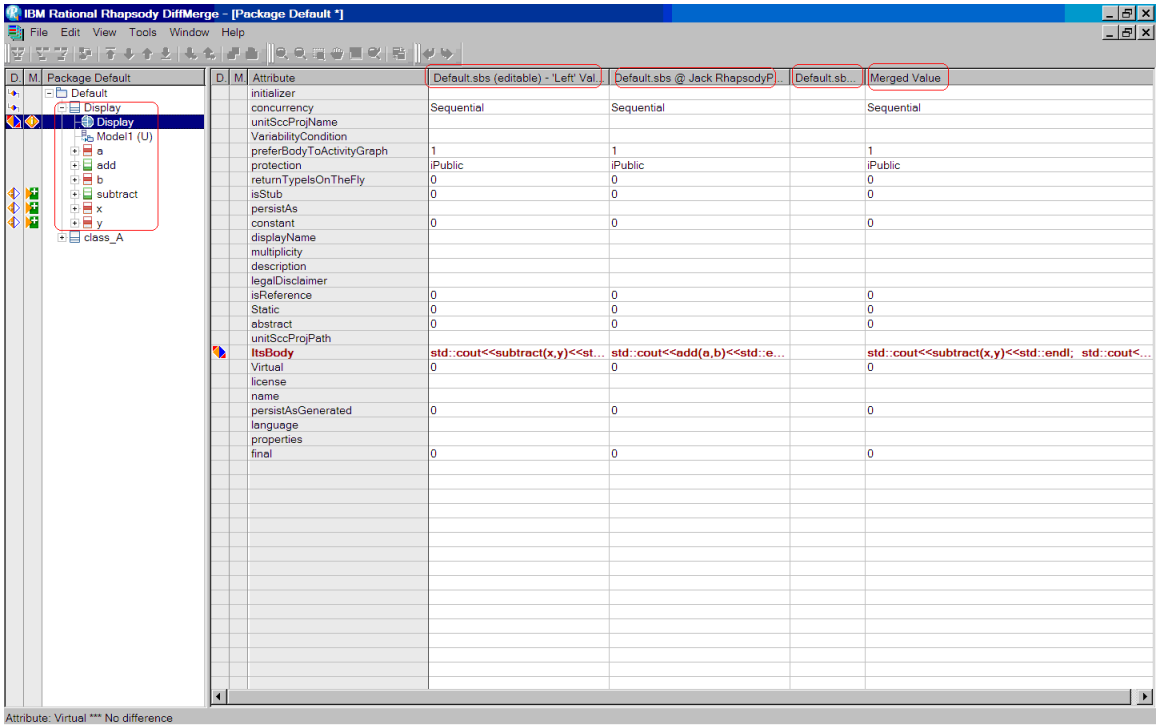
Resolve with Mine

The project will overwrite the proposed changes from the accepted change set with the current versions in the repository workspace.

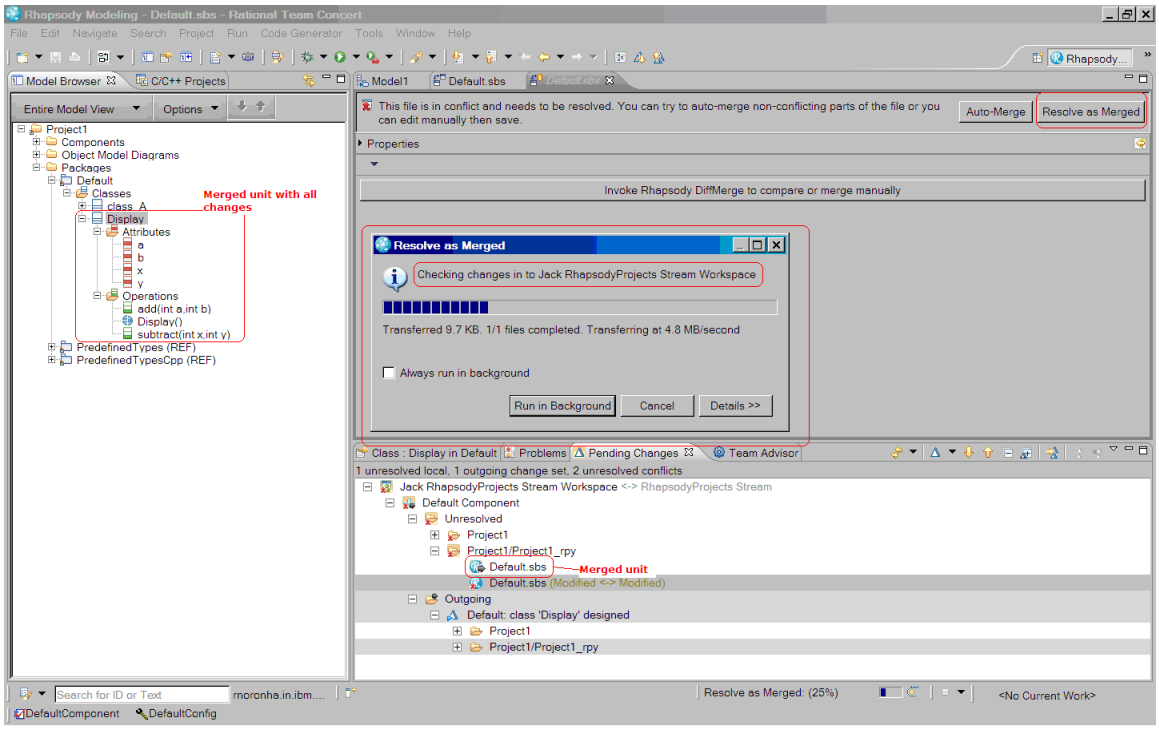
Resolve as merged

The project will overwrite the current versions in the repository workspace with the merged versions that have been manually edited to resolve the conflict.

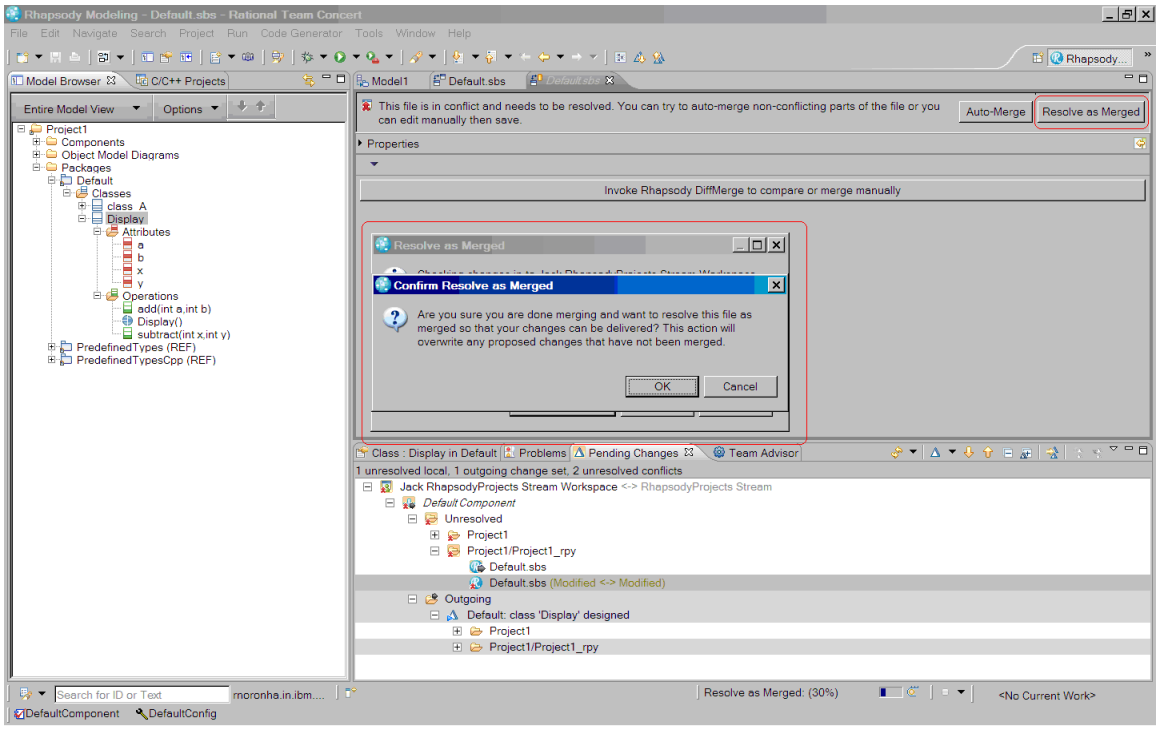
In this scenario, Jack manually merges the units with the Rhapsody Diff-merge tool as **Resolve as merged**. This is the chosen option because there is a need to incorporate the proposed changes with the local changes in the merged unit.



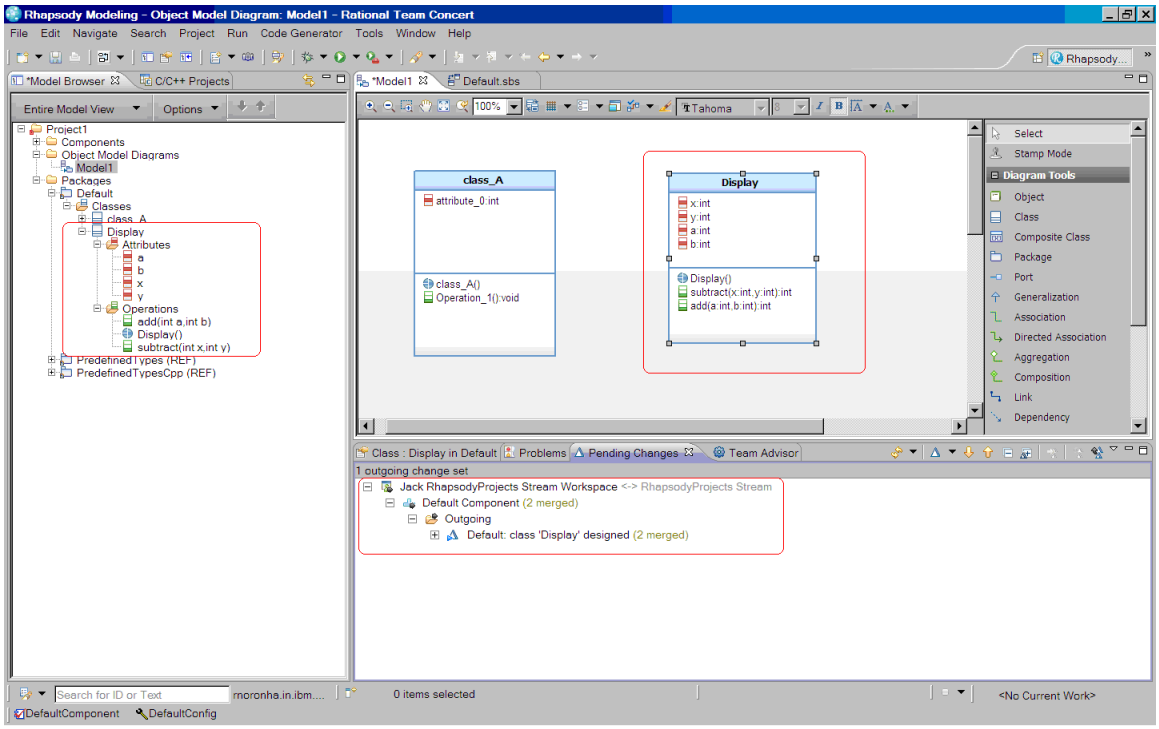
Merging the units with the Rhapsody Diff-merge tool



Merged unit updated in Jack's local workspace

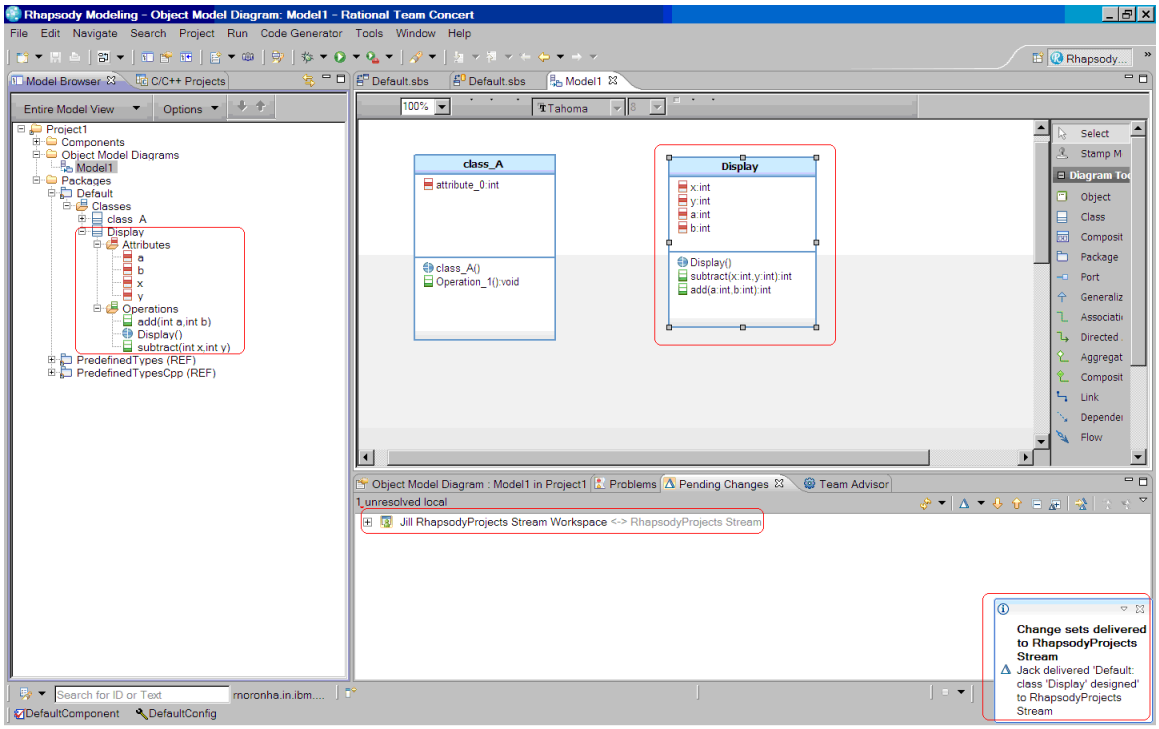


Conflict resolved as Merged



Conflict resolved at Jack's workspace

Jack delivers the merged change set. Jill accepts the change set and incorporates the changes in the local workspace.



In summary, you have basic source control operations with common scenarios while collaborating towards the development of a project using Rational Rhapsody and Rational Team Concert.

CONCLUSION

In this white paper introduces you to Rational Team Concert with Rhapsody for managing change and configuration management operations along with project management capabilities. This integration makes it easier for personnel from various roles in the software industry to collaborate with each other on a common platform. This collaboration improves the agility of the software development process.

At this point the platform integration between Rational Team Concert and Rhapsody is viable only on Windows operating systems

Apart from the parallel development process explored in the document, Rational Team Concert source control component also provides additional advanced features. These features include Component development, Task Integration, Distributed development, Move and rename tracking, Change set searching, Process integration, Dashboard contribution, and Event notification.

IBM Sametime Connect integrates with Rational Team Concert to facilitate instant communication between team members.

References

Key Rational Team Concert Terminology and Definitions

The Repository

Rational Team Concert source control uses a secure repository that is hosted on a server and accessed by clients through a URL. The repository stores objects such as streams, work items, and workspaces that help manage change flow. It also stores controlled artifacts that represent and can be retrieved as files or folders in a file system.

The repository is a secure database that is managed by the Jazz Team Server. It holds all of the objects that are required to support a team process, including components, workspaces, and team areas. Clients connect to a repository by using HTTP.

Project Area

The project area is a system representation of a software project. The project area defines the project deliverables, team structure, process, and schedule.

A project area is stored as a top-level or root item in a repository. A project area references project artifacts and stores the relationships between these artifacts

Team Area

The structure of the project teams is defined by a hierarchy of team areas. Use team areas to manage team membership, roles assignments, and team artifacts. Team areas serve these purposes:

1. Defines the users (team members) on the team and specifies their roles.
2. Defines the timeline in which the team is participating.
3. Customizes the project process for the team.

Work Item

Work item categories group work items by the various components or functional areas of your project. Each category is associated with a team area whose members are responsible for developing that component.

When you create a work item, you set its Filed Against (category) attribute. Members of the team area that is associated with that category receive notifications when the work item is created or modified.

Process

Process is the collection of roles, practices, rules, and guidelines that are used to organize and control the flow of work.

In Jazz, you use process to define user roles and their permissions for performing operations within the tool. If you have customized the process, you can create a process template and make it available to other teams. Process templates can include an informal description of the specified process.

Roles

Roles identify the functions of team members. Permissions for specific operations can be assigned to roles at the project level or within a team area.

Each project area can define a set of roles in the process configuration for the project, and the roles can be edited in the process configuration interface. Teams can also add their own roles in their team area customization.

Permissions

Permissions for performing operations are assigned to individual roles in the process configuration for the project area. You can assign permissions at the project or team level. You can also specify permissions for iteration types, iterations, and timelines

Streams and Components

Streams are like the branches that are found in other source control management systems, but have several additional capabilities. Any component in a repository can be included in zero or more streams. A stream can include at most one version of any component. By using multiple streams, a development organization can work on projects that use different versions of the same components. For example, a stream that is dedicated to the development of a new software release includes the most recent version of the release components. Another stream that is dedicated to the maintenance of an earlier version of that software release will initially include the component versions that were part of that release.

Repository workspaces and sandboxes

Repository workspaces are objects in the repository. Sandboxes are directories in your file system.

In the repository, files and folders are stored as versionable items whose data and metadata can be viewed but not directly modified. To support integration with file-based tools such as editors, compilers, and debuggers, files and folders in a repository workspace are loaded (copied) into a sandbox on your computer. As you modify files and folders in the sandbox, you periodically check them in, which copies the changes to the repository workspace. When all the changes in your sandbox are checked in, the repository and sandbox have the same content.

Change sets

A change set is a repository object that collects a related group of file, folder, and component modifications so that they can be applied to a flow target (workspace or stream) in a single operation.

The change set is the fundamental unit of change in Rational Team Concert source control. The contents of any workspace, component, or stream can be expressed as a collection of change sets, beginning with the one that is created when the initial set of projects was checked in. A change set can include changes to the contents of individual files and changes to a component namespace, such as delete, rename, and move operations.

Links

Jazz community site

https://jazz.net/projects/rational-team-concert/

Rational Team Concert Information Center

http://publib.boulder.ibm.com/infocenter/clmhelp/v3r0/index.jsp