DT228-3 Software Engineering III - Lab 1 (Week 2).

Rational Software Architect (RSA) – Introduction.

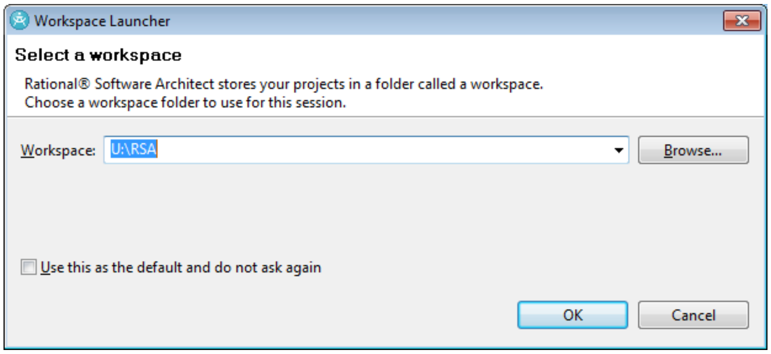
PART I: Creating a Requirements Model – Use Case Diagram

This part of the lab sheet is aimed at showing students how to use RSA to create an initial *Requirements Model*. A requirements model will hold our Actors, Use Cases and Use Case Diagrams. In this lab we will create the Restaurant Booking system's requirements model as described in Mark Priestley's book. In *part II* we will look at creating the *Analysis Model*.

Task 1 – Running RSA

Start up RSA: If there is no icon on the desktop, click the *Start Menu*, choose *All Programs* and scroll to the *IBM Software Delivery Platform* folder. Within this folder, click *Rational Software Architect*.

You will be prompted to choose a workspace (a folder for all your RSA projects and models). If you are using a lab PC, ensure you are using your own drive. Or possibly **D:\My Documents\RSA** on your personal PC or laptop.



Workspace Launcher

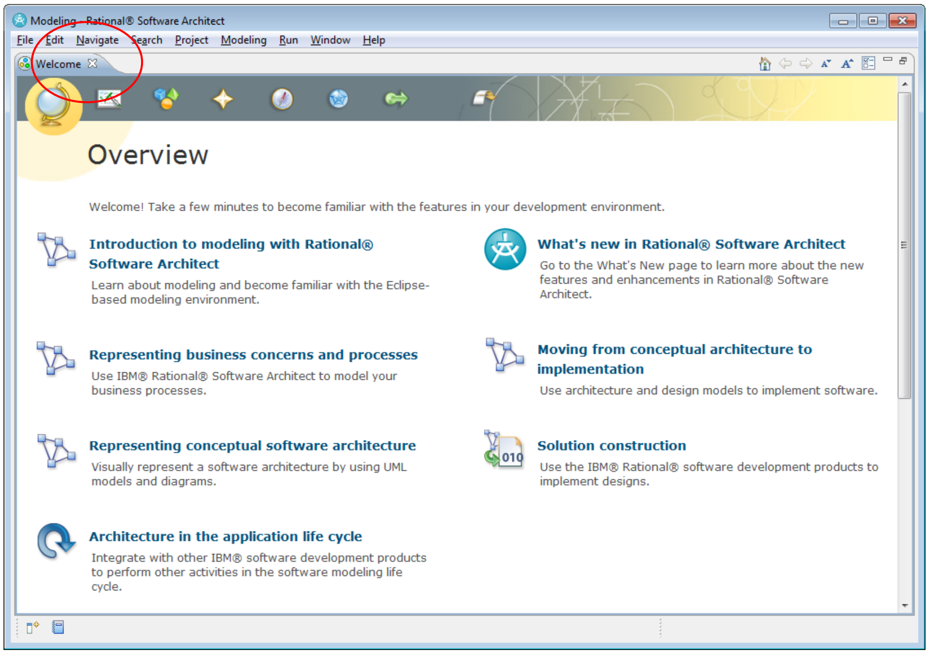
Select a workspace

Rational® Software Architect stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: U:\RSA

□ Use this as the default and do not ask again

You may be presented with the *Welcome* tab as below. The icons on this can direct you to tutorials, samples, first steps and other useful material. You can close down the *Welcome* tab as shown below.


Modeling - Rational® Software Architect
File Edit Navigate Search Project Modeling Run Window Help
Welcome

Overview

Welcome! Take a few minutes to become familiar with the features in your development environment.

Introduction to modeling with Rational® Software Architect
Learn about modeling and become familiar with the Eclipse-based modeling environment.

What's new in Rational® Software Architect
Go to the What's New page to learn more about the new features and enhancements in Rational® Software Architect.

Representing business concerns and processes
Use IBM® Rational® Software Architect to model your business processes.

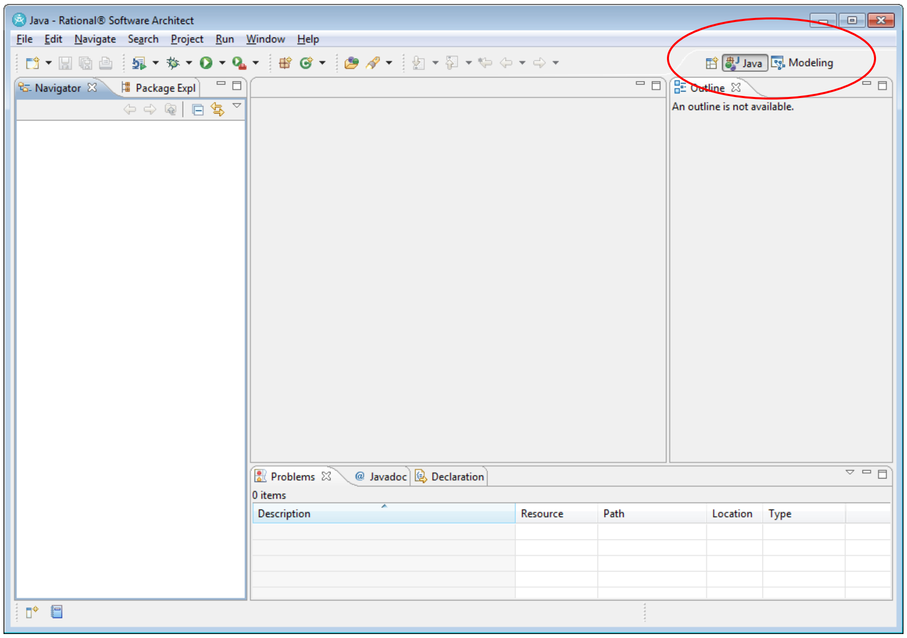
Moving from conceptual architecture to implementation
Use architecture and design models to implement software.

Representing conceptual software architecture
Visually represent a software architecture by using UML models and diagrams.

Solution construction
Use the IBM® Rational® software development products to implement designs.

Architecture in the application life cycle
Integrate with other IBM® software development products to perform other activities in the software modeling life cycle.

Following the closing of the *Welcome* tab you may get something like:


Java - Rational® Software Architect
File Edit Navigate Search Project Run Window Help
Java
Modeling
Navigator
Package Expl
Outline
Problems
Javadoc
Declaration

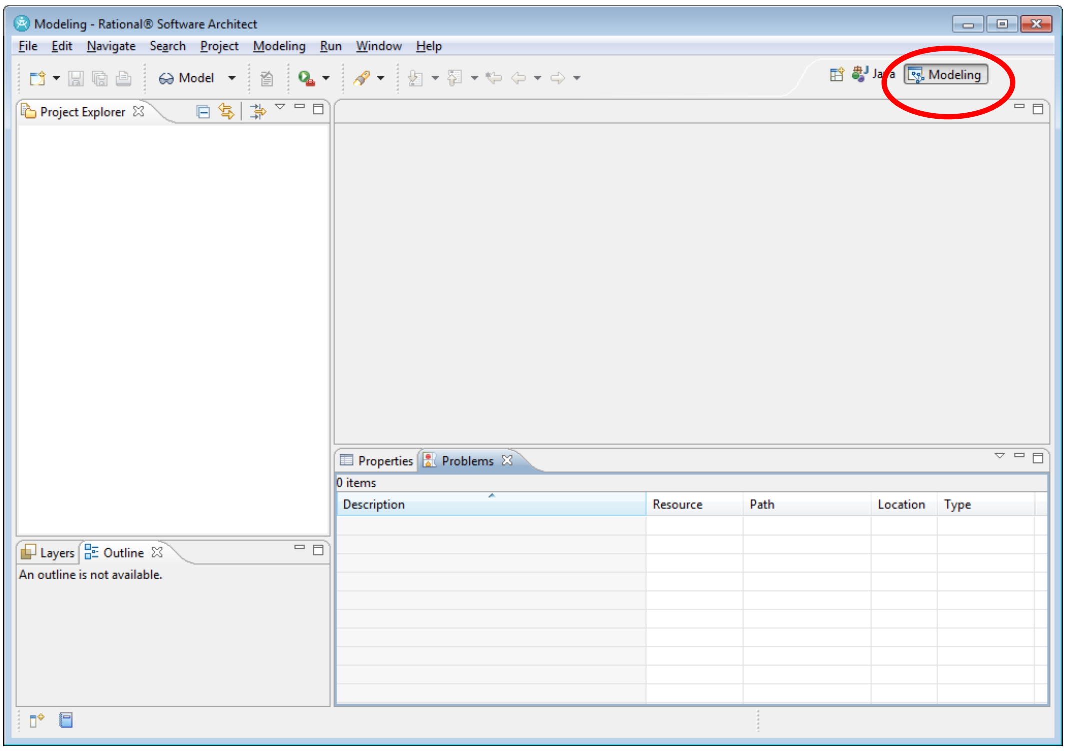
An outline is not available.

0 items

Description	Resource	Path	Location	Type

RSA has many different perspectives (combinations of views) which are specific to the role you wish to perform (e.g. java developer, UML modeller). The above diagram shows that RSA is in the **Java perspective**. You need to be in the **Modelling perspective.**

You can change perspective by clicking on the **Open Perspective** icon on the top right-hand of the window and then selecting the perspective you want. Here we want the *Modelling* perspective as below.

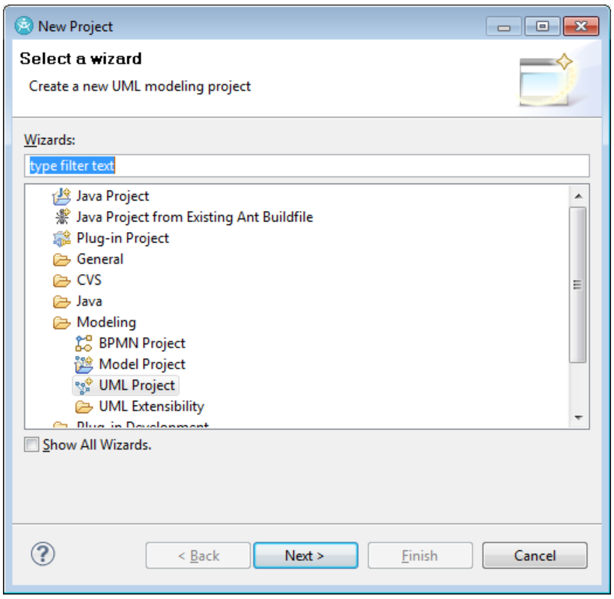


0 items

An outline is not available.

Task 2 – Create Initial *Requirements Model*

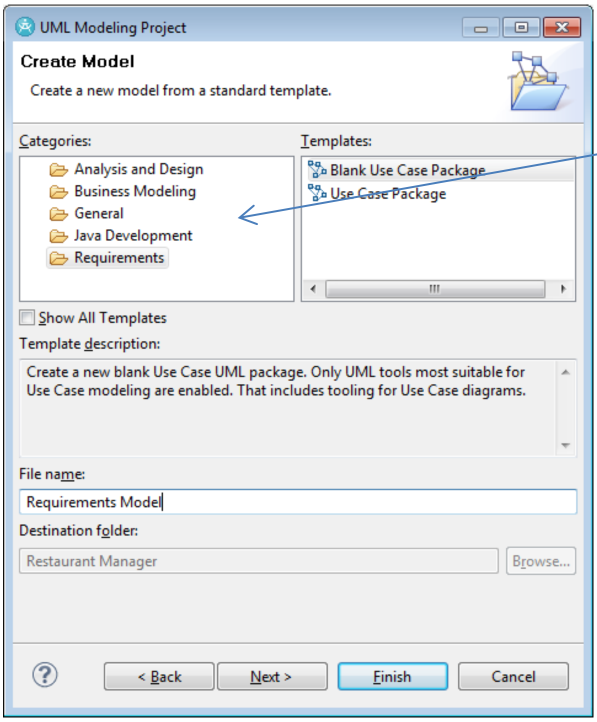
- Next we will create a simple *Requirements Model* for the Restaurant Booking System example. Choose *File -> New -> Project…* and then choose *UML Project* as below:



Select a wizard

Create a new UML modeling project

- Choose *Next* and then name the project *Restaurant Manager*, click *Next*.
- On the next screen (as below), choose the *Requirements* category on the left and the *Blank Use Case Package* template on the right (**please read the note beside the screenshot**).
- Change the *File name* to *Requirements Model* and click *Finish*



Create Model

Create a new model from a standard template.

□ Show All Templates

Template description:

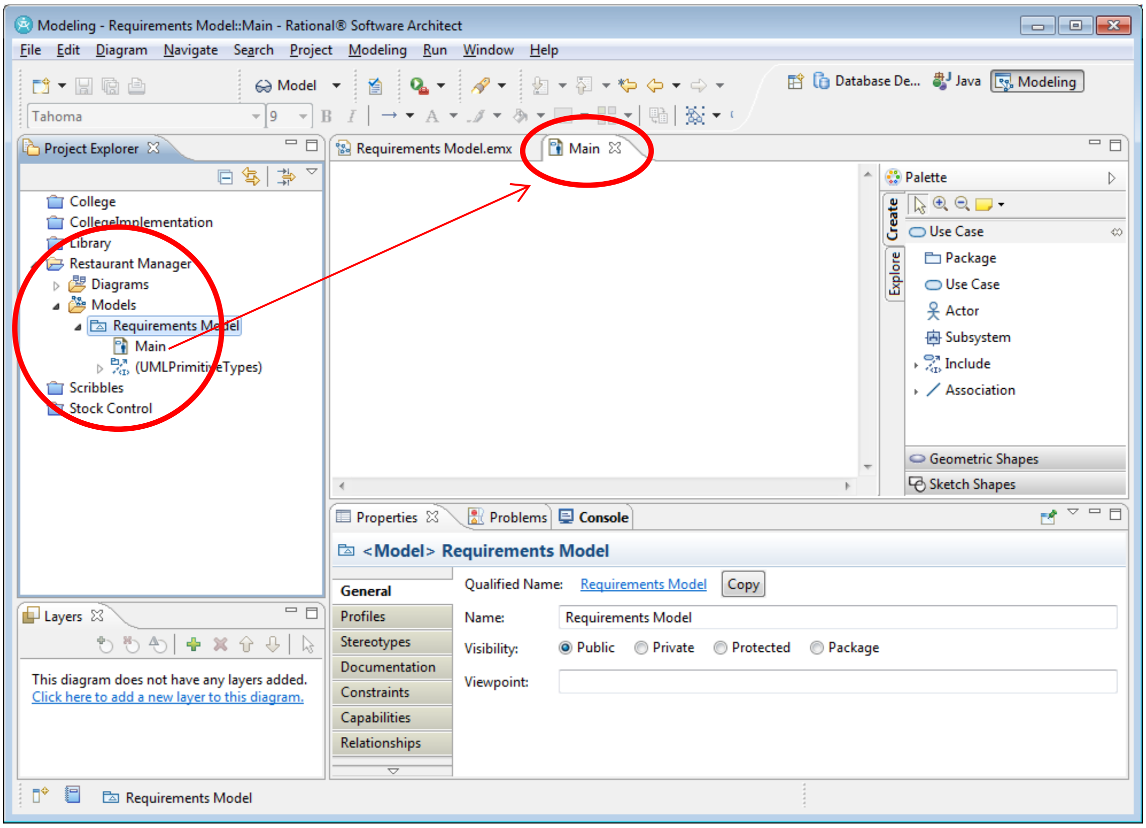
Create a new blank Use Case UML package. Only UML tools most suitable for Use Case modeling are enabled. That includes tooling for Use Case diagrams.

File name: Requirements Model

Destination folder: Restaurant Manager

Note: Depending on the version of RSA, you might not see all the *Categories* listed in the screenshot – if that is the case just choose *Blank Model*, also you have the option to choose *Use Case Diagram* as the default diagram to create (go ahead and choose that option).

- You should have the following screen:



This diagram does not have any layers added.
Click here to add a new layer to this diagram.

<Model> Requirements Model

Qualified Name: Requirements Model

Name: Requirements Model

Visibility: ● Public ○ Private ○ Protected ○ Package

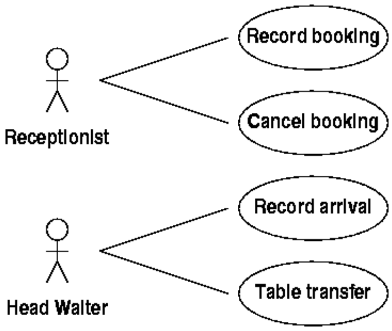
Viewpoint:

Note:

You now see your *Requirements Model* under the Models folder. Within this model, we will create our *Actors*, *Use Cases* and *Use Case Diagram*. Note that a default empty *Use Case Diagram* has already been created for us called *Main*. This is also open in the editor as shown above.

Task 3 – Add the Actors to our Requirements Model

Consider the following use case diagram from the book:

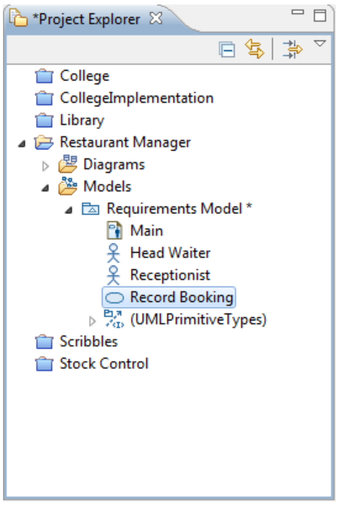


We have identified two actors, *Receptionist* and *Head Waiter*. Add these actors to our requirements model as follows:

- Right-click on *Requirements Model* in the Project Explorer and choose *Add UML -> Actor* and name it *Receptionist*
- Repeat the process to create the second actor.

Task 4 – Add some initial Use Cases to our Requirements Model

- Right-click on *Requirements Model* in the Project Explorer and choose *Add UML -> Use Case* and name it *Record Booking*. You should see the following in your Project Explorer:



- Repeat the process to add the *Cancel Booking* use case. You should now have another use case in your model above the *Record Booking* use case.

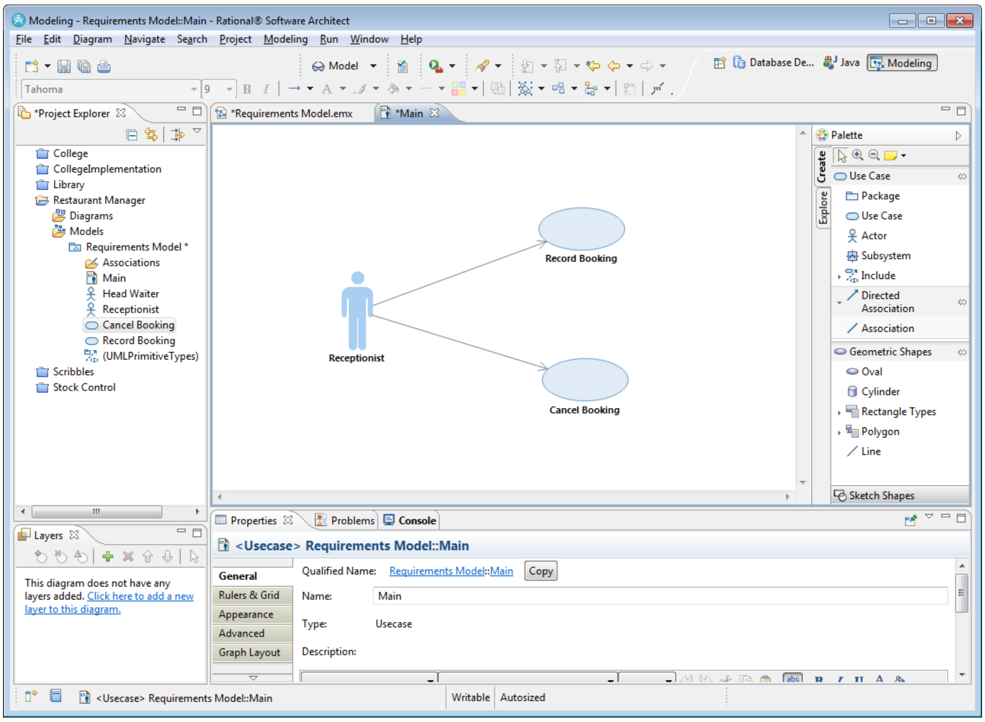
Task 5 – Develop your Use Case Diagram

There are two main ways to easily add to your Use Case diagram

1. You can drag an **existing model element** (such as an actor or a use case) directly from your requirements model within the project explorer and drop it into the use case diagram.
2. Alternatively, you can drag a **new model element** from the *Palette* on the right and drop it into the diagram. This will create a new model element in your requirements model and so you should then be able to see that new element in the project explorer.

Tip: If you delete an element in your diagram, depending on how you do it, it may or may not delete the element from your model (remember, the diagram is just a view of your model). Sometimes students do this and then add the elements to the diagram again using the Palette which creates another element in the model and so you end up with duplication.

- Use the first method above to add the *Receptionist* actor and both use cases to the *Main* use case diagram.
- From the Palette on the right, click the *Directed Association* model element (**note**: if you do not see a *Directed Association* model element, just pick the *Association* element and delete the rolenames and multiplicities). Then click on the actor in your diagram and drag the line to one of the use cases (ensure the use case is highlighted before you release the mouse button).
- Add another *Directed Association* (or *Association*) from the actor to the second use case. You should now have something like the screen below:



This diagram does not have any layers added. Click here to add a new layer to this diagram.

<Usecase> Requirements Model::Main

Qualified Name: Requirements Model::Main

Name: Main

Type: Usecase

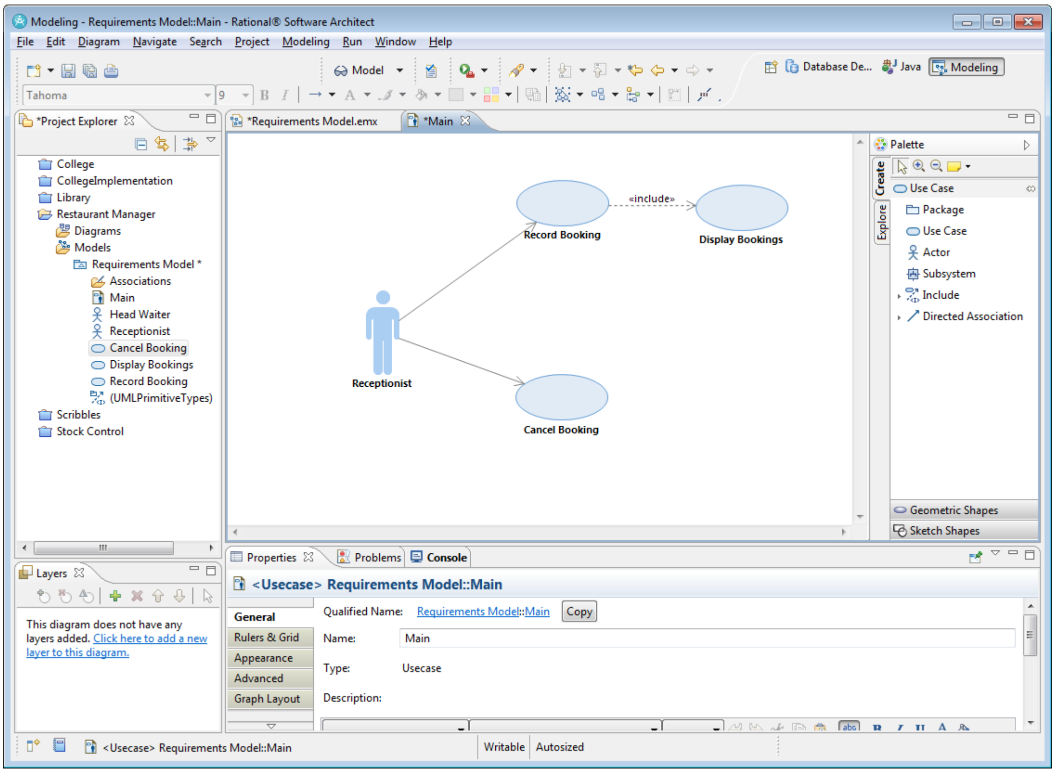
Description:

<Usecase> Requirements Model::Main

Tip: To get straight lines between the actor and use case, you can right-click on the association and choose *Appearance …* then select the *oblique* style within the styles section and click *Apply*.

Task 6 – Add an <<includes>> relationship

- Drag a new use case from the Palette in to your diagram and name it *Display Bookings*. ***Note****: this creates a new use case in our model which you can see in the project explorer).*
- Click on an *include* association in the Palette. Click on the *Record Booking* use case and drag the include line to the new *Display Bookings* use case. You should have something like this:



This diagram does not have any layers added. Click here to add a new layer to this diagram.

<Usecase> Requirements Model::Main

Qualified Name: Requirements Model::Main

Name: Main

Type: Usecase

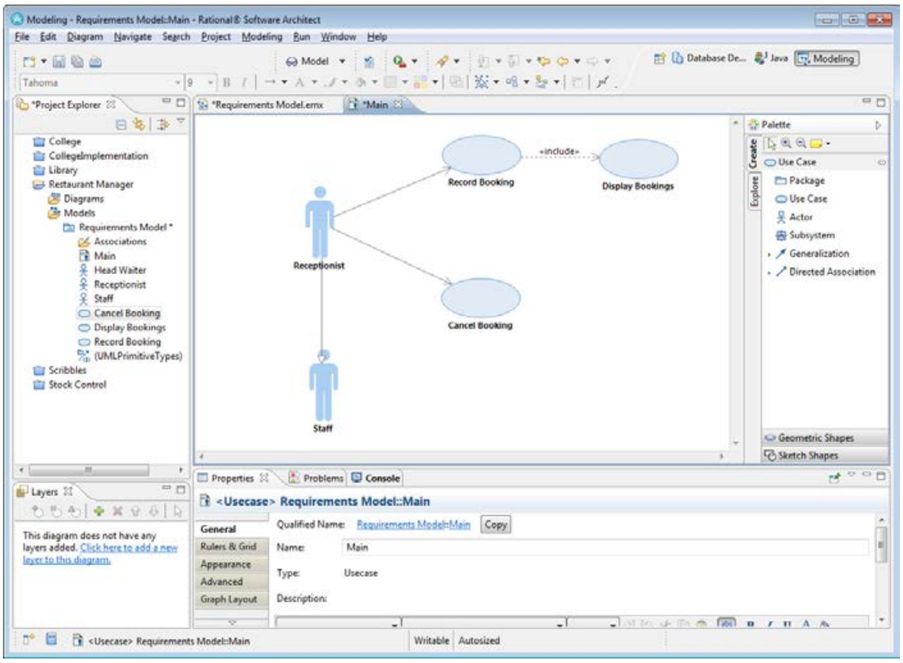
Description:

<Usecase> Requirements Model::Main

Task 7 – Create a Generalisation for your Actors

In this task we will create a *Generalised* actor called *Staff* and we will create a generalisation between the existing actors and the new *Staff* actor.

- As before or by using the Palette, create a new actor called Staff and add it to the use case diagram.
- Using the Palette, create a *Generalisation* association showing that the *Receptionist* actor is a specialisation of the *Staff* actor. You should have something like this:



<Usecase> Requirements Model::Main

Qualified Name: Requirements Model::Main

Name: Main

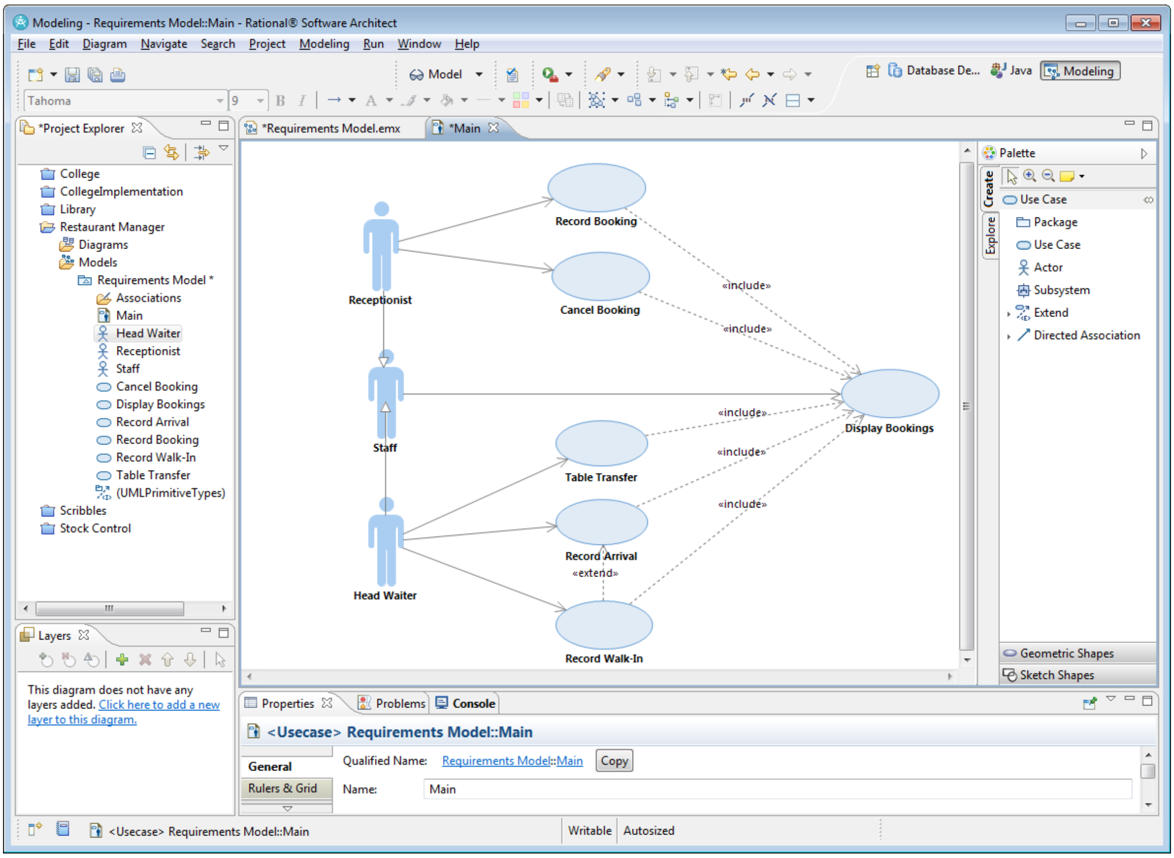
Type: Usecase

Description:

<Usecase> Requirements Model::Main

Task 8 – Complete the Use Case Diagram

- Add to your requirements model and diagram so that you end up with the use case diagram below.
- Note that the *Record Walk-In* use case *extends* the *Record Arrival* use case and also that the *Head Waiter* actor is a specialisation of the *Staff* actor.
- Also note that we have modelled that a *Staff* actor interacts with the *Display Bookings* use case. Because we have also modelled that both the *Receptionist* and *Head Waiter* actors are specialisations of the *Staff* actor we do not then explicitly model that they interact with the *Display Bookings* use case – this is implicit due to the generalisation / specialisation between the actors.



<Usecase> Requirements Model::Main

Qualified Name: Requirements Model::Main

Name: Main

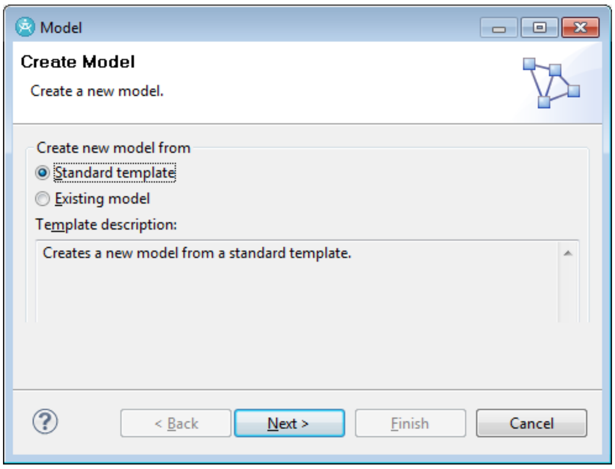
<Usecase> Requirements Model::Main

PART II: Creating an initial Analysis Model – Class Diagram

In this part of the lab sheet, we will create an initial Analysis Model for the Restaurant Booking System. This will entail creating a class diagram firstly.

Task 1 – Create a new Model

- In the project explorer, right-click on *Models* and select *Create Model*. You should see the following screen:



Model

Create Model

Create a new model.

Create new model from

● Standard template

○ Existing model

Template description:

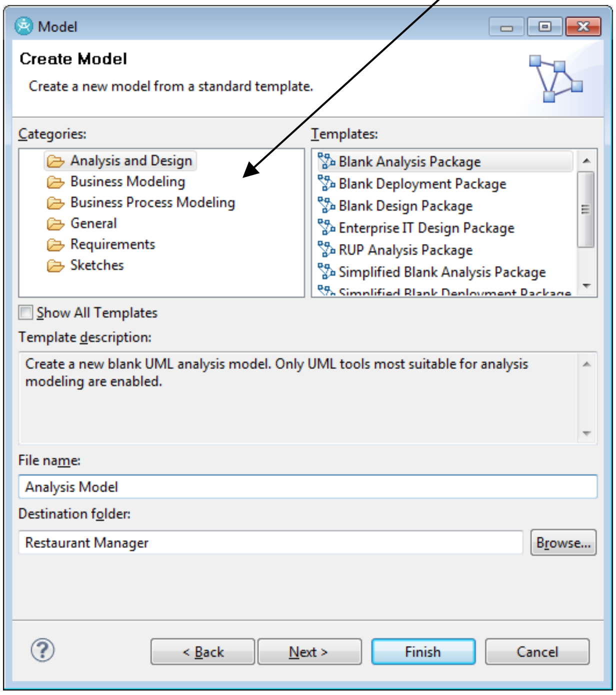
Creates a new model from a standard template.

Click *Next*. On the next screen (*Create Model)* do the following as shown below:

- Ensure the *Analysis and Design* category is selected on the left and then select the *Blank Analysis Package* template on the right
- Change *File name* to ”Analysis Model”
- Click *Finish*

You should now have a screen which is something like:

Note: Depending on the version of RSA, you might not see all the *Categories* listed in the screenshot – in this case just choose *Blank UML Model*, also you have the option to choose *Class Diagram* as the default diagram to create (go ahead and choose that option).



Create Model

Create a new model from a standard template.

□ Show All Templates

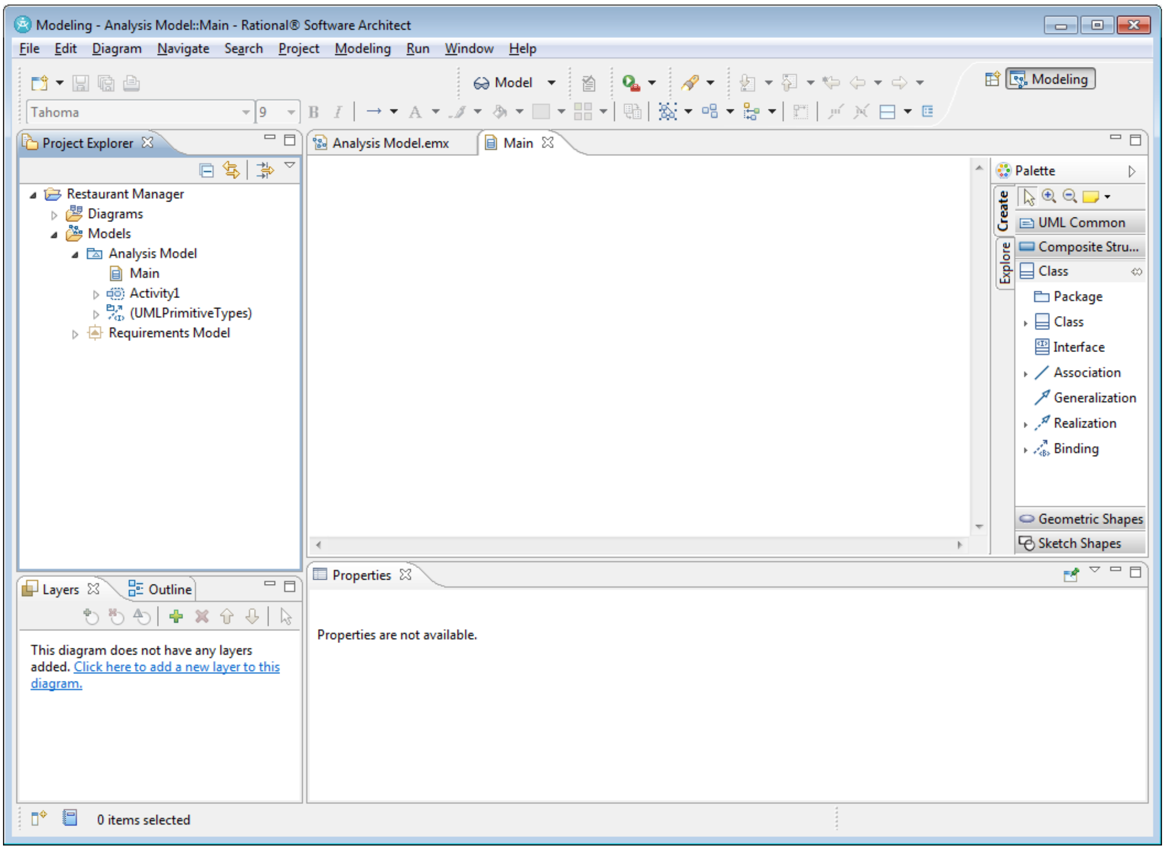
Template description:

Create a new blank UML analysis model. Only UML tools most suitable for analysis modeling are enabled.

File name: Analysis Model

Destination folder: Restaurant Manager

- Click *Finish* and you should see the screen below:

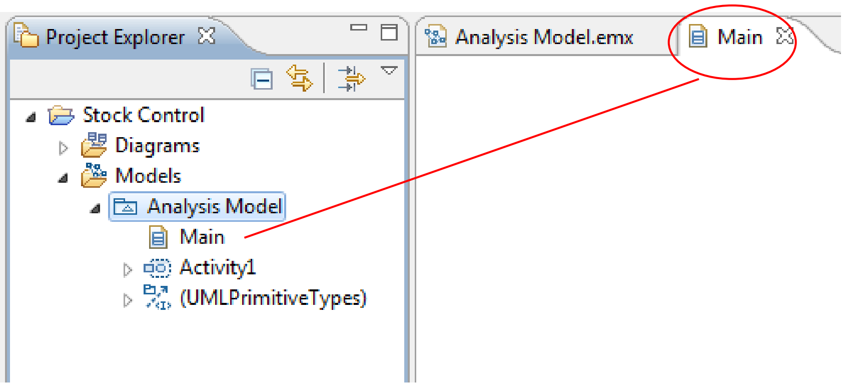


This diagram does not have any layers added. Click here to add a new layer to this diagram.

Properties are not available.

0 items selected

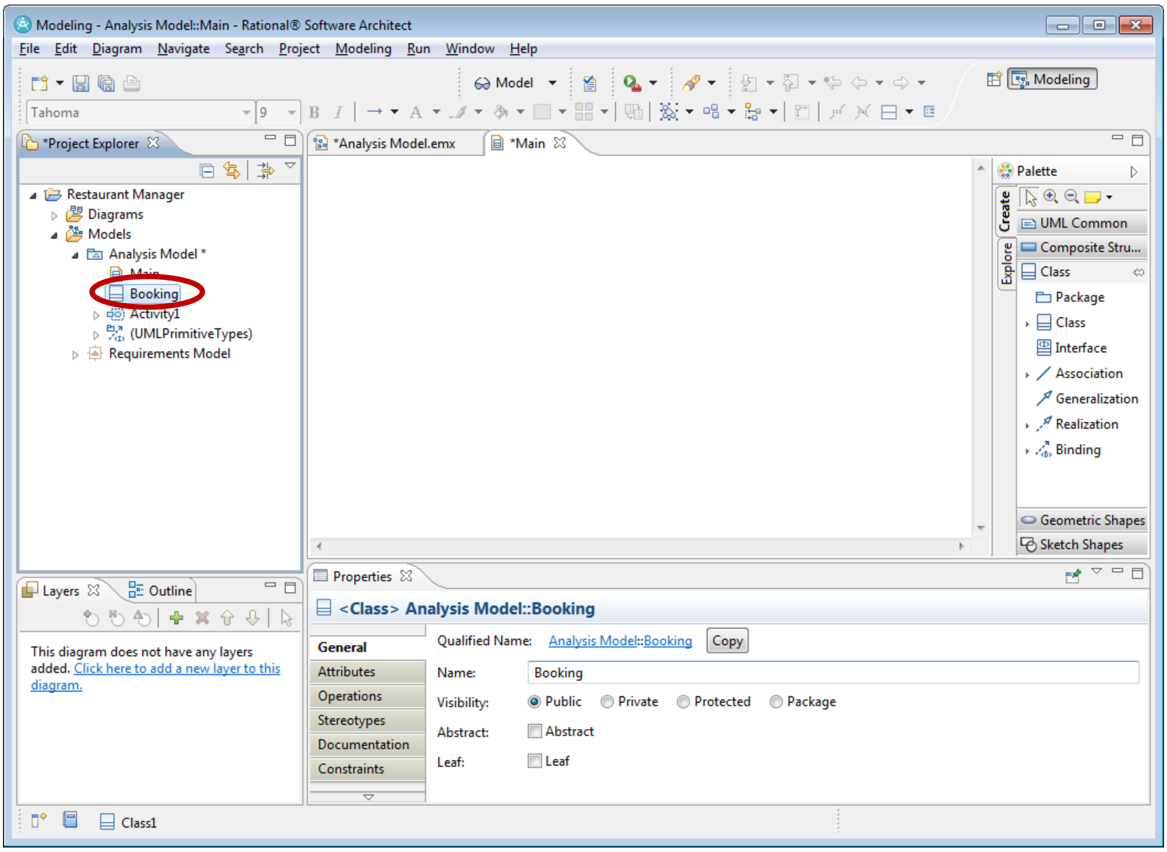
In the above screen, to the left, we see the *Analysis Model* in the Project Explorer. To the right we see the Palette which provides model elements that we use in our UML *class diagram*. By expanding the *Analysis Model* in the Project Explorer, we will see that RSA has created a default empty *class* *diagram*. The name of this class diagram is *Main* and it has been opened in the model editor. Note the icon to the left of *Main* in the Project Explorer – these differ depending on the type of UML diagram it is.



Task 2 – Create some Classes

Here we will create a number of classes for our Booking System. These will then be part of our *Analysis Model*. There are two ways in which we can do this: we can add the class to our model using the Project Explorer or we can drag a *Class* from the Palette into our *class diagram*. By dragging the class from the palette into our diagram, RSA automatically adds the class to our model which can be seen in the project explorer.

- In the project explorer window, right click on *Analysis Model* and choose *Add UML -> Class,* name the class *Booking*. You should have the following screen (note that the *Booking* class is now a part of our analysis model:



This diagram does not have any layers added. Click here to add a new layer to this diagram.

<Class> Analysis Model::Booking

Qualified Name: Analysis Model::Booking

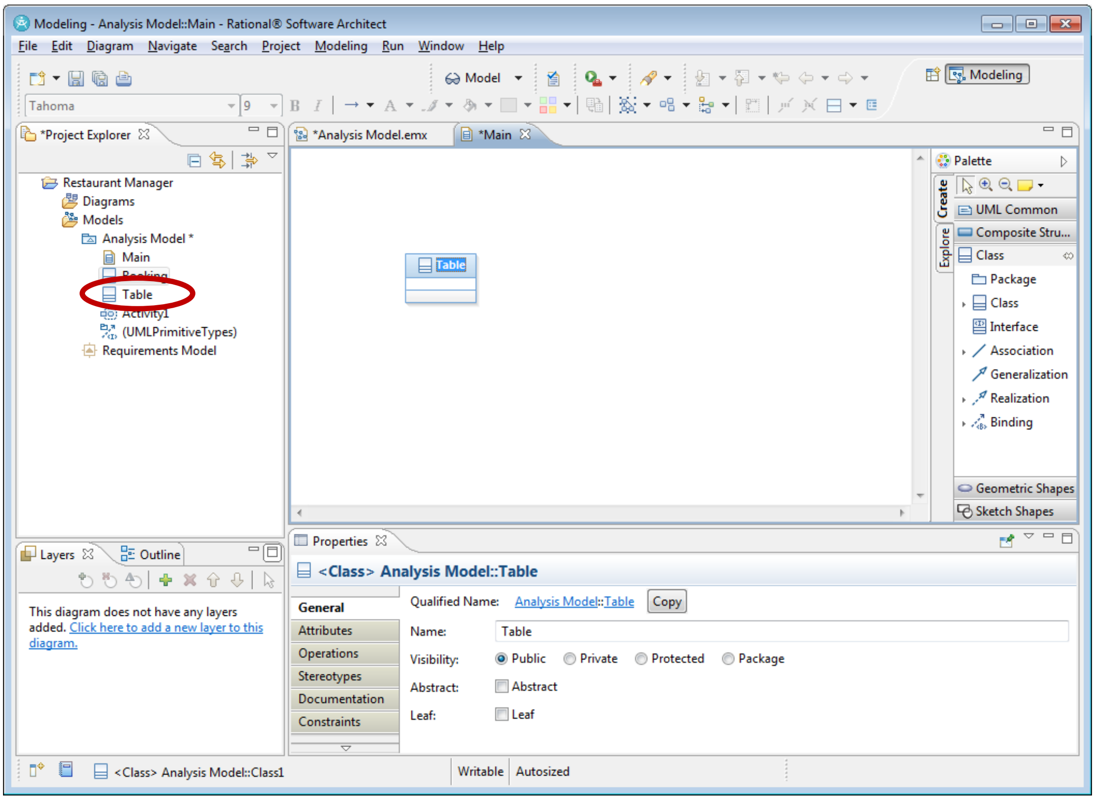
Name: Booking

Visibility: ● Public ○ Private ○ Protected ○ Package

Abstract: □ Abstract

Leaf: □ Leaf

- Next, drag the *Class* icon from the Palette into the *Main* diagram that is open in our model editor and name it *Table*. We have created two classes using different techniques. You should now have the following screen (note that the *Table* class is now part of our model also):



<Class> Analysis Model::Table

Qualified Name: Analysis Model::Table

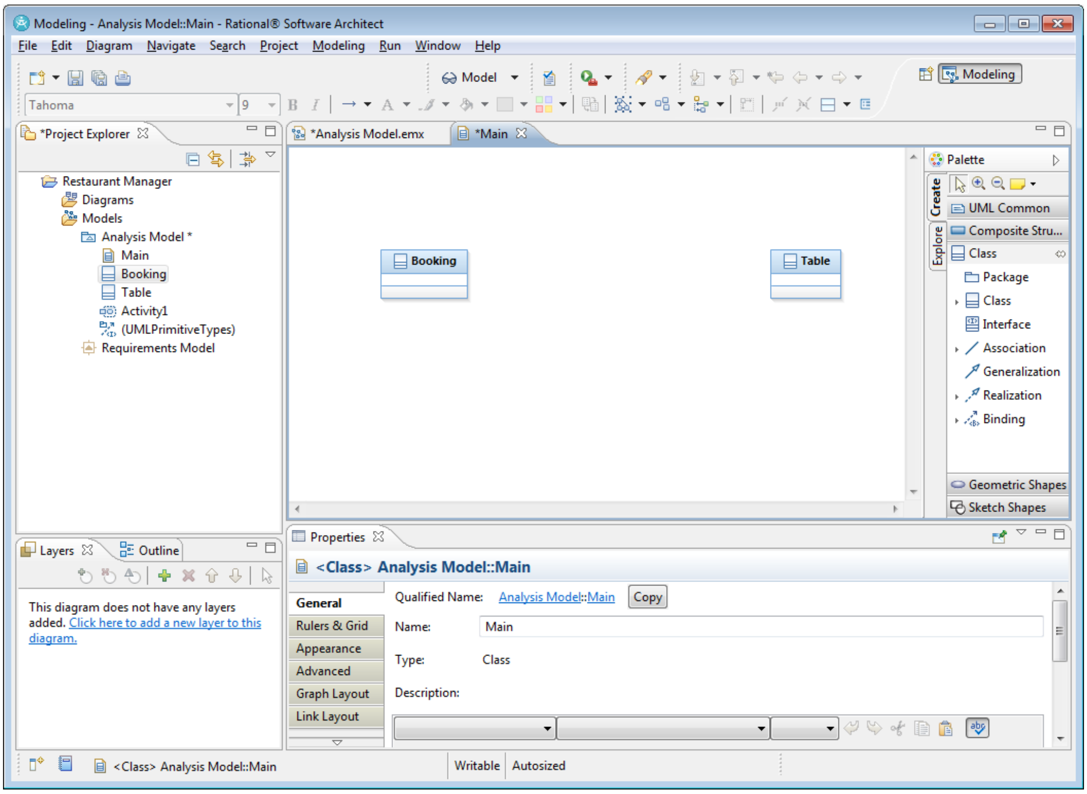
Name: Table

Visibility: ● Public ○ Private ○ Protected ○ Package

Abstract: □ Abstract

Leaf: □ Leaf

- Next, drag the *Booking* class from the Project Explorer into the *class diagram*. You can move them around so that you should have the following:



<Class> Analysis Model::Main

Qualified Name: Analysis Model::Main

Name: Main

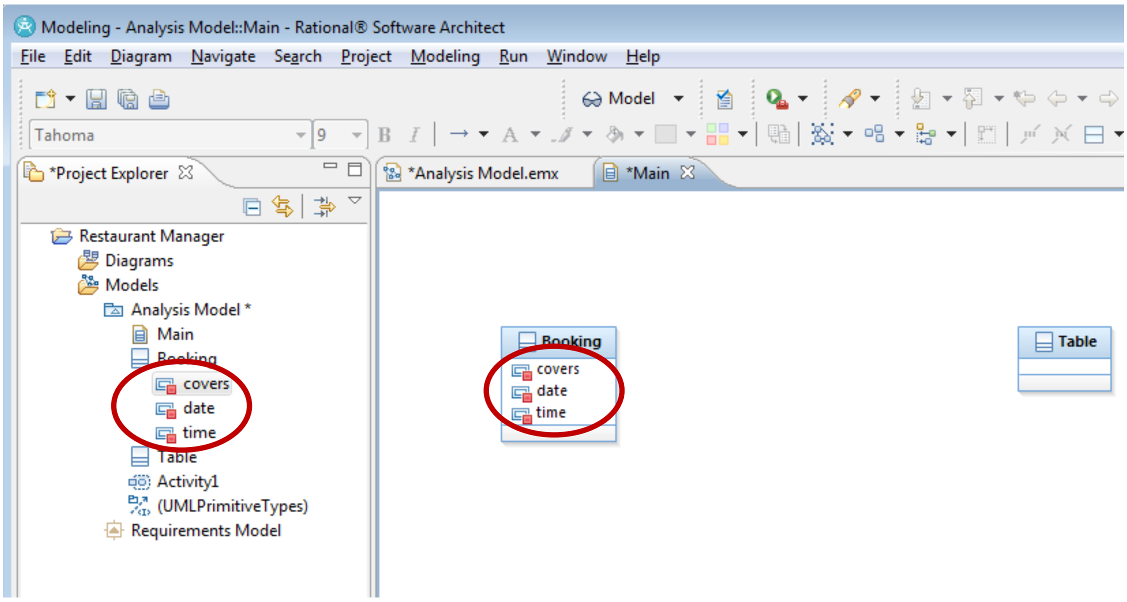
Type: Class

Description:

Task 3 – Add Class Attributes

We will add three attributes to the *Booking* class.

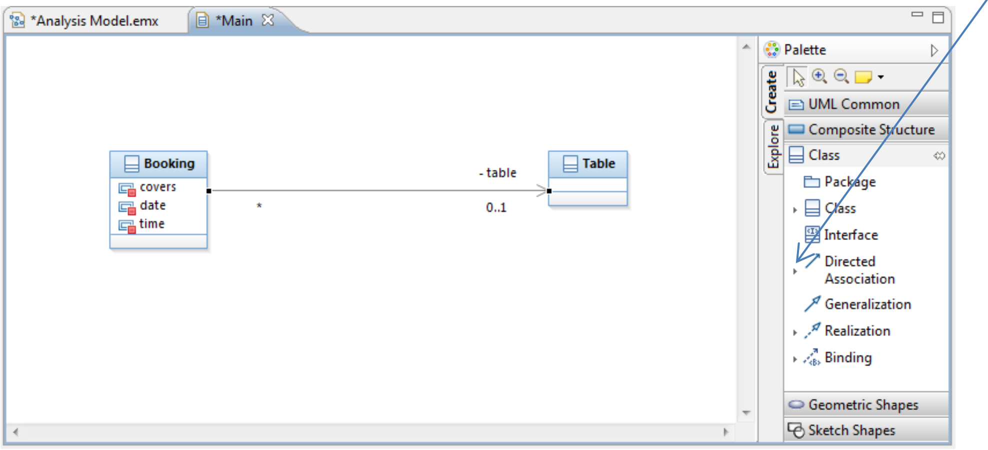
- Within the Project Explorer, right-click on *Booking* and choose *Add UML -> Attribute* and name it “covers”.
- We can also add attributes to classes within the class diagram – right click on the *Booking* class within the class diagram and choose *Add UML -> Attribute* and name it “date”. Note the attribute has been added in to our Model (see the Project Explorer).
- Finally, add in another attribute called “time”. You should now have a screen like:



Task 4 – Add an Association

We will add an association to our model. The association will show how the *Booking* class and *Table* class are associated.

- Within the Palette on the right, click once on the *Directed Association* model element (see note in the margin below). Once you have done that, click and hold the mouse button while over the *Booking* class in the diagram and drag it over to the *Table* class – this should draw a line between the two denoting the association. You should see the following screen:



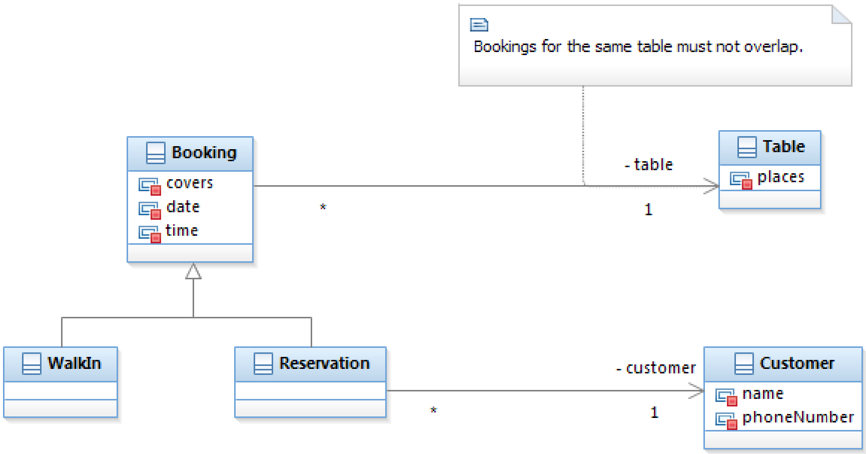
Note the arrow to the left of the elements in the Palette. You can click on this to expand the selection. If you only see *Association* and not *Directed Association* then just use *Association*.

There are a number of things to note here:

- RSA has added an association name to the *Table* side of the association. This shows that an instance (object at runtime) of the *Booking* class would have a reference called *table* which may point to a *Table* object.
- Also, the arrow denotes that the association is one way (unidirectional) i.e. a *Table* object would not have any reference to a *Booking* object even though they are linked.

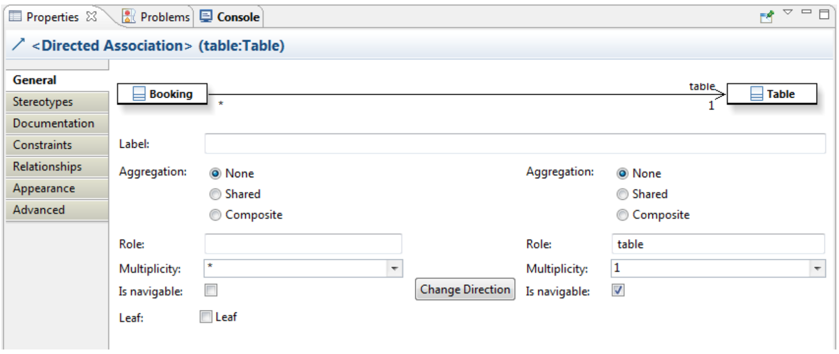
Task 5 – Exercise

See if you can finish the Object Model so that it looks like the screenshot below. There are some notes below to help.



Note: Changing the properties (navigability, multiplicity) of associations

- When you click on an *association* between two classes to select it, your *Properties View* will show the various properties of that association – you should see something like this:



<Directed Association> (table:Table)

Label:

Aggregation: ● None ○ Shared ○ Composite

Role:

Multiplicity: *

Is navigable: □

Leaf: □ Leaf

Aggregation: ● None ○ Shared ○ Composite

Role: table

Multiplicity: 1

Is navigable: ☑

- By checking the *Is navigable* checkbox on either end of the association we can set it as unidirectional or bidirectional.
- We can also set the multiplicity as we want it by using the *Multiplicity* dropdown box on either end.

Final Note

Each RSA project is stored in a separate subfolder of your workspace folder that you specified when starting up the application. Each of these subfolders of your workspace contains all the files for the particular project.

There is also a facility to export and import your projects from RSA.

File->Export->General->Archive File

and

File->Import->General->Archive File