

# Visual Paradigm

## Quick Start



Last update: Apr 23, 2015

© Copyright 2002-2015 Visual Paradigm International Ltd.

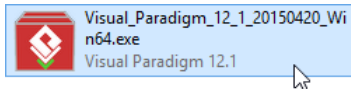
# Table of Contents

<b>Table of Contents .....</b>	<b>2</b>
<b>Getting Started.....</b>	<b>3</b>
Installing Visual Paradigm .....	3
Starting Visual Paradigm .....	3
The Environment .....	5
Saving and Opening Project .....	5
<b>Basic Diagramming Techniques.....</b>	<b>6</b>
Creating Diagram .....	6
Creating and Connecting Shapes .....	6
Resizing Shape.....	7
Adding Control Points to Connector.....	7
Describing Model Element .....	7
Editing Shape Color.....	7
<b>UML Modeling.....</b>	<b>8</b>
Use Case Diagram .....	8
Sequence Diagram .....	8
Activity Diagram.....	9
Class Diagram.....	10
<b>Business Process Modeling (with BPMN).....</b>	<b>12</b>
Business Process Diagram (BPD).....	12
Working with Sub-Process .....	12
Documenting Working Procedure .....	13
Animating Business Process.....	13
<b>Requirements Gathering.....</b>	<b>14</b>
Identifying Use Cases with Use Case Statements .....	14
Writing User Story .....	15
Sprint Management .....	16
Detailing User Story .....	16
Writing User Story Scenario .....	17
Scenario-based wireframing .....	18
Producing Requirement Specification .....	20
<b>Code Engineering .....</b>	<b>21</b>
Java Round-Trip.....	21
C++ Round-Trip .....	21
Instant Reverse .....	21
Instant Generator .....	21
Reverse Engineer Sequence Diagram from Java .....	22
<b>Documentation .....</b>	<b>23</b>
Doc. Composer – Build from Scratch.....	23
Doc. Composer – Fill-in Doc .....	24
Project Publisher .....	25
<b>Collaborative Modeling .....</b>	<b>26</b>
Subscribing to VPository .....	26
Importing a Project.....	27
Committing.....	28
Updating .....	28
Sharing Your Design with PostMania .....	28
<b>Advanced Modeling .....</b>	<b>31</b>
Nickname.....	31
Project Reference .....	31
<b>Impact Analysis .....</b>	<b>33</b>

## Getting Started

### Installing Visual Paradigm

1. Execute the Visual Paradigm installer.



2. Click **Next** to proceed to the **License Agreement** page.
3. Read through the license agreement. If you fully understand and agree with them, choose **I accept the agreement** to accept the terms. Click **Next**.
4. Specify the directory for installing Visual Paradigm. Click **Next** to continue.
5. Specify the name of the **Start Menu** folder that will be used to store the shortcuts. Keep **Create shortcuts for all users** checked if you want the shortcut(s) to be available in all the user accounts in the machine. Click **Next** to continue.
6. In the **Select File Associations** page, keep **Visual Paradigm Project (\*.vpp)** checked if you want your system to be able to open the project file upon direct execution (i.e. double click). Click **Next** to start the file copying process.
7. Once the file copying is finished, you can choose to start Visual Paradigm immediately. On the other hand, you can finish the installation without starting Visual Paradigm. Choose the option **Don't Start** and click **Finish**. This ends the installation of Visual Paradigm.

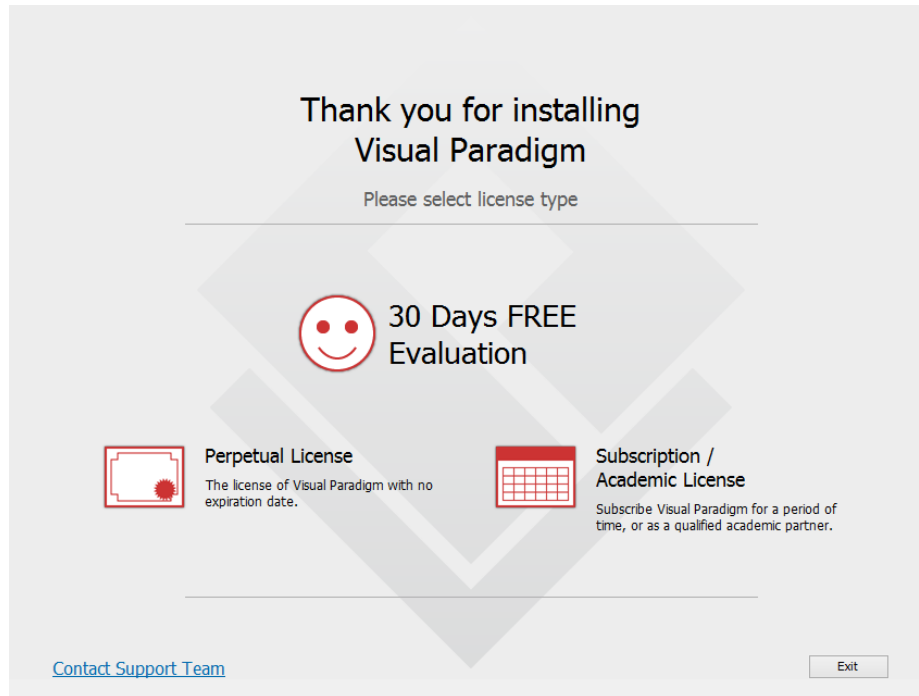
### Starting Visual Paradigm



Windows users can start Visual Paradigm via the Start screen (for Windows 8/8.1) or Start menu (for Windows 7 or earlier). If you selected not to create an entry in the Start menu (during the installation), you can look under the installation folder of Visual Paradigm (the same path specified in step 4 in the section above) and start Visual Paradigm by running **Visual Paradigm.exe** in the **bin** folder.

### Selecting License Type

When you start Visual Paradigm the first time, you are asked to select a way to activate Visual Paradigm.



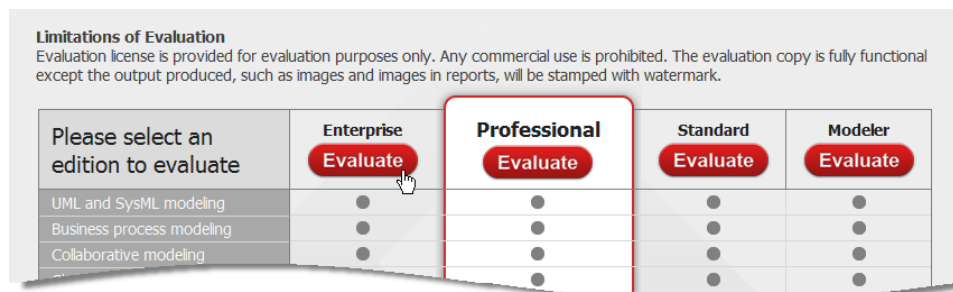
Depending on whether you own a purchased copy or an evaluation copy of Visual Paradigm, you can proceed by following the steps below respectively:

*For Customers*

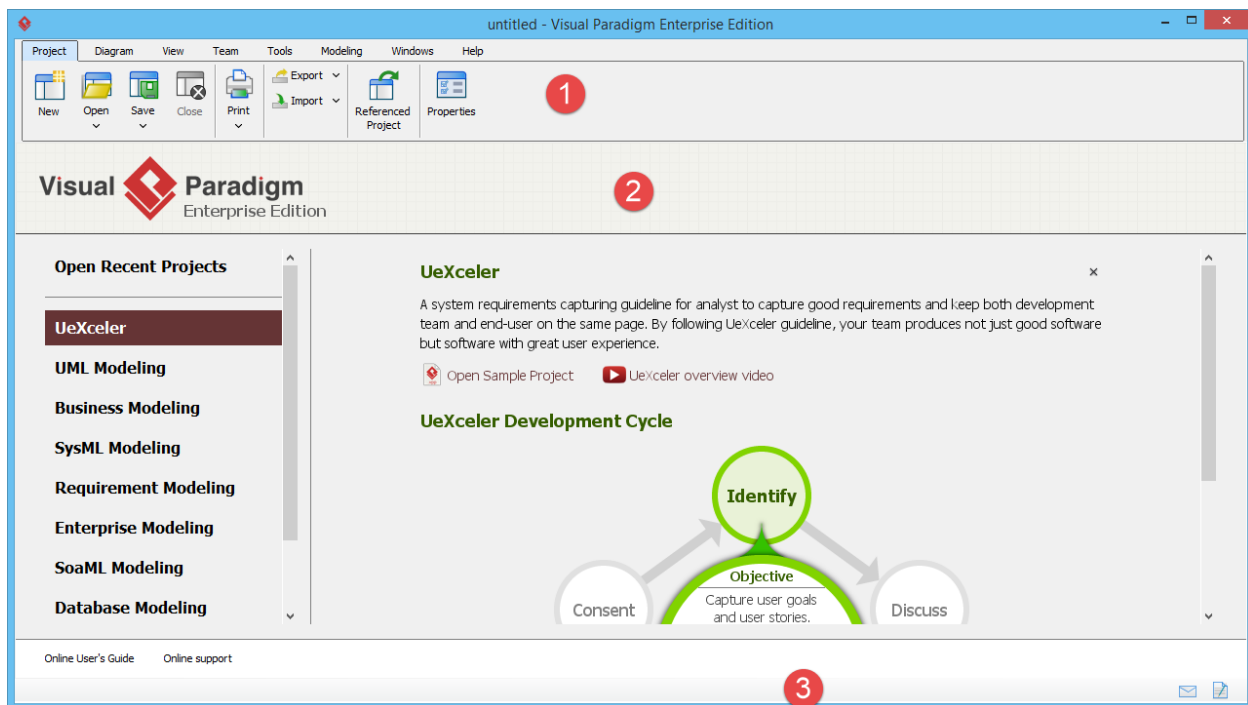
You should receive a notification Email with an activation code. The same activation code can also be found from the license key listed in your customer account. Copy the activation code first, click on **Perpetual License**, paste the copied activation code and then click **Activate** to continue.

*For Evaluators*

If you want to evaluate Visual Paradigm, click **30 Days FREE Evaluation**. You will then be asked to select the edition of product to evaluate. Visual Paradigm features vary by product edition. For more details on the features supported by different editions, check the **Edition Comparison** page. Click on the **Evaluate** button to confirm your edition selection. Then, you can start your 30 days evaluation.



## The Environment



No.	Name	Description
1	Toolbar	A tabbed toolbar that allows you to perform various operations in Visual Paradigm.
2	Diagram editor	The diagram will be displayed in diagram editor.
3	Status bar	Notifications are shown here. You can also open the message pane and description pane from the bottom right of the status bar.

## Saving and Opening Project

To save your work, select either **Project > Save** or **Project > Save as....** When you are saving a project for the first time, you will be asked to specify its location.

To open an existing project, select **Project > Open** from the toolbar and select the project to open.

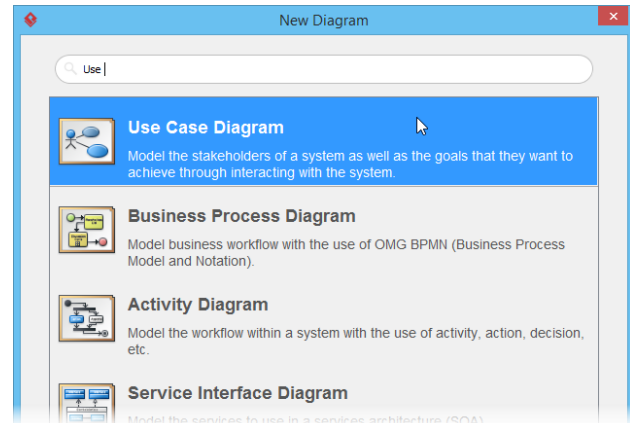
## Basic Diagramming Techniques

This section will go through the steps of creating diagrams, creating shapes and connecting them. You will also learn how to document model elements and make diagrams more readable by coloring shapes.

### Creating Diagram

Let's take use case diagram as an example. To create a use case diagram:

1. Select **Diagram > New** from the toolbar.
2. At the top of the **New Diagram** window, enter the diagram type (or a part of it).
3. Select the type of diagram to create.
4. Click **Next**.
5. Enter the diagram name, its location and description.
6. Click **OK**.

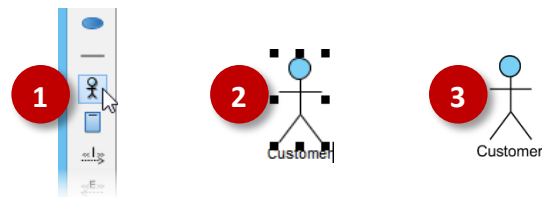


### Creating and Connecting Shapes

#### Using the diagram toolbar

Let's create an actor from the diagram toolbar for now.

1. Click on **Actor** in the diagram toolbar.
2. Click on the diagram to create an actor, and enter its name.
3. Click on the diagram or press **Enter** to confirm.

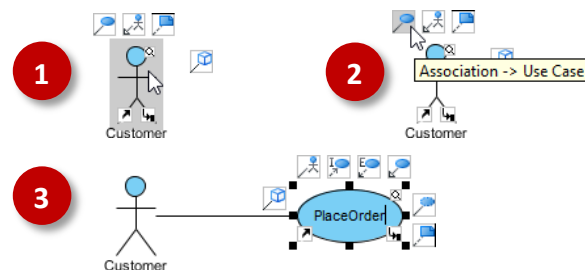


#### Using the Resource-Centric Interface

If you move your mouse pointer over a shape, you will see a number of icons surrounding it. Those are known as the resource icons. They together formed the resource-centric interface.

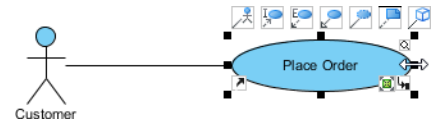
Resource-centric interface allows you to create a new shape that connects with an existing one. You can also use the resource-centric interface to create connector between two shapes. Let's create a use case from actor.

1. Move your mouse pointer over the actor shape.
2. Press on the resource icon **Association -> Use Case** and drag to the right.
3. Release the mouse button to create the use case, and enter its name. This will create a use case that associates with the actor.



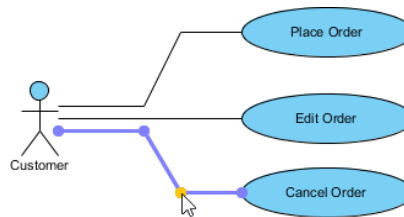
## Resizing Shape

When you click on a shape, you will see several resize handlers appear around the shape. You can drag on these handlers to enlarge or diminish the shape.



## Adding Control Points to Connector

For most diagram types, 'oblique connector' is chosen as the default connector type, meaning that when you draw a connector, it will be an oblique connector. To route such connector, you can add control points to it. To add control points, simply drag on the connector to create a point, and then keep dragging to adjust its position.



There are totally five connector types. If you want to apply another connector type on a connector, right click on that connector and select **Styles and Formatting > Connector Styles > [TYPE]** from the popup menu. If you want to update all the connectors in a diagram, right click on the background of diagram and select **Connectors > [TYPE]** from the popup menu.

## Describing Model Element

Generally speaking, naming model elements alone would not suffice to describe their details. To elaborate, you can enter additional details in the **Description Pane**. Simply go to the diagram and select the shape you need to describe. Click on the **Show Description** button at the bottom right of the status bar to open the **Description Pane** and fill in the details.

In addition to textual description, voice recording is also available. If your machine supports microphone usage, click the **Record** button at the top right of the **Description Pane**. In the **Record Voice** window, click the red circle button to start recording. To stop, click the button with a square inside. To save your recording, click **OK**.

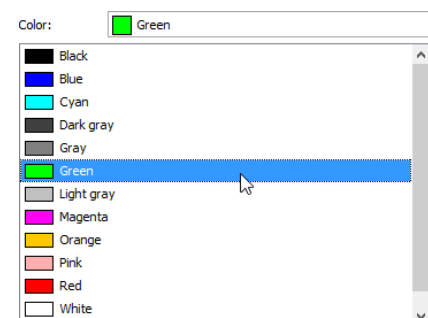


Make sure your recording device is available when applying this feature.

## Editing Shape Color

Make your diagram more expressive by formatting shapes based on their contexts. Let's change the color of a use case shape.

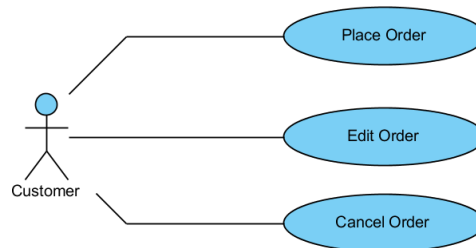
1. Right click on the use case shape and select **Styles and Formatting > Formats...** from the popup menu.
2. Open the **Background** tab in the **Formats** window. Select **Green** for color. Click **OK** to confirm the change.



## UML Modeling

### Use Case Diagram

A use case diagram is used to model and identify the functional requirements of a software system. In a use case diagram, stakeholders and user goals are represented by actor and use cases respectively.



An actor is any person or external system that interacts with the system to achieve a user goal (i.e. use case). The use case diagram above illustrates the use cases of an order processing system. Customer, an actor, interacts with the system to accomplish the goal of order placement, as modeled by the use case *Place Order*. There are other goals that the customer wants to accomplish, such as *Edit Order* and *Cancel Order*. You can apply the diagramming techniques described in the previous section to create such diagram.

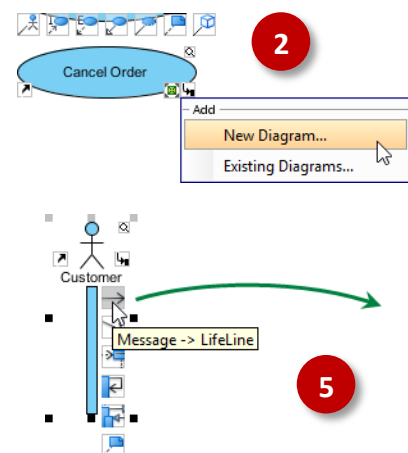
### Sequence Diagram

A sequence diagram is primarily used to show the interaction between objects that are represented as lifelines in a sequential order. More importantly, lifelines show all of their interaction points with other objects in events.

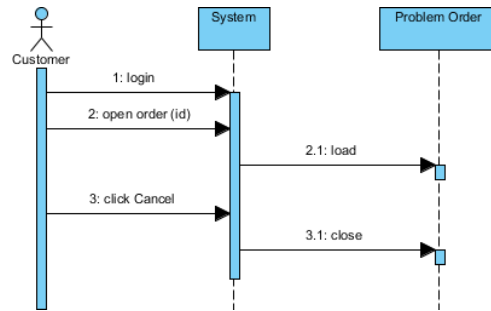
As always, you can create a new diagram via the toolbar, same for sequence diagram. To create sequence diagram, choose **Diagram > New** from the toolbar. Then, select **UML diagrams > Sequence Diagram** in the **New Diagram** window. Here, let's create a sequence diagram with an alternative approach - create as a sub-diagram of use case.

The reason of creating a sequence diagram as a sub-diagram of use case is to visualize the interaction between user and system in accomplishing the use case. Let's see how it works. We will carry on with the use case diagram created in the previous section.

1. Move your mouse pointer over the use case *Cancel Order*.
2. Click on the tiny **Sub Diagrams** button at the bottom right of the shape and select **New Diagram...** from the popup menu.
3. This shows the **New Diagram** window. Select **Sequence Diagram** and then click **Next**.
4. Click **OK**.
5. This creates a blank sequence diagram. Create an actor *Customer* in the diagram. You can find *Actor* from the diagram toolbar.
6. Let's model the interaction of how customer can cancel an order. Move the mouse pointer over the actor. Press on the **Message -> LifeLine** resource and drag it out.
7. Release the mouse button. Name the lifeline *System*, and the message *login*.
8. Complete the interaction by adding other messages and the *Order* lifeline.





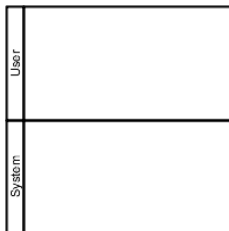


To go back to the parent use case, you can click on the shortcut on top of the diagram:

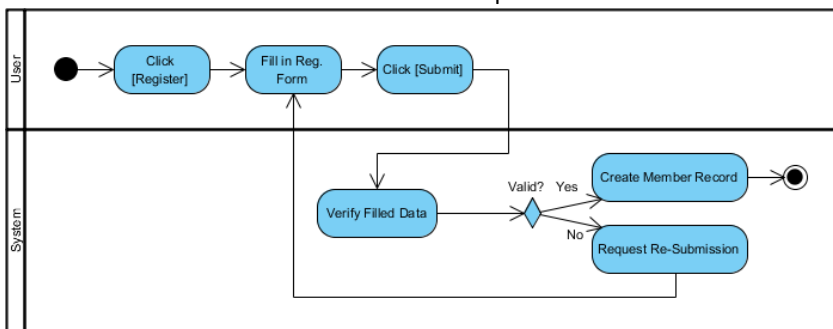
## Activity Diagram

An activity diagram is essentially a flowchart, showing the flow of control from one activity to another. Unlike a traditional flowchart, it can model the dynamic aspects of a system because it involves modeling the sequential steps in a computational process. Let's make use of an activity diagram to model the registration process.

1. Create an empty activity diagram named *Registration*. You can create an activity diagram by selecting **Diagram > New** from the toolbar. Then, select **Activity Diagram** in the **New Diagram** window. Click **Next** and then click **OK**.
2. Create a swimlane for grouping actions based on the participants. Select **Horizontal Swimlane** from the diagram toolbar and click on the diagram to create one.
3. Double click on the headers and rename them to *User* and *System*, respectively.



4. Create initial nodes, actions, decision node and activity final node inside the swimlane. Do remember to resize the swimlane to accommodate the shapes.

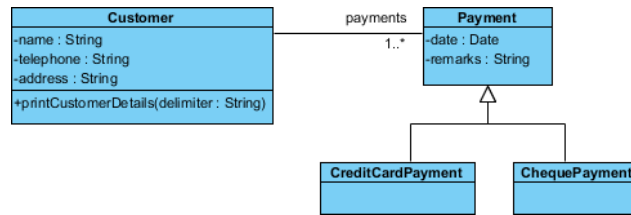


The decision node represents the moment where a decision has to be made. Multiple flows will be produced based on different results of decision making.

The **Control Flow -> Action** resource helps you create action shapes rapidly.

## Class Diagram


A class diagram models the blueprints of objects required by a system and the relationships between them. Let's make use of class diagram to model the domain classes of the order processing system.

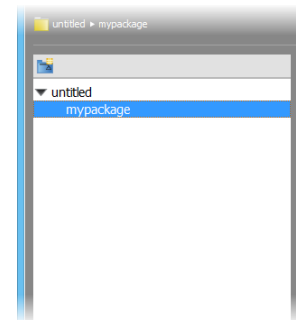


## Class Diagram and Package Header

When you create a class diagram, you are prompted to enter the package header (see the image below). Package header is a quick way to create new class diagram in a package. The diagram and the shapes created in the diagram will all be grouped by the package specified. Let's say if you want the class diagram to be grouped by a package *mypackage*, enter *mypackage* in package header.



By doing so, a new package will be created, with the new class diagram put inside. You can inspect the structuring in the **Model Structure** view, opening by clicking  at the top right of any diagram.



If the package you specified exists, the new class diagram will be put in that existing package. If the specified package does not exist, a new package will be created.

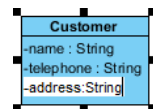


You may enter fully qualified name – for example, *com.example.mypackage*. By doing so, multiple packages will be created (if not exist), nested package hierarchy will be formed.

## Attributes and Operations

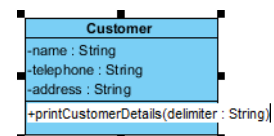
Properties of a class are represented by attributes. To add an attribute:

1. Right click on a class and select **Add > Attribute** from the popup menu.
2. Type the name and type of attribute inline - for example, *name : String*.
3. Press **Enter** to confirm.
4. You can now create the next attribute by repeating step 2. To stop, press **Esc**.



Features of a class are represented by operations. To add an operation:

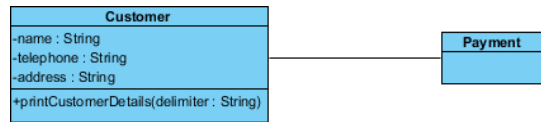
1. Right click on a class and select **Add > Operation** from the popup menu.
2. Type the name and optionally the method signature (i.e. parameters and return type) - for example, *printCustomerDetails(delimiter : String)*.
3. Press **Enter** to confirm.
4. You can now create the next operation by repeating step 2. To stop, press **Esc**.



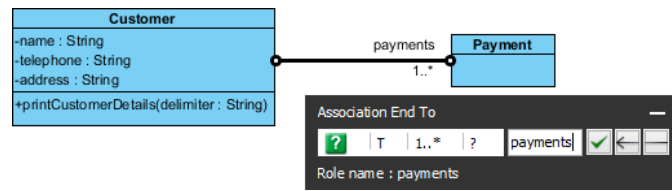
## Classes with Association

To associate two classes:

1. Move the mouse pointer over the source class.
2. Press on the **Association -> Class** resource icon and drag it out.
3. Release the mouse button to create a class with an association.
4. Name the class and press **Enter**.



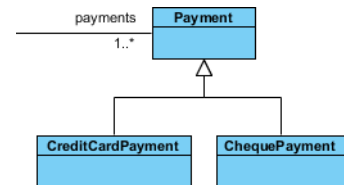
To edit an association end, double-click on the end to open the **Association Editor**. Enter a name for the role of the association in the middle text box and adjust properties like multiplicity and navigability as needed.



## Classes with Generalization

Generalization models “a-kind-of” relationship among classes. In Visual Paradigm, generalization can be created easily via the resource-centric interface, from super-class to sub-class.

1. Move the mouse pointer over the super class.
2. Press on the **Generalization -> Class** resource icon and drag it out.
3. Release the mouse button to create a class with a generalization.
4. Name the sub-class and press **Enter**.









## Business Process Modeling (with BPMN)

### Business Process Diagram (BPD)

[All Editions](#)

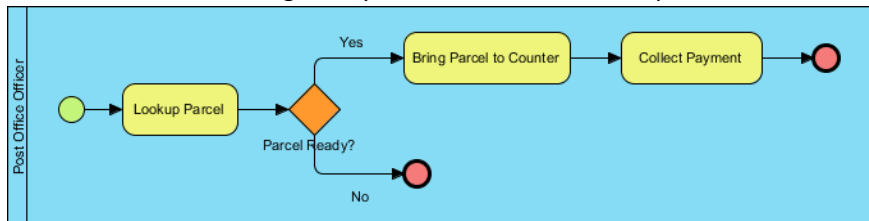
A Business Process Diagram (BPD) is used when you want to represent operational workflow. A BPD is mainly composed of activity elements such as tasks and sub-processes. They both represent work that an organization performs in a business process.

Here is a list of frequently-seen elements in a BPD:

Symbol	Name	Description
	Start event	Where the process begins and under what condition
	Intermediate event	Drive business flow based on the event it specifies. Intermediate event can be attached to an activity for modeling an event that may happen during the execution of that activity. It may also be connected by a connecting object for modeling an event that may happen after the execution of the flow element before.
	End event	Indicate where a business process completes
	Task	Atomic work which cannot be further broken down.
	Sub-process	Non-atomic, complex work that can be elaborated into smaller works
	Gateway	The diagram will be displayed in diagram pane.

Let's make use of BPD to model a parcel collection process.

1. Create an empty BPD named *Parcel Collection*. You can create a BPD by selecting **Diagram > New** from the toolbar. Then, select **Business Process Diagram** in the **New Diagram** window. Click **Next** and then click **OK**.
2. Create a pool for the post office officer, the person who execute the process. Select **Horizontal Pool** from the diagram toolbar and click on the diagram to create one. Name the pool *Post Office Officer*.
3. Create start event, tasks, gateway and end event inside the pool.

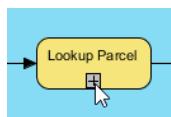


### Working with Sub-Process

#### Converting Task to Sub-Process

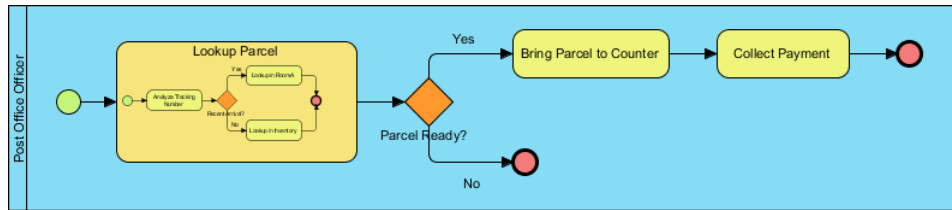
When your business expanded, you may need to revise your business process design by including more details. In such case, you may need to convert a task into a sub-process and to detail its workflow in another business process diagram. To convert a task to a sub-process, right click on the desired task and select **Convert to Sub-Process** from the popup menu.

#### Drilling-Down Sub-Process



A sub-process can be expanded to show a lower-level process. To expand a sub-process, click on the "+" symbol at the bottom of the sub-process shape. By doing this, a new BPD will be opened.

You can then model the internal details using activities, gateways, events, etc. When you go back to the parent level BPD, you will see the sub-level details appear in the sub-process shape.

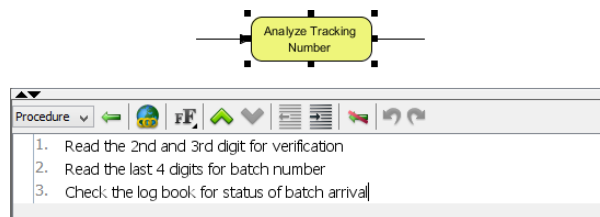


To collapse an expanded sub-process shape, click “-” at its bottom. To expand it, click “+” again.

## Documenting Working Procedure

[All Editions](#)

The necessary procedure to handle a task/sub-process can be specified by filling in the operating procedure. To specify working procedure of a task/sub-process, simply select the desired task/sub-process and enter the procedure in the **Procedure Editor** below the diagram. If you don't see the editor opened, right click on the background of diagram and select **Show Procedure Editor** from the popup menu.

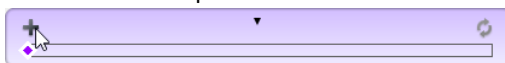


## Animating Business Process

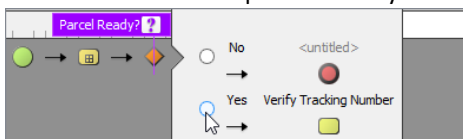
[Professional Edition +](#)

When you present your business process design to someone, like your clients or manager, having a way to visualize the workflow dynamically can help you expressing the design more effectively. Moreover, that's what the animation feature can help. The animation feature allows you to animate your design from the very first element, which is typically the start event until the final element. To animate a BPD:

1. Right click on the background of BPD and select **Show Animation Panel** from the popup menu.
2. Click “+” at the top of the **Animation Panel**.

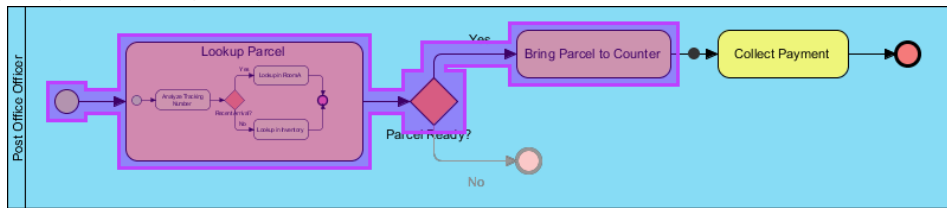


3. Enter the name of the path to animate – for example, *Collect parcel (success)*.
4. If your process design contains gateway (or any other kinds of element for decision making), you may need to construct the path manually.



5. Click ▶ to play the path. Shapes have been passed by are highlighted in purple by default. And whenever the animation is progressing through a sequence or message flow, a dark ball would appear and move

along the line to guide your line of vision.



## Requirements Gathering

Professional Edition +

Capturing the right requirements is the key in developing a system that can fully fulfill customer's needs. Visual Paradigm supports agile team in gathering requirements with use case and user story. Let's see how the requirements gathering features work.

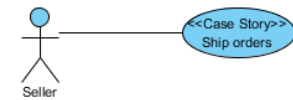
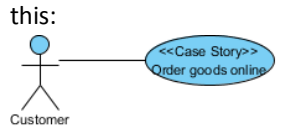
### Identifying Use Cases with Use Case Statements

A use case represents a high level function that yields a measurable result of values. The Use Case Statements tool provides an effective way in finding out use cases. It separates problem space and solution space. It also provides a user point of view in determining the problem to be addressed, keeping the whole team focused on users' real needs. Let's see how the Use Case Statement tool works.

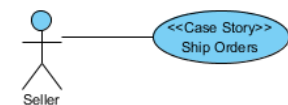
1. Create a new project first. You can create a new project by selecting **Project > New** from the toolbar. (We will go back to QuickStart.vpp later on.)
2. Select **Diagram > UeXceler** from the toolbar.
3. The **Use Case Statement** tab is opened. Let's enter two statements:
  - a. As a customer, I want to order goods online so that I can save time on shopping.
  - b. As a seller, I want to ship orders so that I can deliver goods to my customers.

4. Generate use cases from the above statements. Move your mouse over the first use case statement. At the end of the statement, click on **Create Use Case**.

5. A use case *Order goods online* is created and is visualized in a use case diagram. Go back to the **Use Case Statement** tab and create use case for the second statement. You should have a use case diagram like



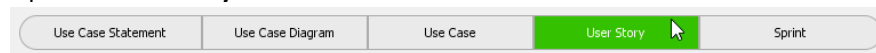
6. Rename the use cases to make them more understandable. Rename them to *Place Order* and *Ship Orders* respectively.



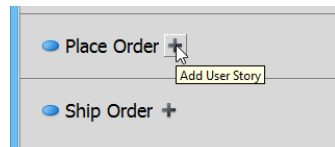
## Writing User Story

Written by user(s) (or customer team), user stories describe functionality that are needed by and valuable to the users. As an integral part of many agile development processes, user stories offer a quick way in recording user requirements without the need to write any detailed requirement documents or to have any prior consideration of system behaviors. Let's see how to write user story in Visual Paradigm.

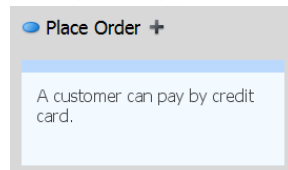
1. Open the **User Story** tab in UeXceler.



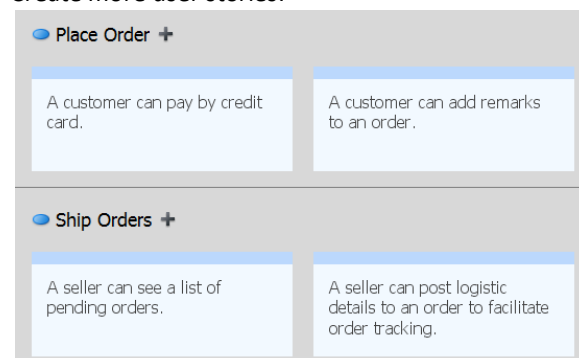
2. Let's say the system allows payment to be made by credit card. Let's write a user story about that. Click on + next to *Place Order*.



3. A story card appears. Enter *A customer can pay by credit card*. Press **Ctrl-Enter** to confirm editing.



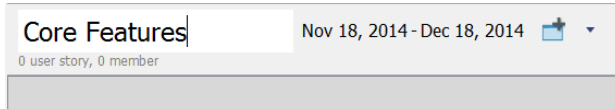
4. Create more user stories:



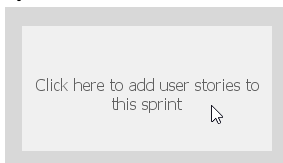
## Sprint Management

Customer team and development team work collaboratively to select the user stories to be included in the sprint with a reasonable sprint duration. Let's see how to manage sprint.

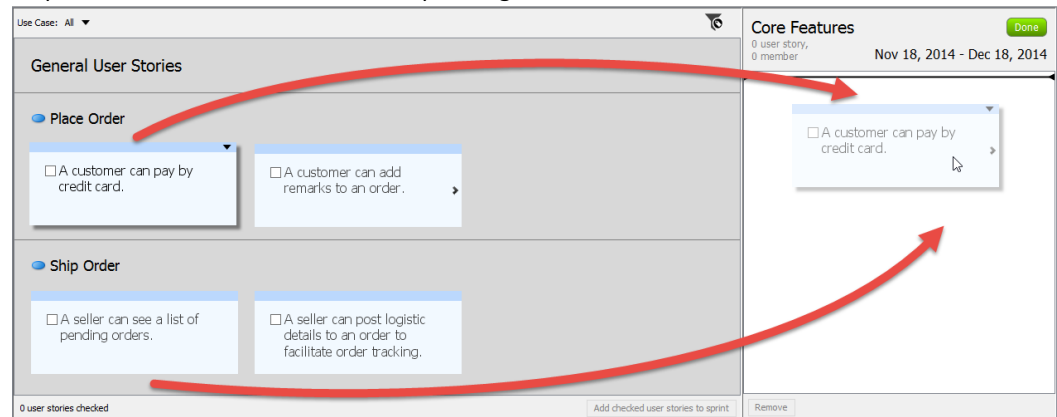
1. Open the **Sprint** tab.
2. Suppose the end user wants to have the core features available as soon as possible. Let's create a sprint that covers all those core features. At the top left corner of the screen, rename *Unnamed Sprint* to *Core Features*.



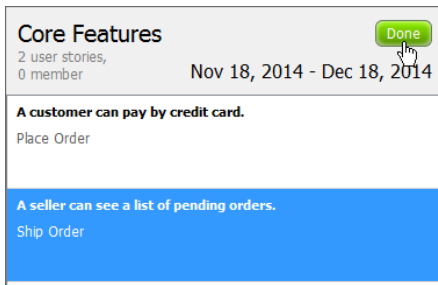
3. Next to the sprint name you can click on the date range to edit the duration. Here we just skip it first. Now, include the user stories into the sprint. Click on the caption **Click here to add user stories to this sprint**.



4. Let's say the following user stories are regarded as core features by both the user. Press on them and drag them onto the pane on the right hand side.
  - a. Place Order - A customer can pay by credit card.
  - b. Ship Orders - A seller can see a list of pending orders.



5. Click **Done**.

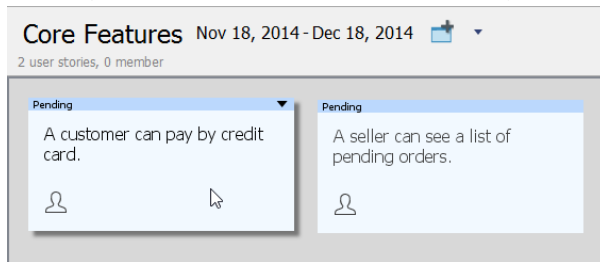


## Detailing User Story

User story is used as a reminder of conversation. During the conversation, discussions are made between the end user and the development team to find out the details of user requirements. These details can include many things like system flow, screen layout, etc. Let's see how to detail a user story.



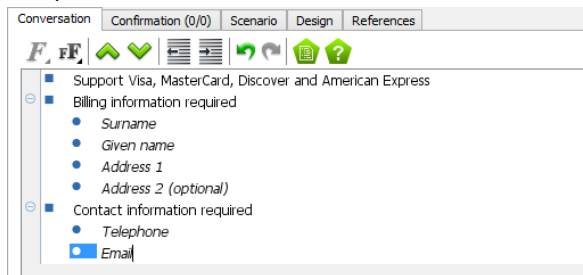
1. In the **Sprint** tab, double click on the user story *A customer can pay by credit card* to open it.



2. Suppose the following items are determined during the discussions.

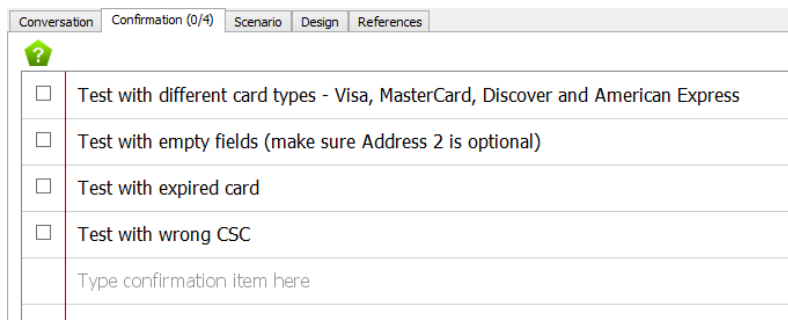
- Support Visa, MasterCard, Discover and American Express
- Billing information required
  - Surname
  - Given name
  - Address 1
  - Address 2 (optional)
- Contact information required
  - Telephone
  - Email

Enter them as conversation notes under the **Conversation** tab. You can press **Enter** to create a new item, and press **Tab** to add an indentation.



3. Open the **Confirmation** tab to note down the item to be tested during user acceptance test.

- Test different card types – Visa, MasterCard, Discover and American Express
- Test with empty fields (make sure Address 2 is optional)
- Test with expired card
- Test with wrong CSC

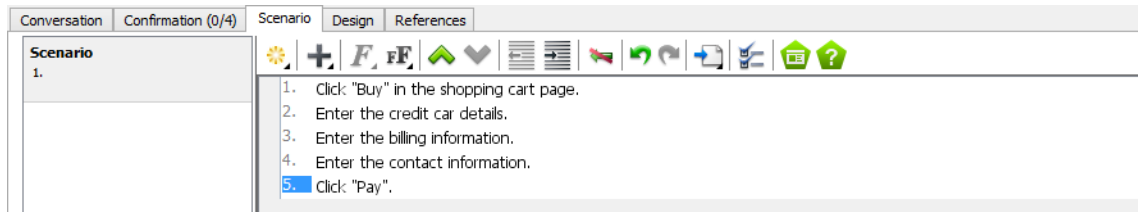


## Writing User Story Scenario

Now, you have some conversation items collected from the end user. You can make use of these notes in forming a standard user story scenario that describes the interactions between user and system.

1. Open the **Scenario** tab.
2. Enter the following steps as the scenario of this user story.

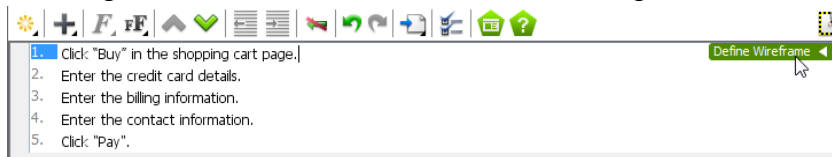
1. Click "Buy" in the shopping cart page.
2. Enter the credit card details.
3. Enter the billing information.
4. Enter the contact information.
5. Click "Pay".



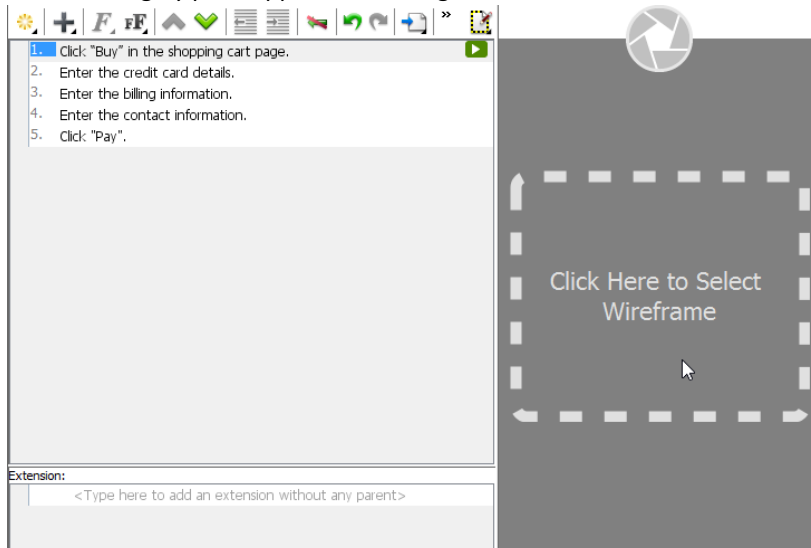
## Scenario-based wireframing

Wireframe is a sketch of the system to be developed. We use wireframe to show users how the system will look like, with minimal cost and effort. In Visual Paradigm, wireframe can be drawn and associated with scenario. Let's try.

1. Click on step 1.
2. On the right hand side, click on the button that has a triangle on it.



3. You see the gray pane appears on the right hand side? Click on it to select a kind of wireframe to create.

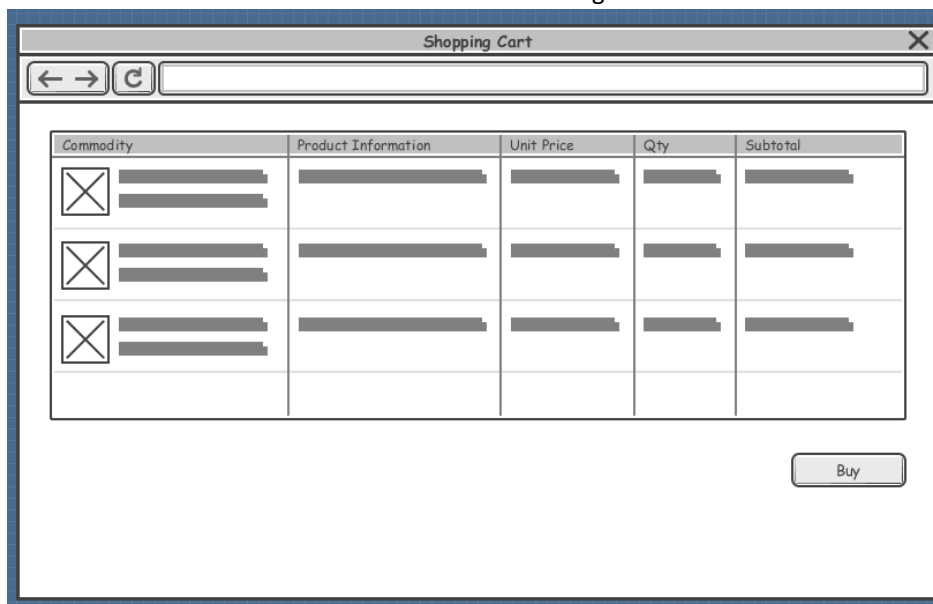


4. In the popup window, select **Website**.

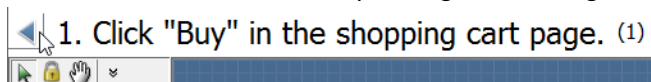
5. Click **New Website Wireframe**.



6. A new wireframe appears with an empty browser window in it. Apply the diagramming techniques described above to draw a wireframe like the following.



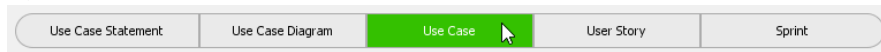
7. Go back to the scenario editor by clicking on the triangle button next to the step title.



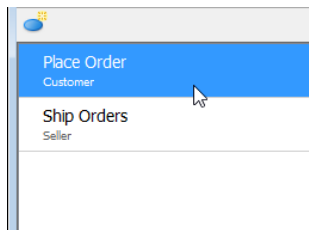
## Producing Requirement Specification

You can generate a functional specification for one use case which includes the details of its user stories such as the conversation notes, confirmation items, scenarios, wireframe, etc. The end user can then verify the specification to confirm the development plans. Let's try.

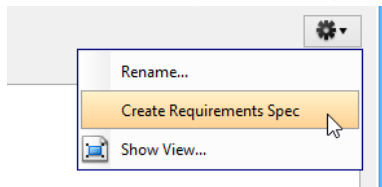
1. Open the **Use Case** tab in UeXceler.



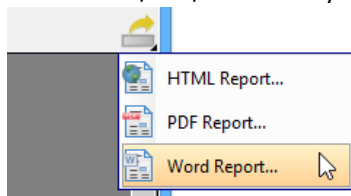
2. Select the *Place Order* use case on the left hand side.



3. On the right hand side, click on the gear button.
4. Select **Create Requirements Spec** from the drop down menu.



5. This creates a requirement specification in Doc. Composer. Now, you may edit the document, click export it into HTML/PDF/Word file by clicking on the **Export** button at the top right corner of Doc. Composer.



6. That's all for requirements gathering features. You can now open **QuickStart.vpp** and carry on with the rest of this guide.

## Code Engineering

### Java Round-Trip



Round-trip engineering enables you to produce source code from UML model (and the other way round) and keep the source code and UML model synchronized. With Java round-trip, you can generate Java source code from class model, or the reverse a Java code-base to class diagram.

To generate Java source code from UML model:

1. Select **Tools > Code > Generate Java Code...** from the toolbar.
2. Edit the output folder in the **Generate Code** window.
3. Click **OK** to generate code.

To reverse engineer class model from Java source code:

1. Select **Tools > Code > Reverse Java Code...** from the toolbar.
2. Edit the source folder in the **Reverse Code** window.
3. Click **OK** to reverse code.
4. You can then create class diagram with the reversed classes. Simply open the **Model Explorer** to find the classes and then drag them to class diagram to visualize them.

### C++ Round-Trip



Round-trip engineering enables you to produce source code from UML model (and the other way round) and keep the source code and UML model synchronized. With C++ round-trip, you can generate C++ source code from class model or the reverse, a C++ code-base to class diagram.

To generate C++ source code from UML model:

1. Select **Tools > Code > Generate C++ Code...** from the toolbar.
2. Edit the source and Cpp folder in the **Generate Code** window.
3. Click **OK** to generate code.

To reverse engineer class model from C++ source code:

1. Select **Tools > Code Engineering > Reverse C++ Code...** from the toolbar.
2. Edit the source and Cpp folder in the **Reverse Code** window.
3. Click **OK** to reverse code.
4. You can then create class diagram with the reversed classes. Simply open the **Model Explorer** to find the classes and then drag them to class diagram to visualize them.

### Instant Reverse

Instant Reverse allows you to reverse different types of source into UML class models, such as Java, C++, CORBA, Ada, PHP, Python, Objective-C, etc. To reverse, select **Tools > Code > Instant Reverse.....** from the toolbar, then select the kind of programming language to reverse. Select the source files and proceed.

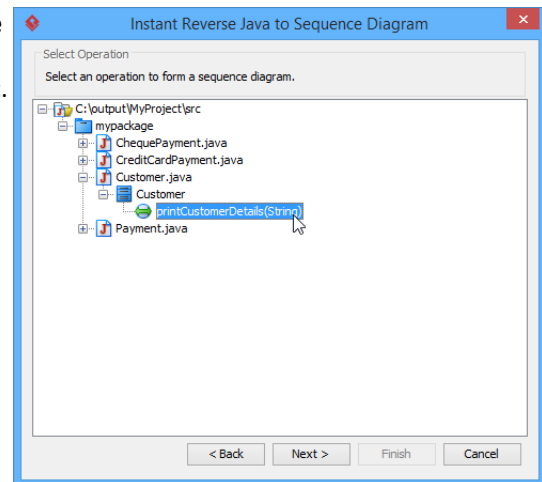
### Instant Generator

Instant generator produces source code from your model at a particular instant. Unlike the code generation support in round-trip engineering, instant generator is a one-off. To generate code, select **Tools > Code > Instant Generator...** from the toolbar, then select the programming language in which to generate.

## Reverse Engineer Sequence Diagram from Java

Sequence diagram can help represent interactions between objects in runtime. Visual Paradigm enables you to reverse your Java source code to sequence diagram. You can gain a better understanding of a piece of Java source code by reading its corresponding diagram, instead of looking at possibly a thousand lines of code. To reverse Java code to sequence diagram:

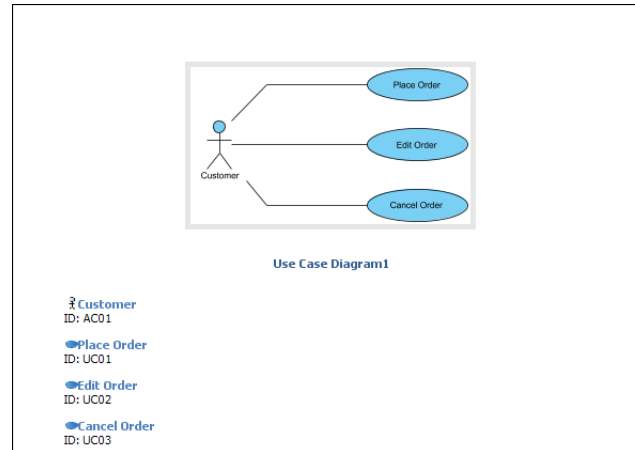
1. Select **Tools > Code > Instant Reverse Java to Sequence Diagram...** from the toolbar.
2. Add the folder that contains the source code. Click **Next**.
3. Expand the tree to select the class and its operation to be reversed. Click **Next**.
4. In the **Choose Diagram** screen, select **Create new sequence diagram**.
5. Click **Finish**



## Documentation

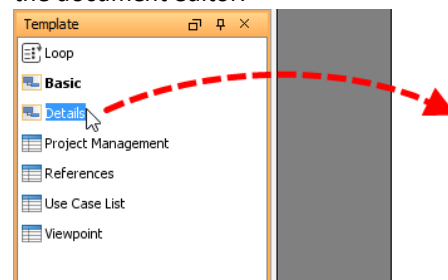
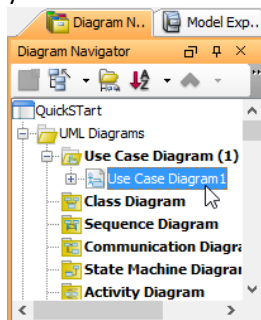
### Doc. Composer – Build from Scratch


Doc. Composer is a document creation tool. You develop a document by dragging and dropping templates to the document editor, forming a complete document. The result can then be exported to MS Word, HTML and PDF file.



To use Doc. Composer:

1. Create a new document via the toolbar (**Tools > Doc > Doc. Composer**).
2. Select **Build Doc from Scratch**.
3. Select in **Diagram Navigator**, **Model Explorer** or **Class Repository** the model element/diagram that you want to show in your document.
4. The **Template Pane** is updated to list out the available templates based on your selection. Choose the appropriate template and drag it to the document editor.

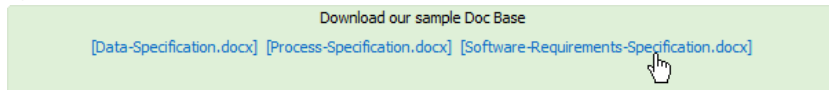


To export document as HTML/PDF/MS Word file, click on  at the top right corner of Doc. Composer.

## Doc. Composer – Fill-in Doc

Fill-in Doc is a mode of Doc. Composer that helps you integrate design details into your project documentation. To use Fill-in Doc:

1. Create a new document via the toolbar (**Tools > Doc > Doc. Composer**).
2. Select **Fill-in Doc**.
3. To make things simple, let's try with a sample provided. Click on **Software-Requirements-Specification.docx**.



4. This file is what we called a Doc Base. A Doc Base is a semi-completed version of your project documentation or report. It contains only background information, possibly filled by you or your colleague. The design details are leave empty and be filled by Doc. Composer. Open the file with Microsoft Word and take a look. You will see many variable-like fields throughout the document, like this:

```
Use Cases
${DIAGRAM, "Use Case Diagrams", "UseCaseDiagram", LoopInProject, "Basic"}
```

Those fields are known as Doc Fields. A Doc Field is a special piece of text within a Doc Base. Doc Fields will be replaced by your actual project content when being read by Doc. Composer during document exporting.

5. Exit Microsoft Word and go back to Visual Paradigm.
6. Click on ... next to the **Doc base** field.
7. Select **Software-Requirements-Specification.docx** and click **Open** to open it.
8. Doc. Composer analyzes your Doc Base and presents the Doc Fields that exist in your document. Your screen should look like this:

Doc base: C:\MyDocs\Software-Requirements-Specification.docx [Open]

**Project Name**

**Document Version**

**Use Case Diagrams**  (UseCaseDiagram)

Use Case	Description
<b>Use Cases</b> <input type="text"/> (UseCase)	[PROPERTY] description

**Requirement Diagrams**  (RequirementDiagram)

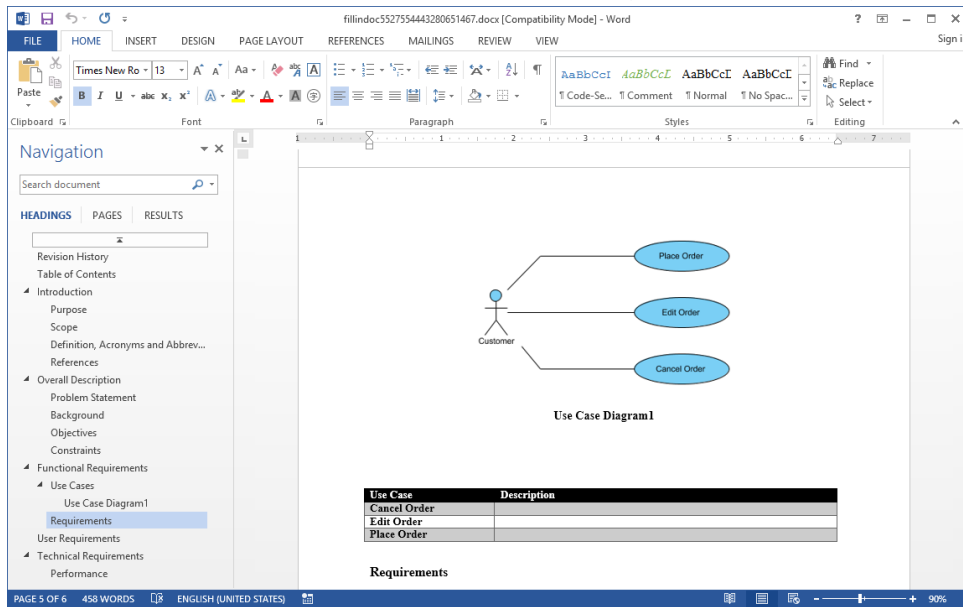
Requirement	Description
<b>Requirements</b> <input type="text"/> (Requirement)	[PROPERTY] description

**Actor**

Actors	Description
<b>Actors</b> <input type="text"/> (Actor)	[PROPERTY] description



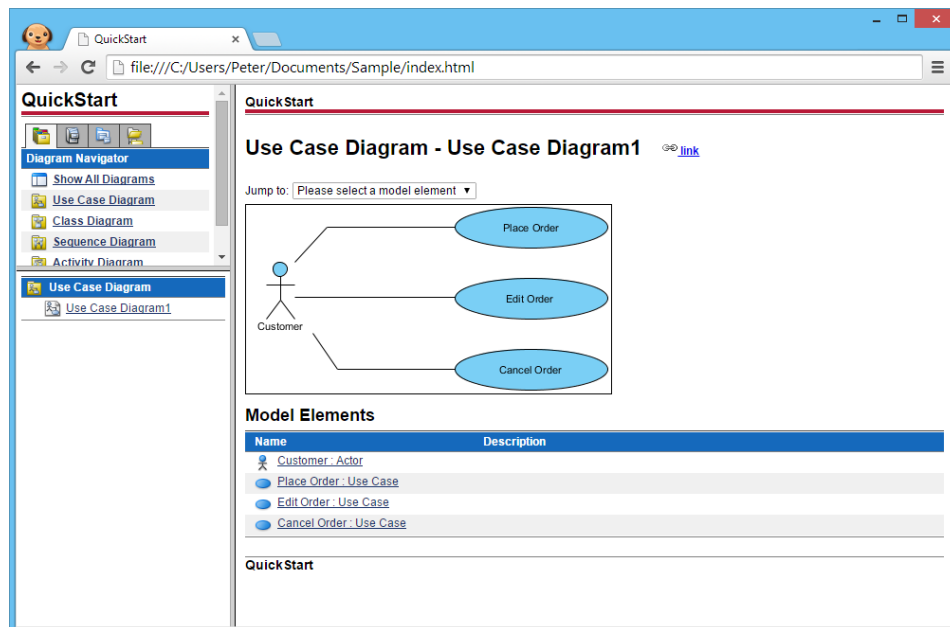
9. Now, click **Preview** to take a look at the end result. A temp. document will be produced, like this:



If you want to know more about Doc. Composer, click [here](#) to download the Doc. Composer. Writer's Guide.

## Project Publisher

Project Publisher helps produce Web contents from your project file.



To publish project with Project Publisher:

1. Select **Tools > Doc > Publish Project...** from the toolbar.
2. In the **Project Publisher** window, enter the output folder.
3. Click **OK** to start publishing.

## Collaborative Modeling

[All Editions](#)

If you work as a team and need to work together on the same design, or to share your design with your teammates, you need to make use of VPository.

VPository is the cloud-based repository where designs can be stored. You can upload your project to VPository and allow your teammates to open and edit it.

### Subscribing to VPository

In order to use VPository, you have to subscribe to it first. Subscription of VPository is made inside Visual Paradigm. To subscribe to VPository for your team:

1. Select **Team > Login** from the toolbar of Visual Paradigm.
2. In the **Teamwork Client** window, click **Subscribe to VPository...** at bottom left.
3. You should see the VPository subscription page opened in your web browser. Enter your name, email address and the cloud entry point. The cloud entry point is the unique URL of your VPository. Your team will access the entry point to enter VPository for managing and opening the projects stored in VPository.
4. Read through the Terms of Service and Privacy Policy. Check **I agree to VPository Teams of Service and Privacy Policy** if you agree to them.
5. Click **Subscribe to VPository**.

6. Check your Email client. You should receive an Email with **VPository subscription confirmation** as the subject.
7. Click on the hyperlink **Confirm your VPository Subscription** in the Email.

8. This opens another web page, asking you for details about the repository and you, the manager of the VPository account. Enter the information and click **Start My VPository**.

9. You've finished subscribing to VPository. Now, go back to Visual Paradigm. You should see that the login details have been filled for you automatically.

10. Click **Login**. Because your VPository is newly setup, you are suggested to create a project in it. Just click **OK** to continue.
11. This shows the **Import Project to VPository** screen. If you are opening the Quick Start project, the option **Currently Opened Project** should be selected for **Import from**. Just keep it selected. In practice, you can change to import a new project or to import an existing .vpp project file to VPository.

12. At the bottom of the window, click **Add Project Member**.
13. Let's invite your teammate to work on this project. Click **Invite New Member...** at the bottom.
14. Enter the name and Email address of your teammate and click **OK**.

15. Click **Import**. Invitees will receive invitation Emails. By accepting the invitation, they can login VPository from Visual Paradigm, with the cloud entry point and their own password (provided during the acceptance of invitation).
16. Click **Open** at the bottom right of the **Teamwork Client** window to open the imported project.

## Importing a Project

To import a new project to server:

1. Select **Team > Utilities > Open Teamwork Client** from the toolbar of Visual Paradigm.
2. In the **Teamwork Client** window, select **Project > Import Project to Repository** from its menu.
3. Specify the project source in the **Import from** field, edit the **project name**, **author** and **description**.

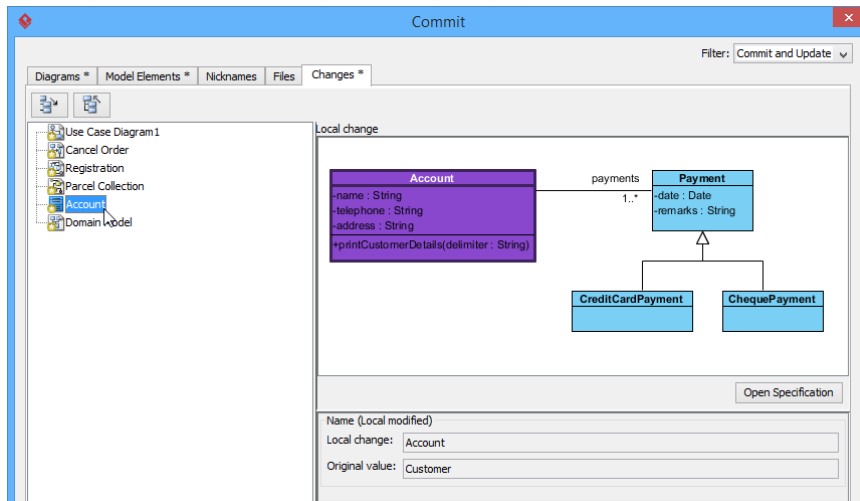
4. Invite your teammate(s) by clicking **Add Project Member** at the bottom of the window.
5. Click **Import**.

## Committing

Commit is the process of uploading changes made in the working copy back to server. When you, one of the team members, make changes in a project, you can share your works by committing those changes to the VPository and let others to update the changes from VPository.

To commit changes:

1. Select **Team > Commit** from the toolbar.
2. Review the changes in the **Commit** window, which involves both the changes you made and the changes made by others and will be updated to your project.



3. Click **Commit**.



When finish committing, the changes you made will be uploaded to server and at the same time, latest changes in server are updated to your local project copy.

## Updating

Update is the process of refreshing the working copy by merging changes that others have made and committed to server.

To update changes from VPository:

1. Select **Team > Update** from the toolbar.
2. Review the changes in the **Update** window.
3. Click **Update**.

## Sharing Your Design with PostMania

When you finish your design, you might want to share it with someone in order to ask him/her for verification, approval or take certain action based on your design (e.g. to execute a process plan modeled with a BPD). In Visual Paradigm, an effective way to share your design is by using PostMania.

PostMania is a tool built into VPository. It allows you to share your design with someone. It also allows the reader known as a “viewer” in PostMania to share his comments with you.

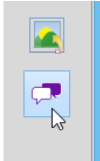
## Sharing Diagram

To share your design with PostMania:

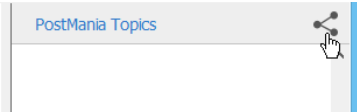
1. Open the diagram you want to share with others.
2. Open the **PostMania Topic Pane** by opening the action bar. Click on the tiny button on the right hand side of the diagram to open the action bar.



3. Select **PostMania Topic Pane** from the action bar.



4. Click on the **Share** button at the top of the **PostMania Topic Pane**.



5. In the **Share Diagram** window, enter the name and Email address of the people to share with.

Share Diagram

Share diagram with someone else

Share this diagram with someone, say your client, stakeholder, teammate, or whomever. Allow him/her to view the diagram online and give you some feedback. By supplying their name and email addresses below, we will send them invitation email, inviting them to join PostMania for viewing and commenting diagrams.

People to invite

Victor victor@demo-vp.com

Name Email

☐ Invitation Note

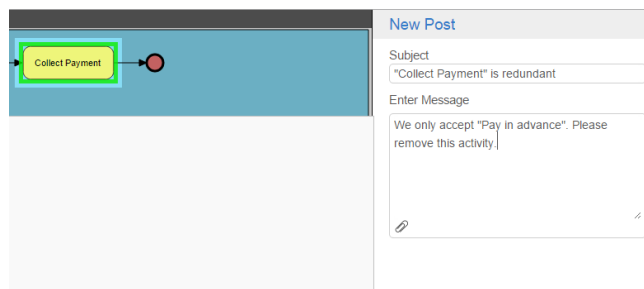
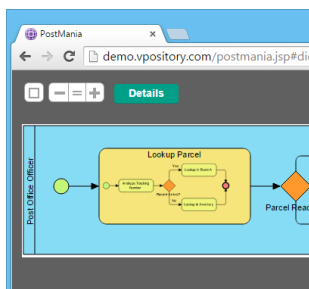
☐ Viewer List

☐ Share by Link

Bulk Share Send Invitation Email Cancel

6. Click **Send Invitation Email**. In order for these people to view the shared diagram, they have to join PostMania. Here we are going to send them invitation Emails.

Invitation Emails will be sent out in three minutes. Invitees can then accept the invitation to join PostMania. After that, they can view the diagram you shared in web browser. They can also post comments to diagrams, to shapes or to properties like description, flow of events (for use case), etc.



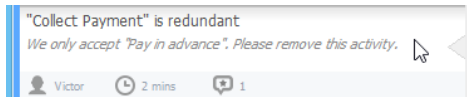
## Reacting to Comments

When a viewer has posted a comment, as a member, you will see a notification in status bar, like this:

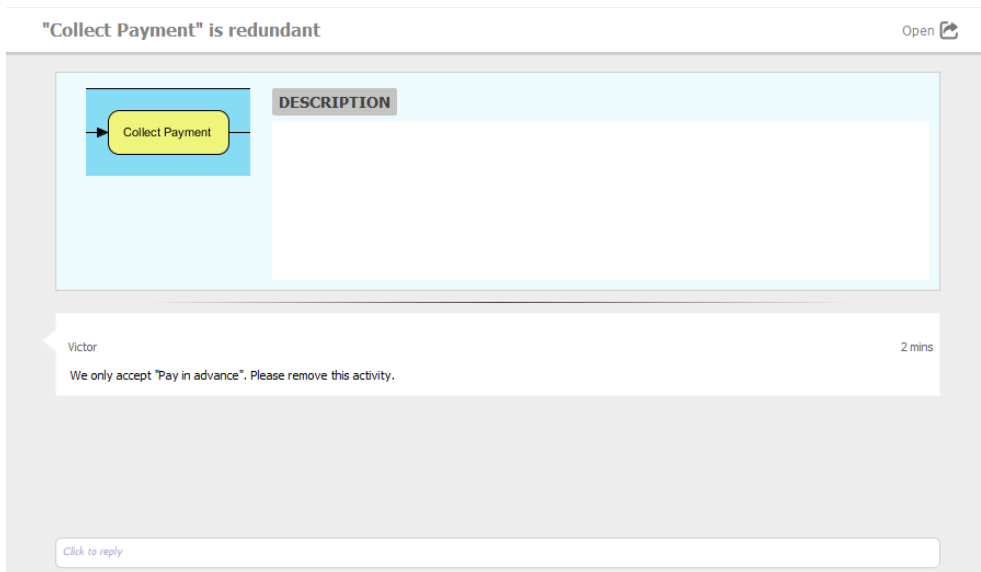
## Visual Paradigm Quick Start



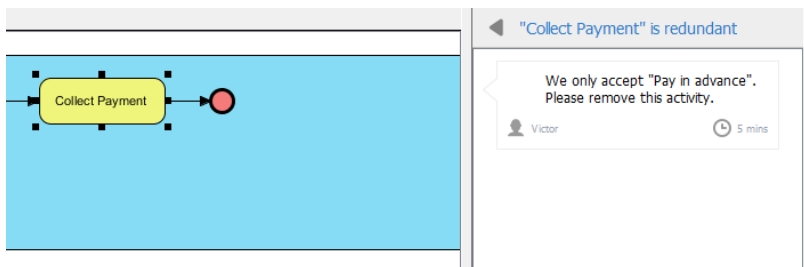
Click on it to open PostMania. Notifications are listed on the left hand side. Click on a notification to view its detail.



By clicking on a notification you can see the unread conversation listed on screen. You can give a quick reply at the bottom of the screen, or click **Open** at the top right corner to open the related diagram.



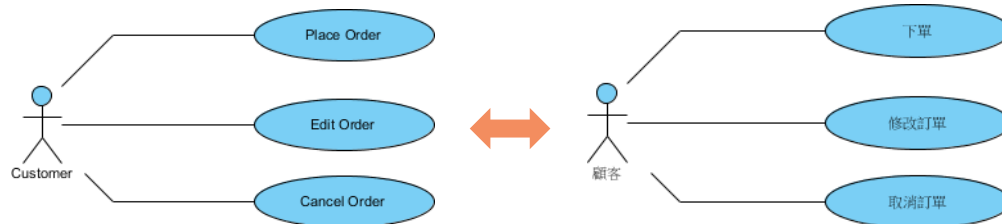
If you click **Open**, the related diagram will be opened, with highlight to the discussed area. You can then refine the design based on client's request, commit the change and add a reply in the **PostMania Topic Pane** to ask the viewer to verify the change.



## Advanced Modeling

### Nickname

If you are working in or working for multinational corporations, you may need to maintain your model in multiple languages. The Nickname feature helps you achieve that. It allows you to maintain multiple languages in one single project without the need to keep multiple project files for same content.



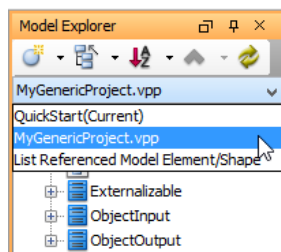
To add a new language:

1. Select **Modeling > Nickname > Configure Nicknames...** from the toolbar.
2. In the **Configure Nickname** window, click **Add User Language....**
3. In the **Add User Language** window, select the language to add.
4. The newly added language is selected in the **Configure Nicknames** window. Keep it selected and click **OK**.
5. Now, you can rename the shapes and diagrams to “translate” the model to the language added. To switch the model back to the original language, select **Modeling > Nickname > Original** from the toolbar.

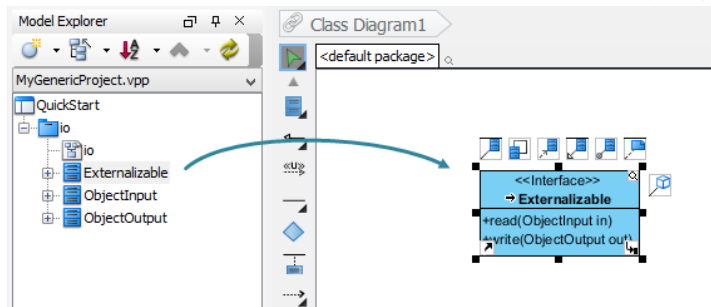
### Project Reference

To avoid creating the same things (e.g. a class) over and over again, it would be useful to have a generic library to keep components for reuse. When you make any changes to the components in the library, those changes will ripple down to where the components are actually used. In Visual Paradigm, we call this generic library a “Reference Project”. To add project reference and re-use the data in referenced project:

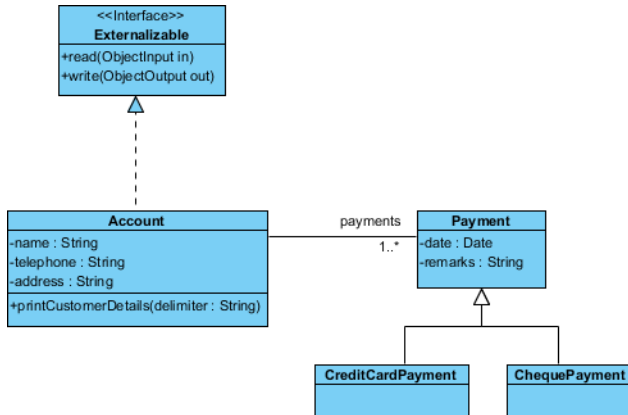
1. Select **Project > Referenced Projects** from the toolbar.
2. In the **Manage Referenced Projects** window, click **Add**.
3. Choose the .vpp project file to reference to and click **Open**.
4. Close the **Manage Referenced Projects** window by clicking **Close**.
5. Open the **Model Explorer**. The referenced project(s) are listed in the drop down menu at the top of the **Model Explorer**. You can switch between the current project and the referenced project through the drop down menu to see the elements in them.



6. You can re-use a model element from referenced project by drag-and-drop.



7. You can connect referenced project data with the data of the current editing project.



## Using Mirror

The benefit of using referenced model is to prevent your working project from becoming oversized as the information of referenced model will not be stored. However, since the referenced model is read-only from its source project, you cannot create a child to it. To deal with this problem, you can create mirror for parent-type elements such as package (Right click on a referenced element and select **Create Mirror Model Element**). The mirrored model element is also read-only on its properties. However, you can add a child model to it.



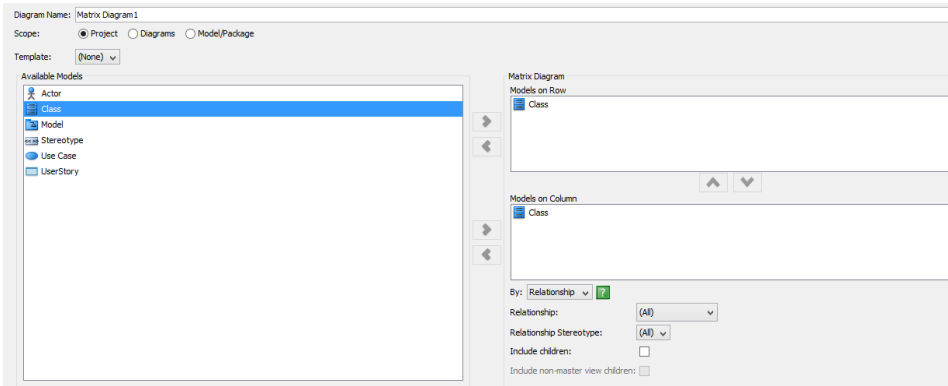
## Impact Analysis

If you want to make a change to some model elements, it would be important to know which other elements will get affected because of it. Impact Analysis can help you with that.

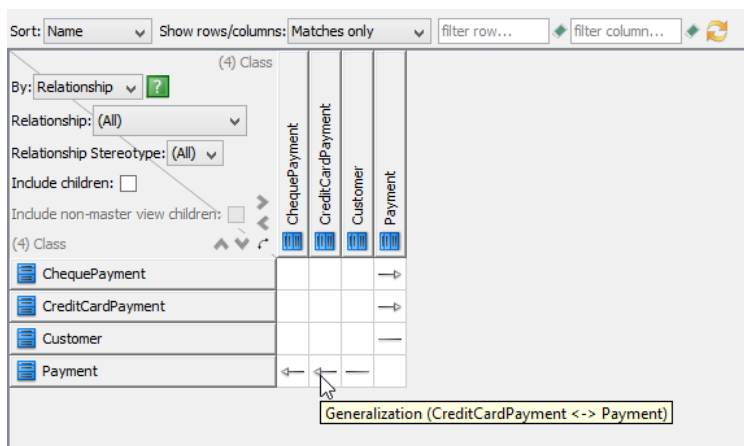
### Matrix Diagram

Matrix Diagram helps you to identify the relationship between model elements of specific type(s), so as to study the consequence of making certain changes. To create a matrix:

1. Create a matrix by selecting **Modeling > Impact Analysis > Matrix** from the toolbar.
2. Let's say if you want to inspect the relationship among all the classes in the project. Add class to both **Models on Row** and **Models on Column** by first selecting class on the left hand side and clicking > to add it into row and column list respectively.
3. Select **Relationship** for **By**. By here means to compare row and column items by the selected criterion.



4. Click **OK**. This produces a matrix which lists the classes in rows and columns, showing their relationships in cells.



Let's say you are thinking about deleting the *Payment* class. By reading the matrix, you realize that the *Payment* class is a super class of *CreditCardPayment* and *ChequePayment*. Deleting the super class *Payment* may risk losing data. So you'd probably need to consider withdrawing the deletion or to move the data from super class to sub class, etc.