

Visual Paradigm Teamwork Server Tutorial

Develop software in a collaborative way



Teamwork Server Tutorial

The software and documentation are furnished under the Teamwork Server license agreement and may be used only in accordance with the terms of the agreement.

Copyright Information

Copyright c 1999-2007 by Visual Paradigm. All rights reserved.

The material made available by Visual Paradigm in this document is protected under the laws and various international laws and treaties. No portion of this document or the material contained on it may be reproduced in any form or by any means without prior written permission from Visual Paradigm.

Every effort has been made to ensure the accuracy of this document. However, Visual Paradigm makes no warranties with respect to this documentation and disclaims any implied warranties of merchantability and fitness for a particular purpose. The information in this document is subject to change without notice.

All examples with names, company names, or companies that appear in this document are imaginary and do not refer to, or portray, in name or substance, any actual names, companies, entities, or institutions. Any resemblance to any real person, company, entity, or institution is purely coincidental.

Trademark Information

Teamwork Server is registered trademark of Visual Paradigm.

Sun, Sun ONE, Java, Java2, J2EE and EJB, NetBeans are all registered trademarks of Sun Microsystems, Inc.

Eclipse is registered trademark of Eclipse.

JBuilder is registered trademark of Borland Corporation.

IntelliJ and IntelliJ IDEA are registered trademarks of JetBrains.

Microsoft, Windows, Windows NT, Visio, and the Windows logo are trademarks or registered trademarks of Microsoft Corporation.

Oracle is a registered trademark, and JDeveloper is a trademark or registered trademark of Oracle Corporation.

BEA is registered trademarks of BEA Systems, Inc.

BEA WebLogic Workshop is trademark of BEA Systems, Inc.

Rational Rose is registered trademark of International Business Machines Corporation.

WinZip is a registered trademark of WinZip Computing, Inc.

Other trademarks or service marks referenced herein are property of their respective owners.

Teamwork Server License Agreement

THE USE OF THE SOFTWARE LICENSED TO YOU IS SUBJECT TO THE TERMS AND CONDITIONS OF THIS SOFTWARE LICENSE AGREEMENT. BY INSTALLING, COPYING, OR OTHERWISE USING THE SOFTWARE, YOU ACKNOWLEDGE THAT YOU HAVE READ THIS AGREEMENT, UNDERSTAND IT, AND AGREE TO BE BOUNDED BY ALL OF THE TERMS AND CONDITIONS OF THIS SOFTWARE LICENSE AGREEMENT.

1. **Limited License Grant.** Visual Paradigm grants to you ("the Licensee") a personal, non-exclusive, non-transferable, limited, perpetual, revocable license to install and use Visual Paradigm Products ("the Software" or "the Product"). The Licensee must not re-distribute the Software in whole or in part, either separately or included with a product.
2. **Restrictions.** The Software is confidential copyrighted information of Visual Paradigm, and Visual Paradigm and/or its licensors retain title to all copies. The Licensee shall not modify, adapt, decompile, disassemble, decrypt, extract, or otherwise reverse engineer the Software. Software may not be leased, rented, transferred, distributed, assigned, or sublicensed, in whole or in part. The Software contains valuable trade secrets. The Licensee promises not to extract any information or concepts from it as part of an effort to compete with the licensor, nor to assist anyone else in such an effort. The Licensee agrees not to remove, modify, delete or destroy any proprietary right notices of Visual Paradigm and its licensors, including copyright notices, in the Software.
3. **Disclaimer of Warranty.** The software and documentation are provided "AS IS," WITH NO WARRANTIES WHATSOEVER. ALL EXPRESS OR IMPLIED REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. THE ENTIRE RISK AS TO SATISFACTORY QUALITY, PERFORMANCE, ACCURACY AND EFFORT IS WITH THE LICENSEE. THERE IS NO WARRANTY THE DOCUMENTATION, Visual Paradigm's EFFORTS OR THE LICENSED SOFTWARE WILL FULFILL ANY OF LICENSEE'S PARTICULAR PURPOSES OR NEEDS. IF THESE WARRANTIES ARE UNENFORCEABLE UNDER APPLICABLE LAW, THEN Visual Paradigm DISCLAIMS SUCH WARRANTIES TO THE MAXIMUM EXTENT PERMITTED BY SUCH APPLICABLE LAW.
4. **Limitation of Liability.** Visual Paradigm AND ITS LICENSORS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY THE LICENSEE OR ANY THIRD PARTY AS A RESULT OF USING OR DISTRIBUTING SOFTWARE. IN NO EVENT WILL Visual Paradigm OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, EXEMPLARY, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE, EVEN IF Visual Paradigm HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

5. **Termination.** The Licensee may terminate this License at any time by destroying all copies of Software. Visual Paradigm will not be obligated to refund any License Fees, if any, paid by the Licensee for such termination. This License will terminate immediately without notice from Visual Paradigm if the Licensee fails to comply with any provision of this License. Upon such termination, the Licensee must destroy all copies of the Software. Visual Paradigm reserves all rights to terminate this License.

SPECIFIC DISCLAIMER FOR HIGH-RISK ACTIVITIES. The SOFTWARE is not designed or intended for use in high-risk activities including, without restricting the generality of the foregoing, on-line control of aircraft, air traffic, aircraft navigation or aircraft communications; or in the design, construction, operation or maintenance of any nuclear facility. Visual Paradigm disclaims any express or implied warranty of fitness for such purposes or any other purposes.

NOTICE. The Product is not intended for personal, family or household use; rather, it is intended exclusively for professional use. Its utilization requires skills that differ from those needed to use consumer software products such as word processing or spreadsheet software.

GOVERNMENT RIGHTS. If the Software is licensed by or on behalf of a unit or agency of any government, the Licensee agrees that the Software is "commercial computer software", "commercial computer software documentation" or similar terms and that, in the absence of a written agreement to the contrary, the Licensee's rights with respect to the Software are limited by the terms of this Agreement.

Acknowledgements

This Product includes software developed by the Apache Software Foundation (<http://www.apache.org>). Copyright© 1999 The Apache Software Foundation. All rights reserved.

Table of Contents

| | |
|--|------|
| Chapter 1 - Introduction | |
| Scenario | 1 -2 |
| Part 1 - Getting Started with Teamwork Server | |
| Chapter 2 - Installing Teamwork Server | |
| Chapter 3 - Teamwork Server Administration | |
| Creating Teamwork Projects | 3 -3 |
| Part 2 -Performing Teamwork Operations | |
| Chapter 4 - Checking-out Project | |
| Chapter 5 - Committing Project | |
| Chapter 6 - Handling Conflicts | |
| Chapter 7 - Checking for Update | |
| Checking update in VP-UML..... | 7 -2 |
| Chapter 8 - Importing and Exporting Project | |
| Exporting Project..... | 8 -2 |
| Modifying the Exported Project | 8 -2 |
| Importing the Exported Project | 8 -3 |
| Chapter 9 - Versioning | |

1

Introduction

Chapter 1 - Introduction

Visual Paradigm's Teamwork Server is a version control system for Visual Paradigm's products - Visual Paradigm for UML (VP-UML), Smart Development Environment (SDE), DB Visual ARCHITECT (DB-VA), Business Process Visual ARCHITECT (BP-VA) and Agilian (AG). Project comparison and merge operations are done automatically and transparent to the users, the development team can thus focus on the modeling tasks instead of complicated version control issues.

Our Teamwork Server is designed to be easy-to-use. By walking through the step-by-step scenario in this tutorial, you and your team members can work together even more smoothly.

Scenario

You are working in a software house, and your team is responsible for the development of a project called "Oriental Kitchen". The core members are:

| Member | Responsibility | Software Used |
|------------|--|---|
| sa | Administer the Teamwork Server. | Teamwork Server (which will be installed during tutorial) |
| vpuml_user | Use VP-UML to analyze and design the system using UML diagrams. | VP-UML (which has been installed before the tutorial) |
| sde_user | Use SDE together with Eclipse to implement the system from UML design using Java. | Eclipse, SDE for Eclipse (which has been installed before the tutorial) |
| dbva_user | Use DB-VA to design the relational data model using Entity Relationship Diagram (ERD). | DB-VA (which has been installed before the tutorial) |

Table 1.1

Are you ready? Let's go straight to the tutorial!



In this tutorial, one person can play more than one role if necessary. In fact, one person can play all roles even using one computer, as long as all the required software is installed.



If you purchased less than three Teamwork Server connections (users), you can still go through the tutorials by requesting an evaluation license key.

Part 1 – Getting Started with Teamwork Server

Part 1 - Getting Started with Teamwork Server

Visual Paradigm's Teamwork Server is a version control system for Visual Paradigm's products. This part shows you how to install the Teamwork Server and configure the Server Administration.

In this part:

- Installing Teamwork Server
- Teamwork Server Administration

2

Installing Teamwork Server

Chapter 2 - Installing Teamwork Server

Team member **sa** will be responsible for the installation of the Teamwork Server.

1. Open the Teamwork Server installer, the **Teamwork Server 3.0 Setup Wizard** is shown. Click **Next**.
2. In **License Agreement**, select **I accept the agreement** and click **Next**.
3. In **Select Destination Directory**, specify the destination directory for the Teamwork Server installation, click **Next**.
4. In **Select Start Menu Folder**, specify the Program group in the Start Menu folder, click **Next**.
5. In **Windows Services**, click **Next** to accept the default options to allow the Teamwork Server is installed as a Windows service.

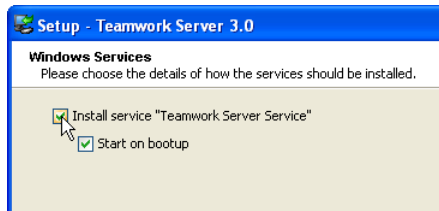


Figure 2.1 - Install service "Teamwork Server services" options

6. In **Installation Type**, select **Server and Admin** to install both the Teamwork Server and the administration program on this computer. Click **Next**.

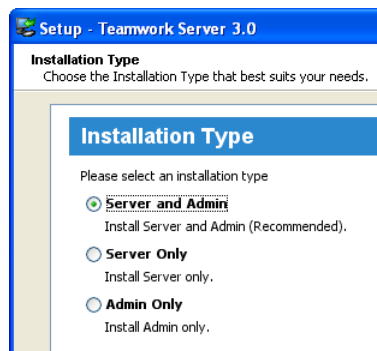


Figure 2.2 - Installation types

7. In **Server Configuration**, accept the default port (1999) and repository of Teamwork Server. Select the license key to be imported and click **Next** to start the installation.



It is very important that a valid license key must be imported to the Teamwork Server; otherwise it will not be functional.

8. When installation is completed, click **Finish** to exit the wizard.

The **Teamwork Administrator** will be started asking for configuring the server administration which will be introduced in the next chapter, [Teamwork Server Administration](#).

3

Teamwork Server Administration

Chapter 3 - Teamwork Server Administration

Team member **sa** will be responsible for the administration of the Teamwork Server.

1. When installation is completed, the **Teamwork Administrator** will be started for initial server administration.
2. In **Change Administrator Password**, specify a new password for the Teamwork Server administrator. Click **Next**.
3. In **Create User**, add the three users with user name **vpuml_user**, **sde_user** and **dbva_user**. It can be done by entering the user information including the user name and password and then clicking the **Add User** button repeatedly. Do not check any permission allowed for these users.

| Operation | Allowe... |
|----------------|--------------------------|
| Create project | <input type="checkbox"/> |
| Update project | <input type="checkbox"/> |
| Delete project | <input type="checkbox"/> |
| Create user | <input type="checkbox"/> |
| Update user | <input type="checkbox"/> |
| Delete user | <input type="checkbox"/> |

Figure 3.1 - Create User and assign permission

If a user is added successfully, a message "User is added" will be shown at the bottom of the **New User** pane.

User "vpuml_user" is added.

Figure 3.2 - Add user successful message

After adding the three users, the user which is added successfully will be displayed in the table of **Existing Users**.

| User Name | Permissions |
|------------|-------------|
| dbva_user | |
| sde_user | |
| vpuml_user | |

Figure 3.3 - The existing users

4. Click **Finish** to exit the **Teamwork Administrator**.

Creating Teamwork Projects

The project manager has decided to work on two modeling projects, namely P1 and P2 for the "Oriental Kitchen" project.

- P1 is a project with an initial class diagram design. It will be handled by vpuml_user and sde_user.
- P2 is a brand new project for relational data modeling. It will be handled by vpuml_user and dbva_user.

Team member **sa**, as the Teamwork Server administrator, will now add these projects to the Teamwork Server and assign the related users to the projects.

1. Click the **Teamwork Server 3.0 Administrator** shortcut to start the **Teamwork Server Administration**.
2. Enter **Admin** for **User name** and enter the **Password**. Fill in the host name or IP address of the Teamwork Server in **Server host**. Enter the **Port number** (by default, **1999**). Click **Login** to login to the server.



Figure 3.4 - Login to the Teamwork Server dialog

3. Click the **Add Project** button on the toolbar.

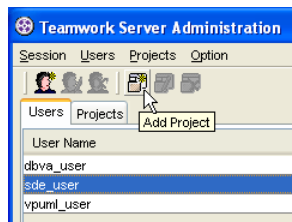


Figure 3.5 - Add Project button on toolbar

The **Add Project** dialog box is displayed.

4. Enter **P1** as the **Project name** (**Relative project directory** will be filled in automatically). Select **Import existing project** and specify the file path of **P1.vpp** for this tutorial. Select users, **vpuml_user** and **sde_user** and click **Add as Read and Update** to assign them to this project.

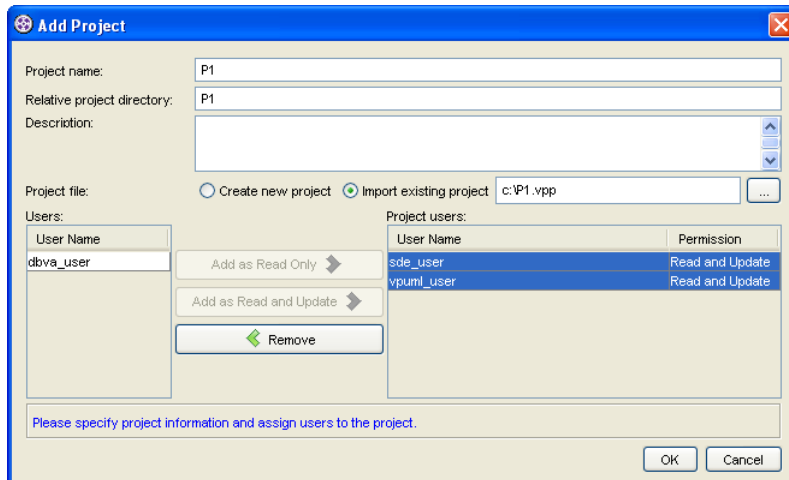


Figure 3.6 - Add an existing project and assign User to project



Granting the **Update project** permission to users allows them to commit the updated project to the server. Otherwise, they do not have the right to commit project to server.

5. Click **OK** to create the project.
6. Click the **Add Project** button on the toolbar.
7. In the **Add Project** dialog box, enter **P2** as the **Project name** and select **Create new project**. Select users, **vpuml_user** and **dbva_user** and click the **Add as Read and Update** button to assign them to this project.
8. Click **OK** to create the project.

After creating the projects, exit the **Teamwork Server Administration** by selecting the **Exit** option from the **Session** menu.

Part 2 – Performing Teamwork Operations

Part 2 -Performing Teamwork Operations

After completing the Teamwork Server setting and creating Teamwork Projects for the tutorial, this part will help you better understand the Teamwork Operations performed by the teamwork clients.

In this part:

- Checking-out Project
- Committing Project
- Handling Conflicts
- Checking for Update
- Importing and Exporting Project
- Versioning

4

Checking-out Project

Chapter 4 - Checking-out Project

As the two modeling projects, P1 and P2 have been created, the users are going to work on the projects independently and simultaneously.

Team member **vpuml_user** who is responsible for both projects P1 and P2, is going to check out project P1 using VP-UML.

1. On the menu, select **Tools > Teamwork > Open Teamwork Client....**

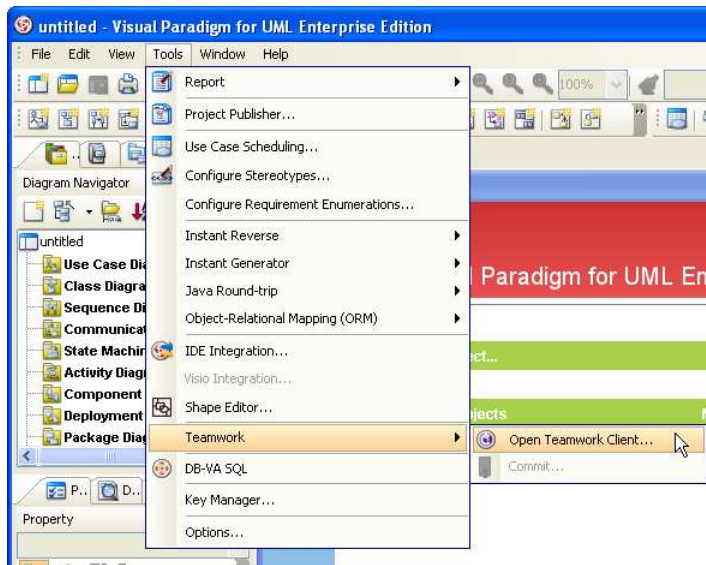


Figure 4.1 - To open Teamwork Client

The **Login to the Teamwork Server** dialog box is displayed.

2. Enter **vpuml_user** as the **User name** and other information including **Password**, **Server host** and **Port number**.



Figure 4.2 - Login to the Teamwork Server dialog

If **vpuml_user** logs successfully, the **Teamwork Client** starts up.

3. Select **P1** from the list of **Projects** and then click **Checkout Project** button. After checking-out project, P1, it will be opened in VP-UML.

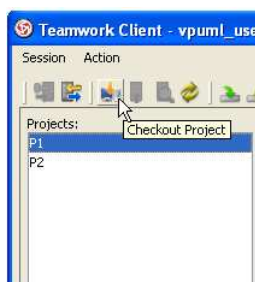


Figure 4.3 - Checkout Project

4. On the **Diagram Navigator**, double-click the class diagram, **Domain Class Diagram** to open it.

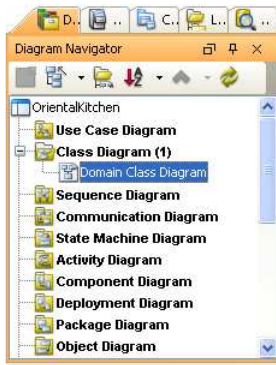


Figure 4.4 - Diagram Navigator

User, **vpuml_user** can examine the domain class for the **Oriental Kitchen** project.

Domain Class Diagram of the Oriental Kitchen project:

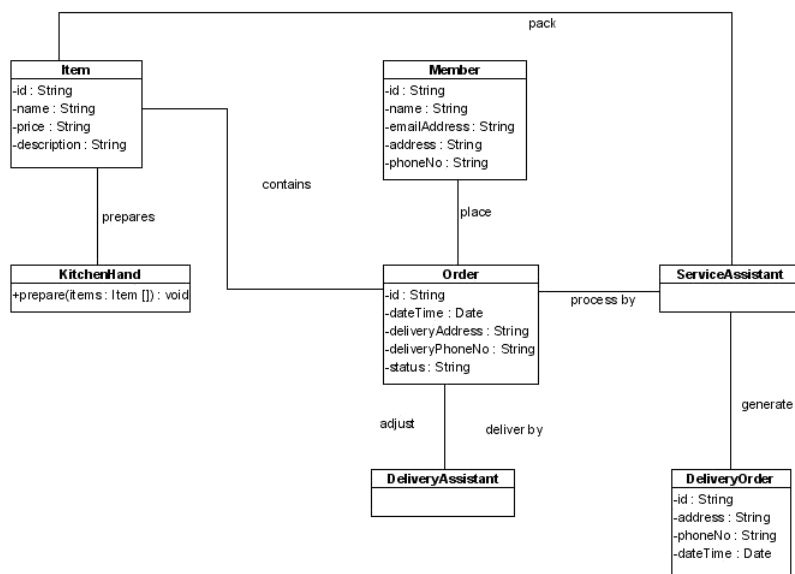


Figure 4.5 - Domain Class Diagram of the Oriental Kitchen project

Team member, **sde_user** who is responsible for project P1, is going to check out project using Eclipse together with SDE-EC.

1. Create a Java Project in Eclipse, named as **OrientalKitchen**.
2. Right-click on the project, **OrientalKitchen**, select **SDE-EC Project > Open Teamwork Client...** from the popup menu.

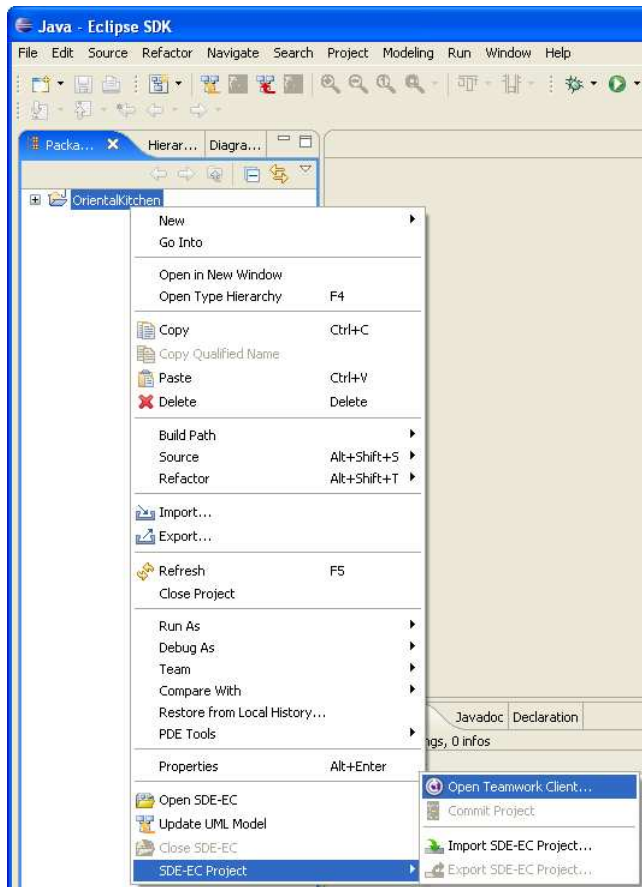


Figure 4.6 - To open Teamwork client

3. As the **Login to the Teamwork Server** dialog box is displayed, enter the login information for **sde_user** to connect to the Teamwork Server and launch the **Teamwork Client**.

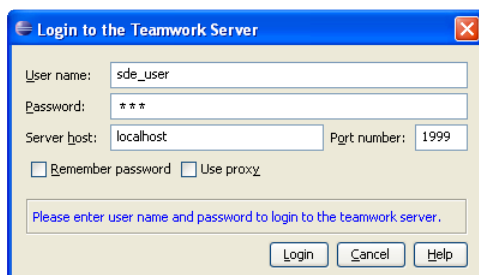


Figure 4.7 - Login to the Teamwork Server dialog

4. Check out and open project P1 by using the **Checkout Project** button.

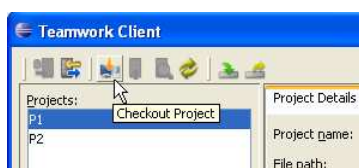


Figure 4.8 - Checkout Project button

- On the **Diagram Navigator**, double-click the class diagram, **Domain Class Diagram** to open it. User, **sde_user** can examine the domain class for the project.

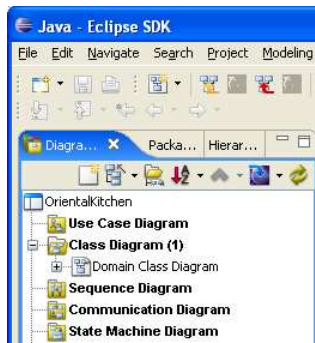


Figure 4.9 - Diagram Navigator

After opening the class diagram, **sde_user** is going to generate Java code for all classes.

- Generate Java code for all classes from the object model in one of the two ways:

Inside the Teamwork working environment, to generate Java code from the object model, you must select the classes which will be used for generating code. The Java code will be generated and added to the Eclipse project automatically.

- Select all classes from the **Domain Class Diagram**, click the **Update to Code** button on the toolbar.

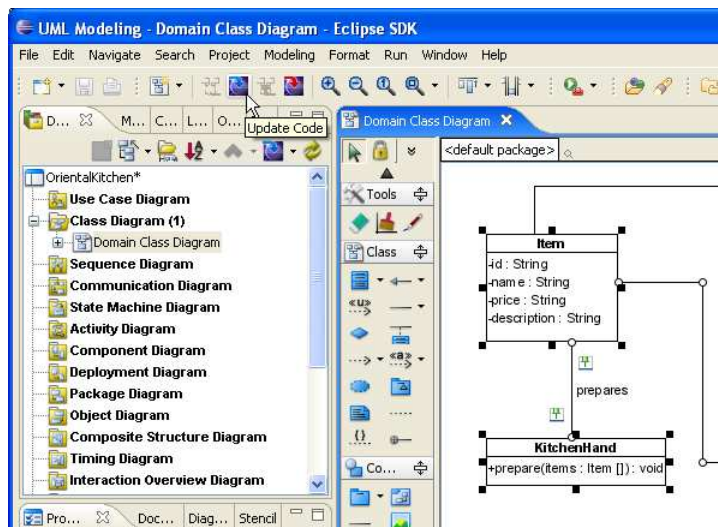


Figure 4.10 - Update Code from selected UML model

- Select all classes from the **Domain Class Diagram**, right-click on a class, select **Update to Code** from the popup menu.

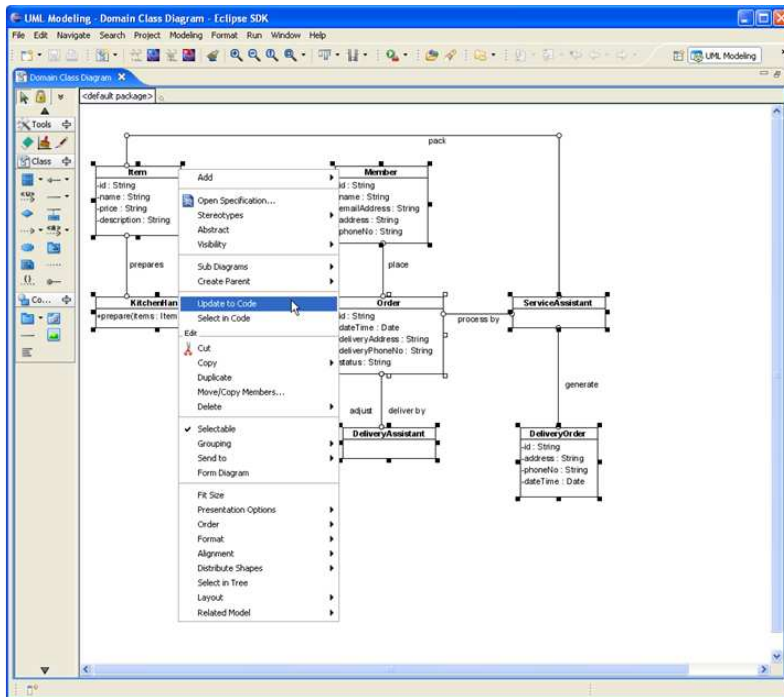


Figure 4.11 - Update the code by clicking on the popup menu

2. Open the Item.java, add the getter and setter methods for all attributes.

After examining the domain class for the project, team member, **vpuml_user** is going to:

1. Rename the class, **Member** to **Customer** by double-clicking on the class.

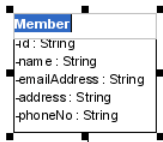


Figure 4.12 - Rename the Class

2. Delete the attribute, **emailAddress** from the **Customer** class by right-clicking the attribute and selecting **Delete** from the popup menu.

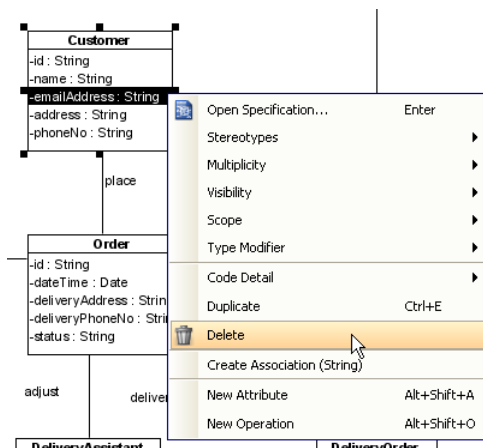


Figure 4.13 - Delete an attribute

Team member, **dbva_user** who is responsible for project P2, is going to checkout project P2:

1. On the menu, select **Tools > Teamwork > Open Teamwork Client....**

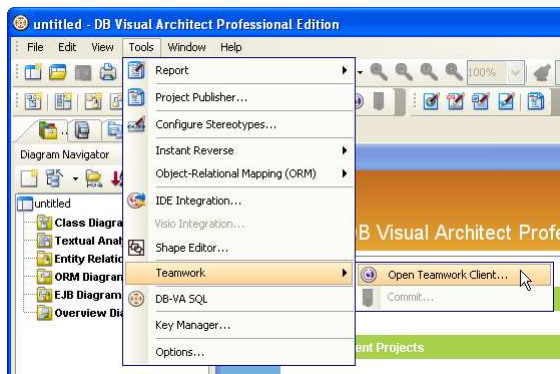


Figure 4.14 - To Open Teamwork Client

2. As the **Login to the Teamwork Server** dialog box is displayed, enter the login information for **dbva_user** to connect to the **Teamwork Server** and launch the **Teamwork Client**.



Figure 4.15 - Login to the Teamwork Server dialog

3. Check out and open project P1 by using the **Checkout Project** button.

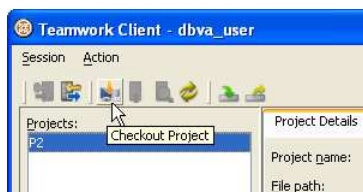


Figure 4.16 - Checkout Project

4. Draw an Entity Relationship Diagram as shown below.

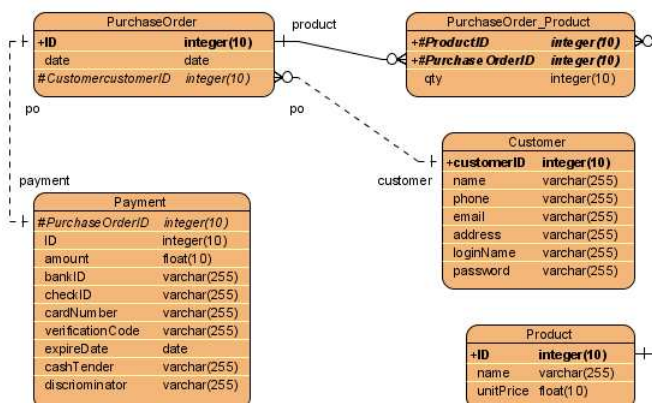


Figure 4.17 - Entities Relationship Diagram

5

Committing Project

Chapter 5 - Committing Project

As the team members, **vpuml_user**, **sde_user** and **dbva_user** work on the teamwork project independently, they will commit the changes to the Teamwork Server after they have completed the changes.

Team member, **sde_user** is going to reverse engineer the modified Java source code to UML models.

1. Navigate to the Java editor for **Item.java**.
2. Click the **Update UML Model** button on the toolbar.



Figure 5.1 - Update UML Model button

After reverse engineering the source code to UML Model, the model for Item class will be updated as shown below.

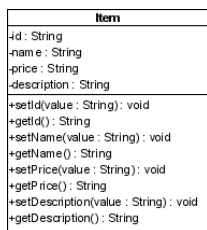


Figure 5.2 - Reversed Class

After modifying the domain class for **Oriental Kitchen** project, **sde_user** is going to commit the changes.

1. Right-click on the project, **OrientalKitchen** from the **Package Explorer**, select **SDE-EC Project > Commit Project** from the popup menu. The **Commit Project** dialog box will be displayed.

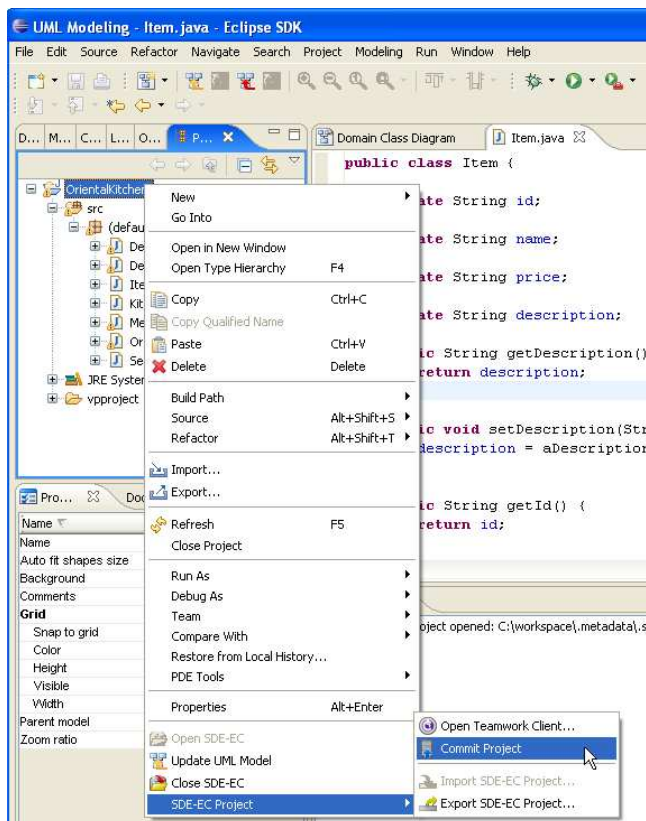


Figure 5.3 - Commit Project in SDE

2. Enter the commit reason "**Item class is updated by adding the getter and setter methods to all attributes**" to the **Description** and click **OK**. The **Commit Model(s)** dialog box will be displayed.

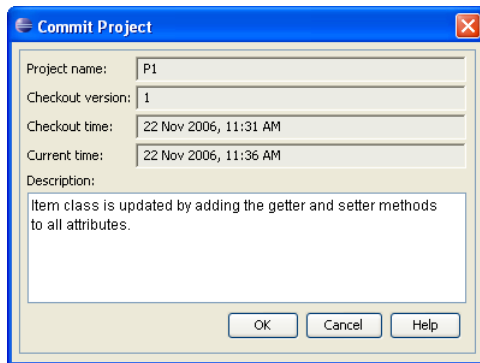


Figure 5.4 - Commit Project dialog

3. Since it is the first change committed to the teamwork project, there should be no conflict found. Click **OK** to proceed with committing the project. Hence, the teamwork project, P1 is updated in the repository of teamwork server.

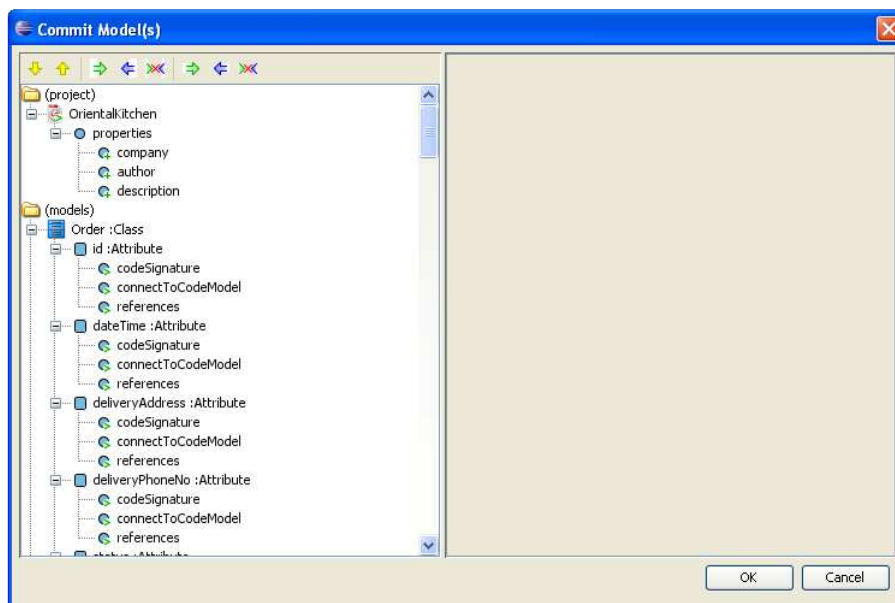


Figure 5.5 - Commit Model(s) dialog

Team member, **vpuml_user** is going to commit changes for renaming **Member** class to **Customer** and deleting the email address attribute.

1. On the menu, select **Tools > Teamwork > Commit...**. The **Commit Project** dialog box will be displayed.

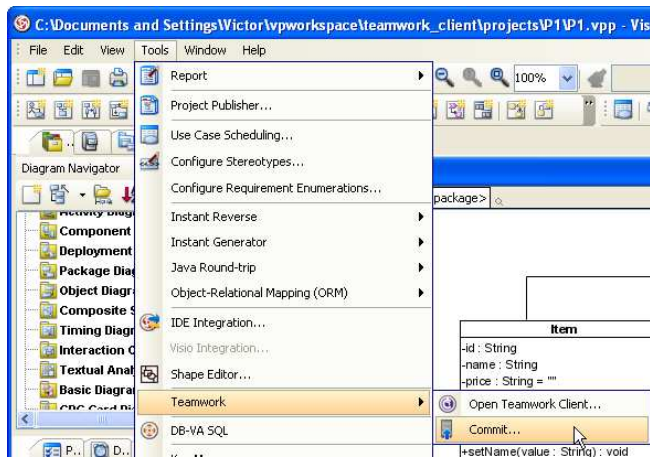


Figure 5.6 - To commit project in VP-UML

2. Enter the description as "**1. Member class was renamed to Customer. 2. Attribute, emailAddress was removed from Customer class.**", and then click **OK**. The **Commit Model(s)** dialog box will be displayed.

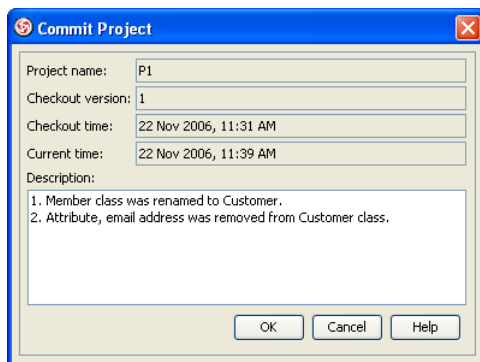


Figure 5.7 - Commit Project dialog

Since the changes made by **vpuml_user** were not related to that made by **sde_user**, there should be no conflict found. Click **OK** to proceed with committing the project. Hence, the teamwork project, P1 is updated in the repository of teamwork server.

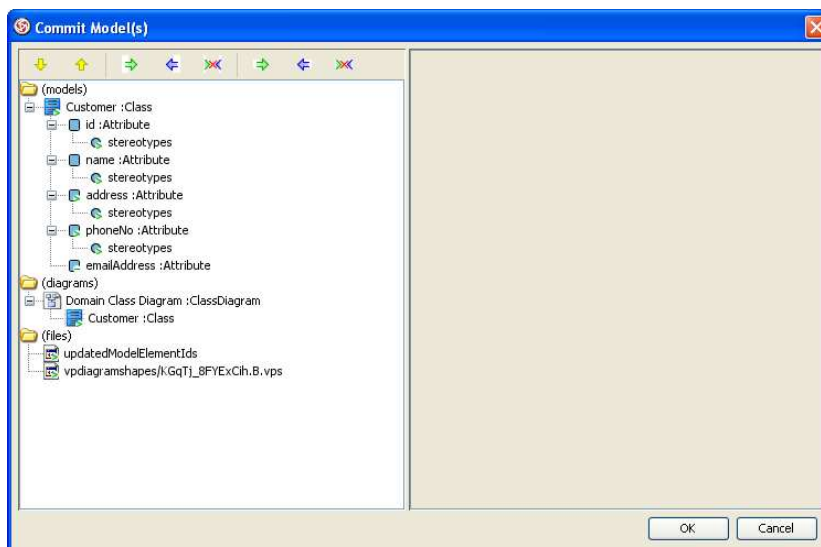


Figure 5.8 - Commit Model(s) dialog

Team member, **dbva_user** is going to commit project after the entity relationship diagram is completed.

1. On the menu, select **Tools > Teamwork > Commit....** The **Commit Project** dialog box will be displayed.

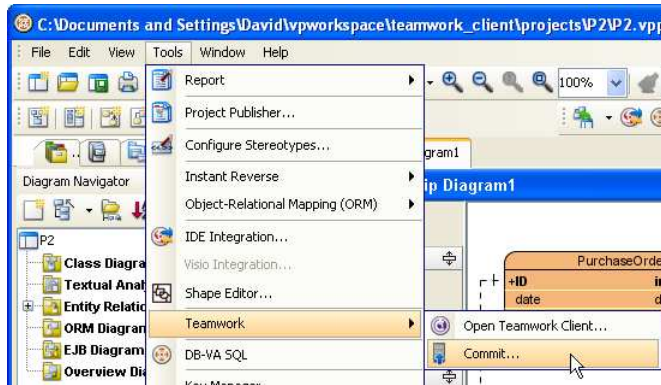


Figure 5.9 - To commit project in DB-VA

2. Enter the description as "**An ERD is created for designing the relational data model.**", and then click **OK**. The **Commit Model(s)** dialog box will be displayed.

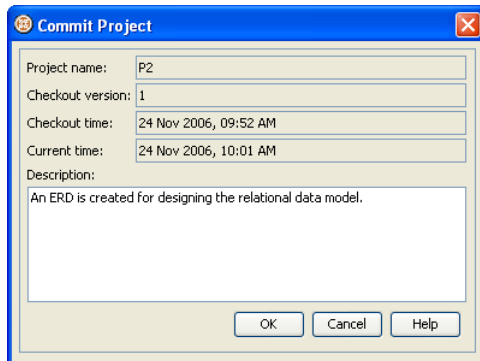


Figure 5.10 - Commit Project dialog

3. Since the Project P2 was created as a new project and adding the ERD to the project is the first change committed, there should be no conflict when **dbva_user** commits project. Click the **OK** to proceed with committing the project. Hence, the teamwork project, P2 is updated in the repository of teamwork server.

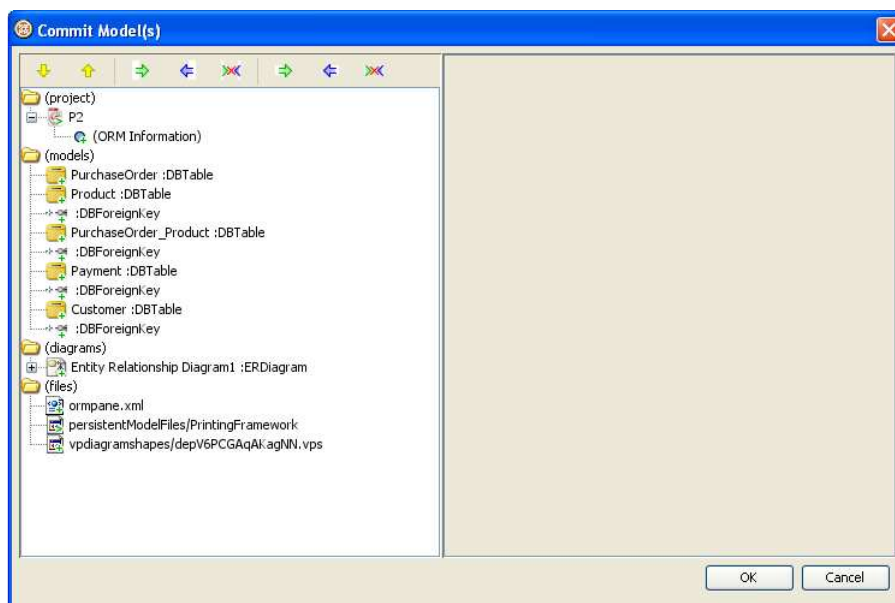


Figure 5.11 - Commit Model(s) dialog

6

Handling Conflicts

Chapter 6 - Handling Conflicts

After the first round of changes committed by the team members, the team members are going to make further changes to the project.

Team member, **vpuml_user** is going to change the data type of attribute, price for the **Item** class from String to double.

1. Select **price**, an attribute of the **Item** class in the class diagram, right-click and select **Open Specification....** The **Attribute Specification** dialog box is displayed.

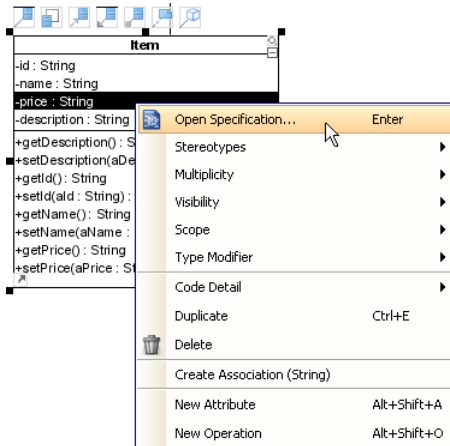


Figure 6.1 - To open specification

2. Enter **"US\$ 0"** for the **Initial value**, and click **OK**.

The initial value of price is set and represented by **" price : String = "US\$ 0"** shown on the class diagram.

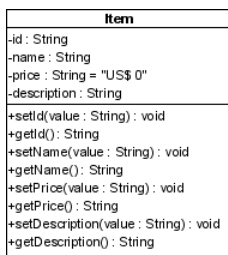


Figure 6.2 - The Item Class is updated

3. Commit project by entering the description as **"The initial value of attribute, price for Item class is set to "US\$ 0".**", and click **OK**. The **Commit Model(s)** dialog box will be displayed.

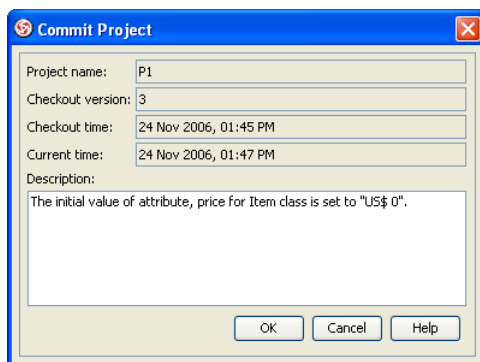


Figure 6.3 - Commit Project dialog

- Click **OK** to proceed with committing the project. Hence, the teamwork project, P1 is updated in the repository of teamwork server.

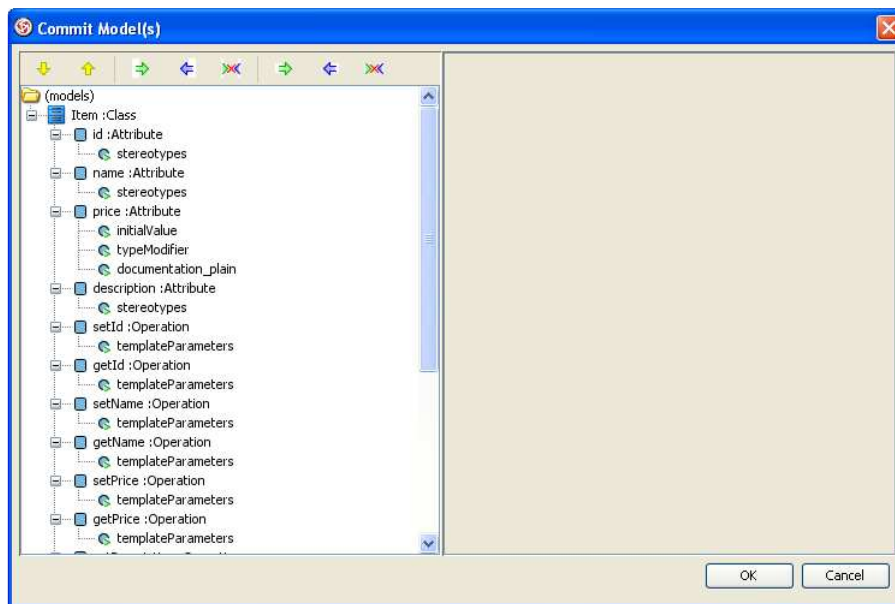


Figure 6.4 - Commit Model(s) dialog

In the meantime, team member, **sde_user** is going to set the initial value of attribute, price for the **Item** class to an empty string, "".

- Select **price**, an attribute of the **Item** class in the class diagram, right-click and select **Open Specification....** The **Attribute Specification** dialog box is displayed.
- Enter "" for the **Initial value**, and click **OK**.

The initial value of price is set and represented by "**price : String = ""**" shown on the class diagram.

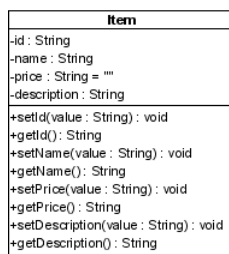


Figure 6.5 - Attributes with default value

- Commit project by entering the description as "**The initial value of attribute, price for Item class is set to "" (empty string).**", and click **OK**. The **Commit Model(s)** dialog box will be displayed.

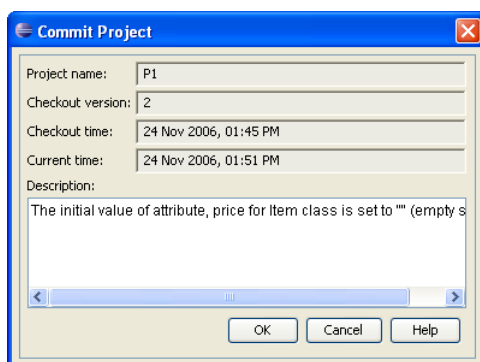


Figure 6.6 - Commit Project dialog

- Since team members, **vpuml_user** and **sde_user** work on the same property, the initial value of the price attribute of Item class at the same time, and **sde_user** committed project after **vpuml_user** committed project, a conflict was detected by the Teamwork Server.

The cross sign is added to the property **initial value**, indicating that the property causes a conflict.

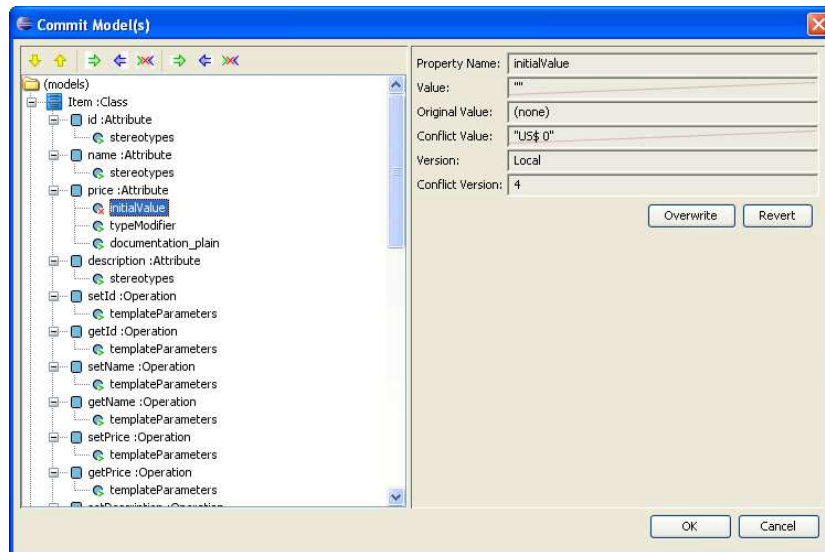


Figure 6.7 - Commit Model(s) dialog

- Click on the **Overwrite** button to overwrite the changes made by another team member, **vpuml_user**.

As **vpuml_user** is responsible for the two teamwork projects, P1 and P2, **vpuml_user** is going to check out project P2 and work on the relational data model.

- Check out and open project P2 by using the **Checkout Project** and **Open Project** buttons from the **Teamwork Client**.
- Open the entity relationship diagram.
- To change the phrase of **PurchaseOrder** on the relationship between **PurchaseOrder_Product** and **PurchaseOrder** entities, select **Open Specification...** from the popup menu. The **Relationship Specification** dialog box is displayed.
- Enter **"has"** for the **Phrase** on **PurchaseOrder** and click **OK**.

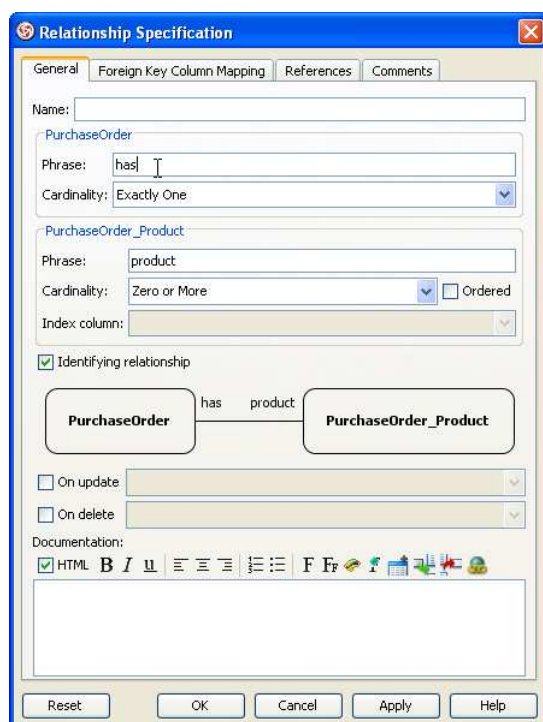


Figure 6.8 - Relationship specification dialog

5. Commit project by entering the description as "**Phrase for PurchaseOrder is set as "has".**" and click **OK**. The **Commit Model(s)** dialog box will be displayed.

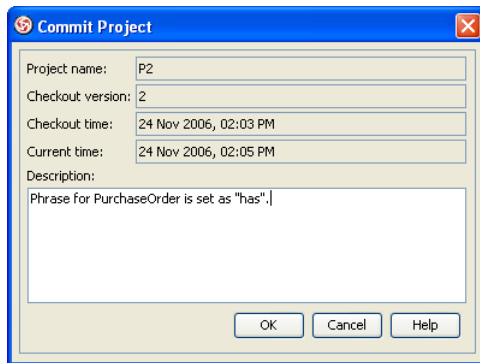


Figure 6.9 - Commit Project dialog

6. Click **OK** to proceed with committing the project. Hence, the teamwork project, P2 is updated in the repository of teamwork server.

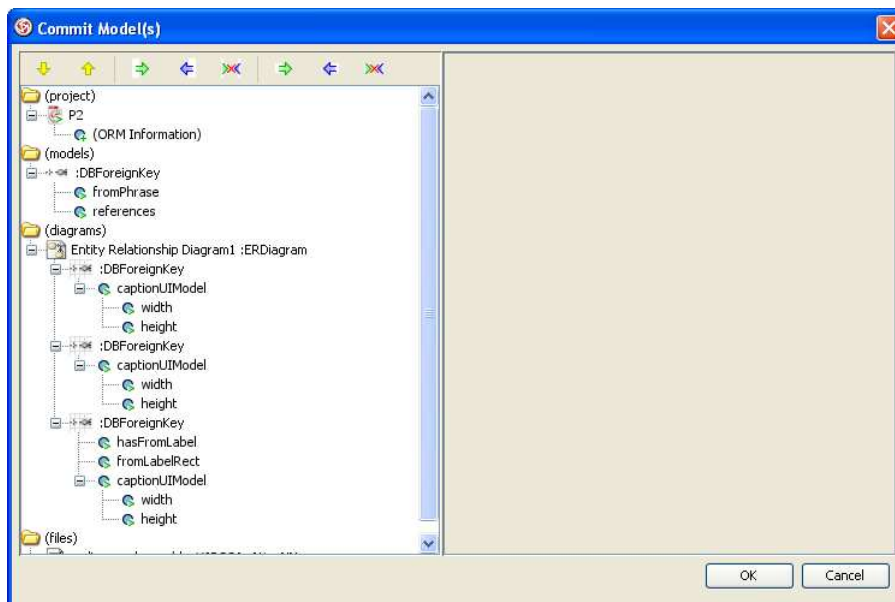


Figure 6.10 - Commit Model(s) dialog

Meanwhile, team member **dbva_user**, is going to modify the relationship between **PurchaseOrder_Product** and **PurchaseOrder** entities.

1. Right-click on the connection line between **PurchaseOrder_Product** and **PurchaseOrder** entities, select **Open Specification...** from the popup menu. The **Relationship Specification** dialog box is displayed.
2. Enter "**contains**" for the **Phrase** on **PurchaseOrder** and click **OK**.

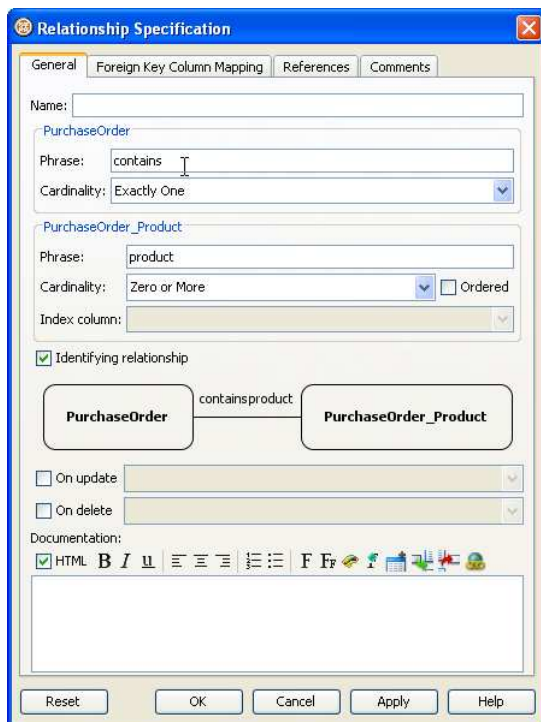


Figure 6.11 - Relationship specification dialog

3. Commit project by entering the description as "**Phrase for PurchaseOrder is set as "contains".**" and click **OK**. The **Commit Model(s)** dialog box will be displayed.

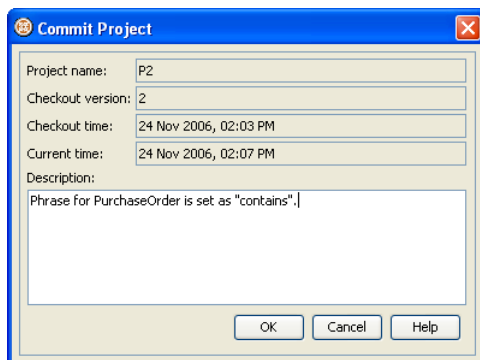


Figure 6.12 - Commit Project dialog

4. Since team members, **vpuml_user** and **dbva_user** work on the same property, the phrase of the PurchaseOrder entity at the same time, and **dbva_user** committed project after **vpuml_user** committed project, a conflict was detected by the Teamwork Server.

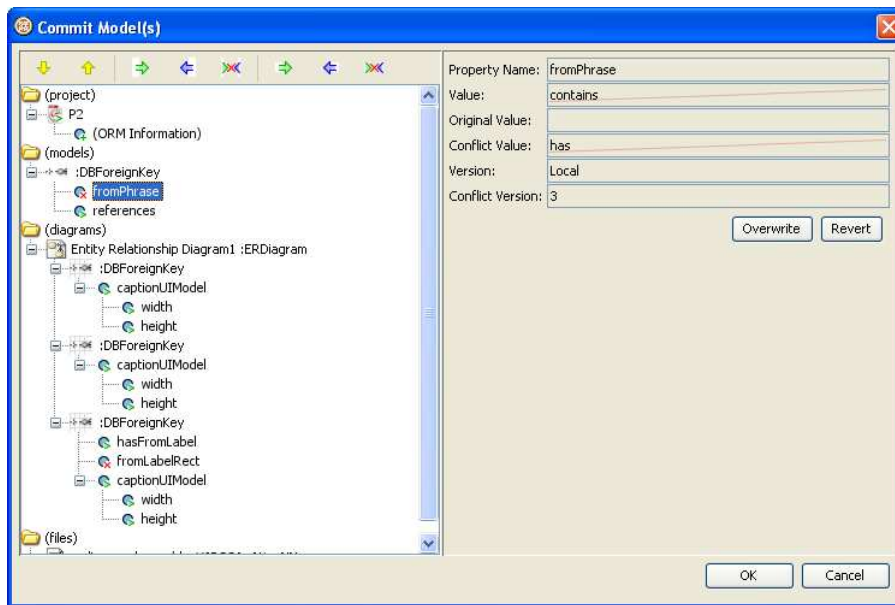


Figure 6.13 - Commit Model(s) dialog

5. Click on the **Revert** button to apply the changes made by another team member, **vpuml_user**.

7

Checking for Update

Chapter 7 - Checking for Update

Checking update in VP-UML

As **vpuml_user** is now working on teamwork project P2, **vpuml_user** would like to generate the class diagram from the entity relationship diagram

1. On the entity relationship diagram, hold down the right mouse button, move the mouse from right to left to form the gesture. A blue path will be shown indicating the gesture.

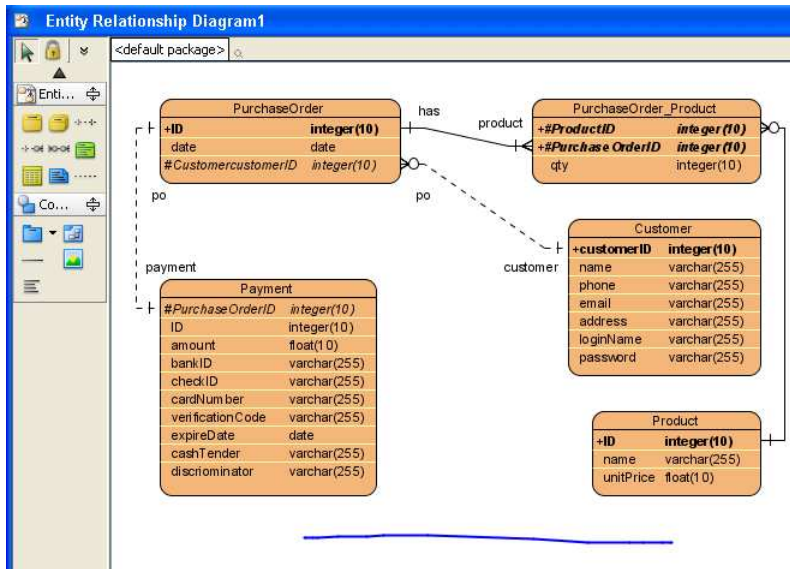


Figure 7.1 - Generate Class Diagram by using Gesture

2. A class diagram is generated as shown below.

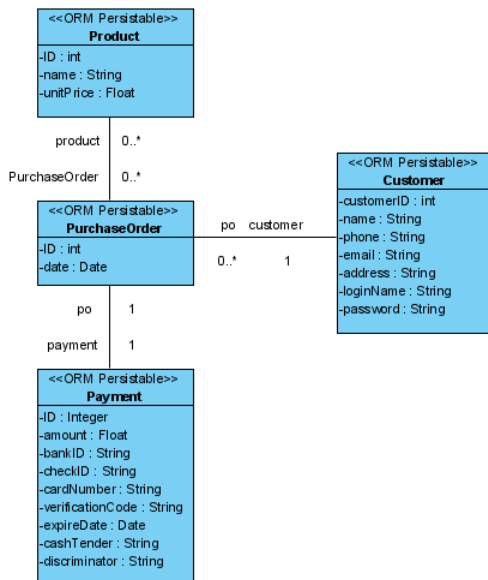


Figure 7.2 - The generated Class Diagram

Team member, **vpuml_user** is going to commit changes for generating the class diagram.

1. Commit project by entering the description as "**ERD is synchronized to class diagram.**", and click **OK**. The **Commit Model(s)** dialog box will be displayed.

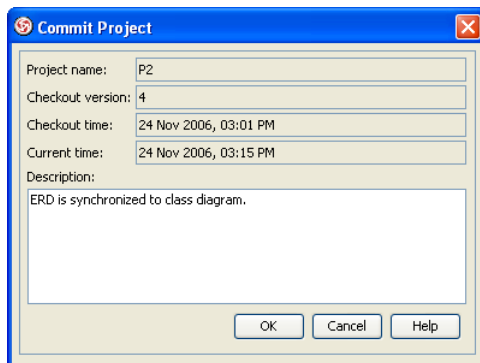


Figure 7.3 - Commit Project dialog

2. Click **OK** to proceed with committing the project. Hence, the teamwork project, P2 is updated in the repository of teamwork server.

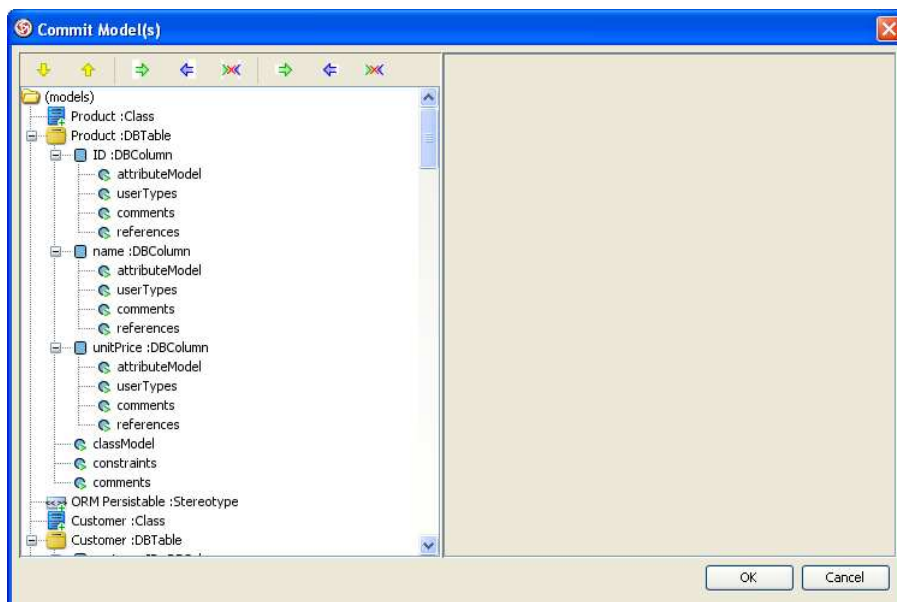


Figure 7.4 - Commit Model(s) dialog

Team member, **dbva_user** is going to check if there is a newer version of the project available on the Teamwork Server.

1. Open the **Teamwork Client**.
2. Click the **Check for Update** button on the toolbar.

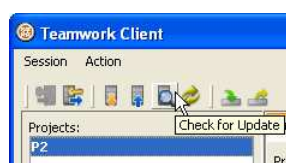


Figure 7.5 - Check for Update button

3. Since there is a newer version of the project P2 committed by another team member, **vpuml_user**, the version number of the latest version is shown on the status bar.

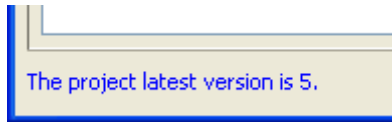


Figure 7.6 - Status bar

4. By knowing that the working copy is not in the latest version, **dbva_user** decided to perform an update to grab the changes made by others. Select **Action > Update Project** to update the project to the latest version. The project opened in DB-VA will be refreshed automatically, which can be indicated by one class diagram added to the **Diagram Navigator**.

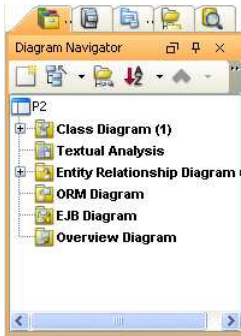


Figure 7.7 - Diagram Navigator

8

Importing and Exporting Project

Chapter 8 - Importing and Exporting Project

Exporting Project

Now, team member, **dbva_user** is going to export the Project, P2 and work on the entity relationship diagram without connecting to the Teamwork Server.

1. Open the **Teamwork Client**.
2. Click the **Export Teamwork Project** button on the toolbar.



Figure 8.1 - Export Teamwork Project button

The **Export Teamwork Project** dialog box is displayed.

3. Specify the location and file name for the exported project, click **Save** to export project.
4. Click the **Logout** button on the toolbar to disconnect to the Teamwork Server.

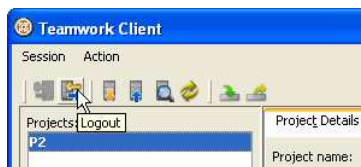


Figure 8.2 - Logout button

Modifying the Exported Project

After exporting the project, **dbva_user** is allowed to modify the project without connecting to the Teamwork Server.

1. Open the exported project. Reposition the shapes of Payment and Customer Entity.

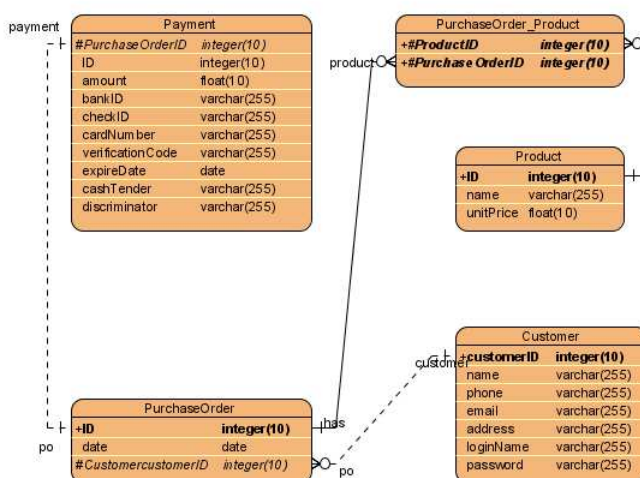


Figure 8.3 - Entity Relationship Diagram

2. Save the project.

Importing the Exported Project

After modifying the exported project by repositioning the shapes of entities, **dbva_user** is going to import the modified project to apply changes to the teamwork project.

1. Open the **Teamwork Client**.
2. Click the **Import Teamwork Project** button on the toolbar.

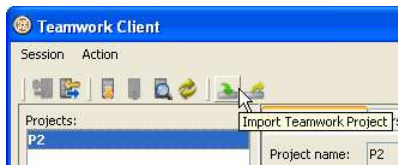


Figure 8.4 - Import teamwork Project button

The **Import Teamwork Project** dialog box is displayed.

3. Select the project to be imported, click **Open** to import project. The **Commit Project** dialog box is displayed.
4. Enter the description as "**Import project with repositioned shapes in ERD.**", and click **OK**. The **Commit Model(s)** dialog box will be displayed.

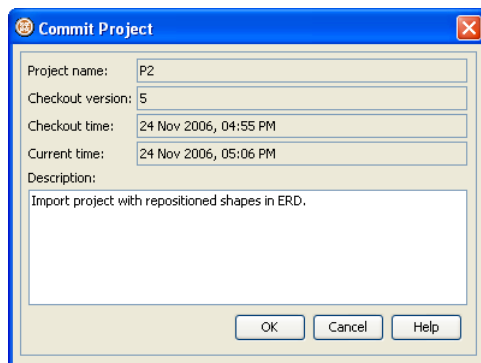


Figure 8.5 - Commit Project dialog

5. Click **OK** to proceed with committing the project. Hence, the teamwork project, P2 is updated in the repository of teamwork server.

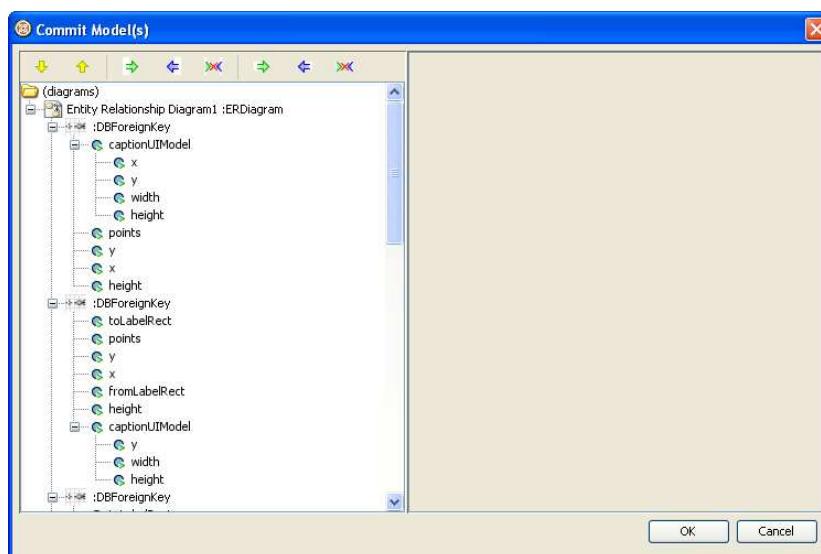


Figure 8.6 - Commit Model(s) dialog

The imported project will be checked-in to the Teamwork Server automatically.

9

Versioning

Chapter 9 - Versioning

At the end of the tutorial, all team members can check the version history of the project.

To view the version history:

1. Open the **Teamwork Client**.
2. Select the **Versions** tab next to the **Project Details** tab.

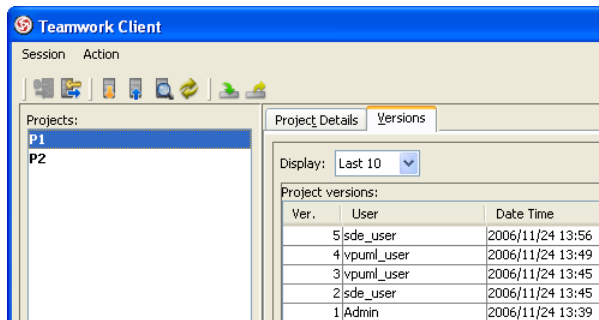



Figure 9.1 - The version tab

The corresponding version history will be displayed according to the project selected from the **Projects** list.

To view the detailed information for a particular version of the project:

1. Click a particular version on **Project versions** table.
2. Click **Open Project** button to view the project for the particular version.

 Modification made to the project opened by Open Project button on the Versions tab is not allowed to commit as a new change to the Teamwork Server.

For team member, **vpuml_user**, the version history for Project P1 is shown below.

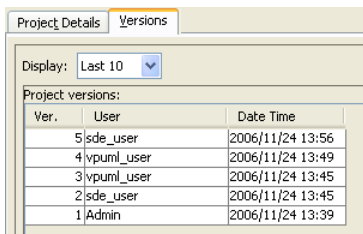


Figure 9.2 - The version history of P1

Detailed information for version 2 of Project P1 is shown:

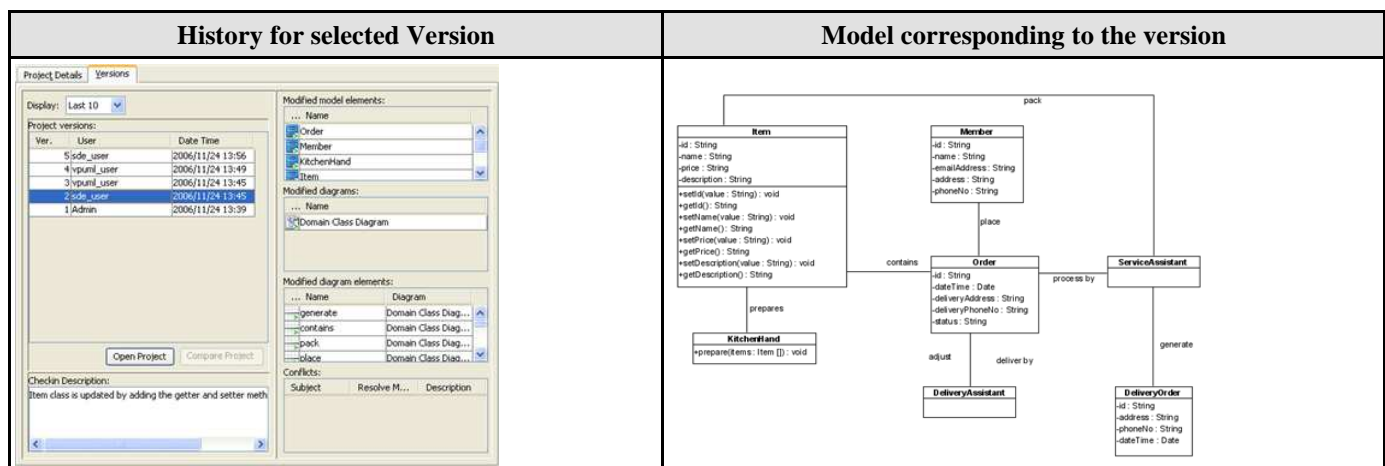


Table 9.1

For team member, **vpuml_user**, the version history for Project P2 is shown below.

| Ver. | User | Date Time |
|------|------------|------------------|
| 6 | dbva_user | 2006/11/24 17:09 |
| 5 | vpuml_user | 2006/11/24 15:16 |
| 4 | dbva_user | 2006/11/24 14:07 |
| 3 | vpuml_user | 2006/11/24 14:06 |
| 2 | dbva_user | 2006/11/24 14:03 |
| 1 | Admin | 2006/11/24 14:01 |

Figure 9.3 - The version history of P2

Detailed information for version 2 of Project P2 is shown:

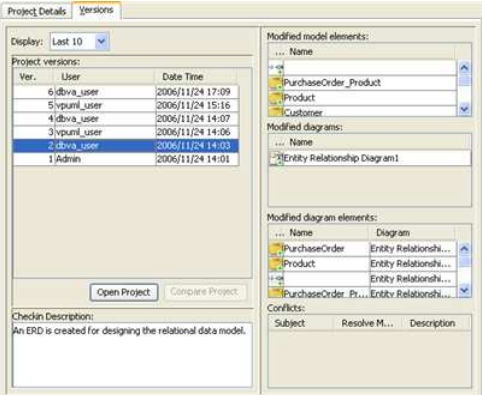
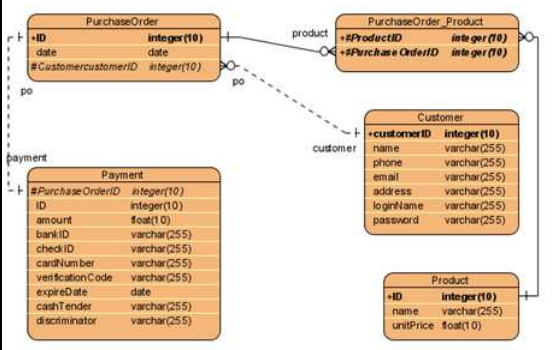
| History for selected Version | Model corresponding to the version |
|---|--|
|  |  |

Table 9.2

For team member, **sde_user**, the detailed information for version 4 of Project P1 is shown:

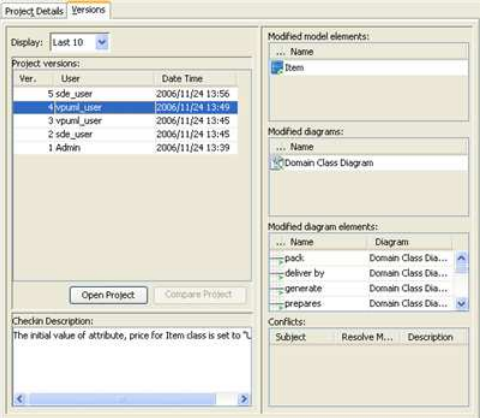
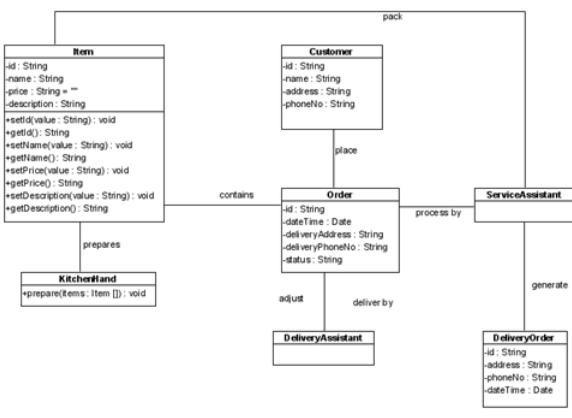
| History for selected Version | Model corresponding to the version |
|---|--|
|  |  |

Table 9.3

For team member, **dbva_user**, the detailed information for version 2 of Project P2 is shown:

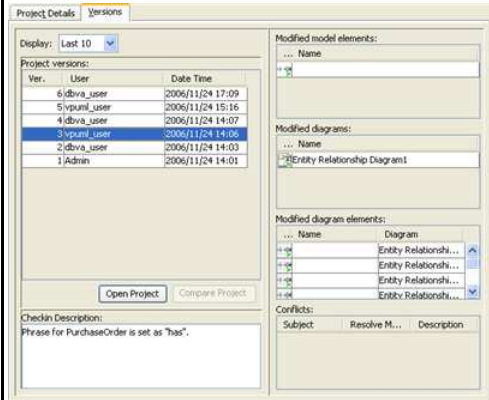
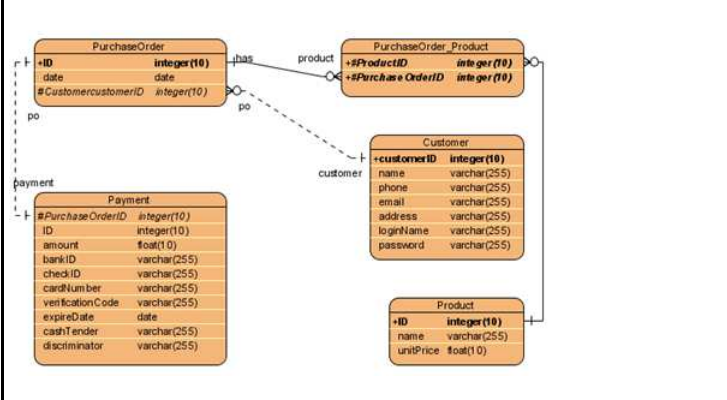
| History for selected Version | Model corresponding to the version | | | | | | | | | | | | | | | | | | | | | |
|--|------------------------------------|------------------|-----------|---|-----------|------------------|---|------------|------------------|---|-----------|------------------|---|------------|------------------|---|-----------|------------------|---|-------|------------------|--|
|  <p>Project Details Versions</p> <p>Display: Last 10</p> <p>Project versions:</p> <table border="1"> <thead> <tr> <th>Ver.</th> <th>User</th> <th>Date Time</th> </tr> </thead> <tbody> <tr> <td>6</td> <td>dbva_user</td> <td>2006/11/24 17:09</td> </tr> <tr> <td>5</td> <td>vpuni_user</td> <td>2006/11/24 15:16</td> </tr> <tr> <td>4</td> <td>dbva_user</td> <td>2006/11/24 14:07</td> </tr> <tr> <td>3</td> <td>vpuni_user</td> <td>2006/11/24 14:03</td> </tr> <tr> <td>2</td> <td>dbva_user</td> <td>2006/11/24 14:03</td> </tr> <tr> <td>1</td> <td>Admin</td> <td>2006/11/24 14:01</td> </tr> </tbody> </table> <p>Modified model elements:</p> <p>Modified diagrams:</p> <p>Entity Relationship Diagram</p> <p>Modified diagram elements:</p> <p>Conflicts:</p> <p>Checkin Description: Phrase for PurchaseOrder is set as "has".</p> | Ver. | User | Date Time | 6 | dbva_user | 2006/11/24 17:09 | 5 | vpuni_user | 2006/11/24 15:16 | 4 | dbva_user | 2006/11/24 14:07 | 3 | vpuni_user | 2006/11/24 14:03 | 2 | dbva_user | 2006/11/24 14:03 | 1 | Admin | 2006/11/24 14:01 |  <p>PurchaseOrder</p> <ul style="list-style-type: none"> #ID integer(10) date #CustomercustomerID integer(10) <p>PurchaseOrder_Product</p> <ul style="list-style-type: none"> #ProductID integer(10) #Purchase OrderID integer(10) <p>Customer</p> <ul style="list-style-type: none"> #customerID integer(10) name varchar(255) phone varchar(255) email varchar(255) address varchar(255) loginName varchar(255) password varchar(255) <p>Payment</p> <ul style="list-style-type: none"> #Purchase OrderID integer(10) ID integer(10) amount float(10) bankID varchar(255) checkID varchar(255) cardNumber varchar(255) verificationCode varchar(255) expireDate date cashTender varchar(255) discriminator varchar(255) <p>Product</p> <ul style="list-style-type: none"> #ID integer(10) name varchar(255) unitPrice float(10) <p>Relationships:</p> <ul style="list-style-type: none"> PurchaseOrder has PurchaseOrder_Product (has) PurchaseOrder has Payment (po) PurchaseOrder has Customer (po) PurchaseOrder_Product has Product (product) |
| Ver. | User | Date Time | | | | | | | | | | | | | | | | | | | | |
| 6 | dbva_user | 2006/11/24 17:09 | | | | | | | | | | | | | | | | | | | | |
| 5 | vpuni_user | 2006/11/24 15:16 | | | | | | | | | | | | | | | | | | | | |
| 4 | dbva_user | 2006/11/24 14:07 | | | | | | | | | | | | | | | | | | | | |
| 3 | vpuni_user | 2006/11/24 14:03 | | | | | | | | | | | | | | | | | | | | |
| 2 | dbva_user | 2006/11/24 14:03 | | | | | | | | | | | | | | | | | | | | |
| 1 | Admin | 2006/11/24 14:01 | | | | | | | | | | | | | | | | | | | | |

Table 9.4