

Papyrus Tutorial

A Practical Guide to Systems Modeling
with Papyrus and SysML



Version 0.1.2

Contents

1	INTRODUCTION	3
1.1	Overview	3
1.2	Assumptions	3
2	PROBLEM DOMAIN	3
2.1	Introduction	3
2.2	Black-box perspective	3
2.2.1	Step 1. Create the new project and new model.	3
2.2.2	Step 2. Organize the model for Black-Box analysis.	10
2.2.3	Step 3. Capturing Stakeholder Needs (Requirements Table)	11
2.2.4	Step 4. Initial System Context	15
2.2.5	Step 5. Use Case and Activity Diagram	22

List of Figures

1	New Papyrus Project	4
2	Select Architecture Context	5
3	Project File Created	6
4	Create New Model	7
5	New Model Dialog	8
6	Vehicle CCU Problem Model	9
7	Vehicle CCU in Model Explorer	9
8	Black Box Packages	10
9	Create a New Package	11
10	Creating a New Requirements Table	12
11	Create New Requirement in Requirement Table	13
12	New Requirement in Model	13
13	User Needs	14
14	User Needs Requirements Table	14
15	Requirements Tree	15
16	System Context package	16
17	Create New Block	17
18	Vehicle In Use Block	18
19	Create Internal Block Diagram	19
20	Vehicle In Use Internal Block Diagram	19
21	Part Type Initialization Window	20
22	Part Type Initialization Window	21
23	Climate Control Unit with Label Removed	21
24	Vehicle In Use IBD	22

25	Use Case Package	22
26	Use Case Diagram Creation	23
27	New Use Case Diagram	24
28	Expand the System Context package	25
29	Vehicle In Use block in Use Case Diagram	26
30	Vehicle Occupant block in Use Case Diagram	26
31	Use Case Tool Palette	27
32	Vehicle Occupant block in Use Case Diagram	27
33	Use Case Tool Palette	28
34	Association Created	28
35	<i>Feel Comfortable Temperature as a child of 2 Use Cases</i>	29
36	Create New Activity Diagram	30
37	<i>Feel Comfortable Temperature Activity Diagram</i>	30
38	New Activity Partition (Swimlane)	31
39	Vehicle Occupant Activity Partition Creation	32
40	Vehicle Occupant Activity Partition Properties	33
41	Vehicle Occupant Activity Partition Typed by Vehicle Occupant Block	33
42	Vehicle Occupant Activity Partition in Activity Diagram	34
43	Create Climate Control Unit Activity Partition	35
44	Climate Control Activity Partition in Activity Diagram	35
45	Activity Diagram Initial Node	36
46	Create Opaque Action	37
47	<i>Turn On Climate Control Action Created</i>	37
48	<i>Turn On Climate Control Action in Activity Diagram</i>	38
49	<i>Feel Comfortable Temperature Activity Diagram</i>	39
50	Control Flow	40
51	Control Flow	41
52	Control Flow at Decision Node	41
53	Creating the <i>[OK]</i> Guard Condition	42
54	Creating the <i>[Not OK]</i> Guard Condition	43
55	Creating More Control Flows	44
56	Final Control Flows	44
57	Create Fork Node	45
58	Temperature Output	46
59	Temperature Input	47

1 INTRODUCTION

1.1 Overview

The purpose of this tutorial is to use the *Papyrus* program to create a model of a simple system utilizing the SysML language. A vehicle climate control unit will be modeled starting with the Problem Domain and then moving to the Solution Domain.

1.2 Assumptions

This tutorial assumes that *Papyrus* and the SysML tools have been properly installed and will not cover the procedures to do so. This is not a SysML tutorial but rather how to model an example system with SysML in the Papyrus program. This tutorial uses *Papyrus* version 2019-03 and SysML 1.4 running on a Windows machine.

2 PROBLEM DOMAIN

2.1 Introduction

In the problem domain we will analyze stakeholder needs and refine them with SysML model elements to gain a clear understanding of the problem that the System Of Interest (Sol) must solve. We will first analyze the Sol from a black-box perspective where we focus on the interaction of the system with the environment without knowledge of the internal structure of the Sol. We then analyze the system from a white-box perspective to perform the functional analysis of the system.

2.2 Black-box perspective

We will start this project by organizing our model and capturing our stakeholder needs.

2.2.1 Step 1. Create the new project and new model.

1. Create a new *Papyrus* project by selecting **File > New > Papyrus Project** (see figure 1).

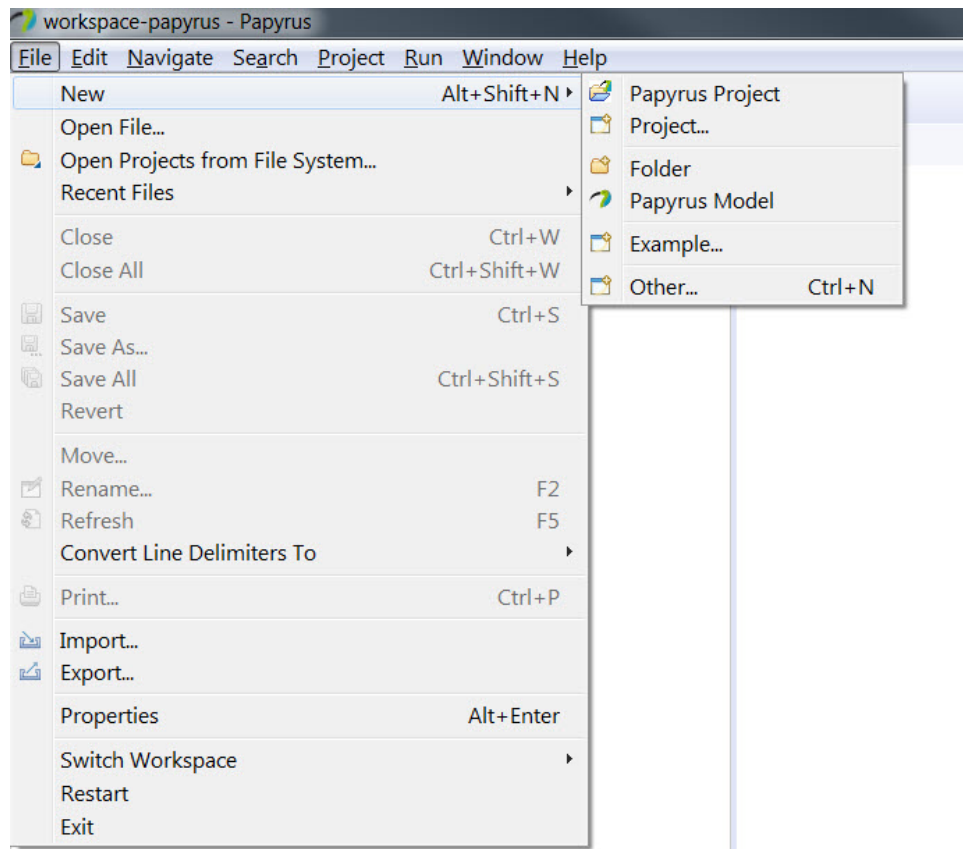


Figure 1: New Papyrus Project

2. The *Select Architecture Context* dialog will then appear. Ensure that the *SysML 1.4* checkbox is selected and then select **Next** (see figure 2).

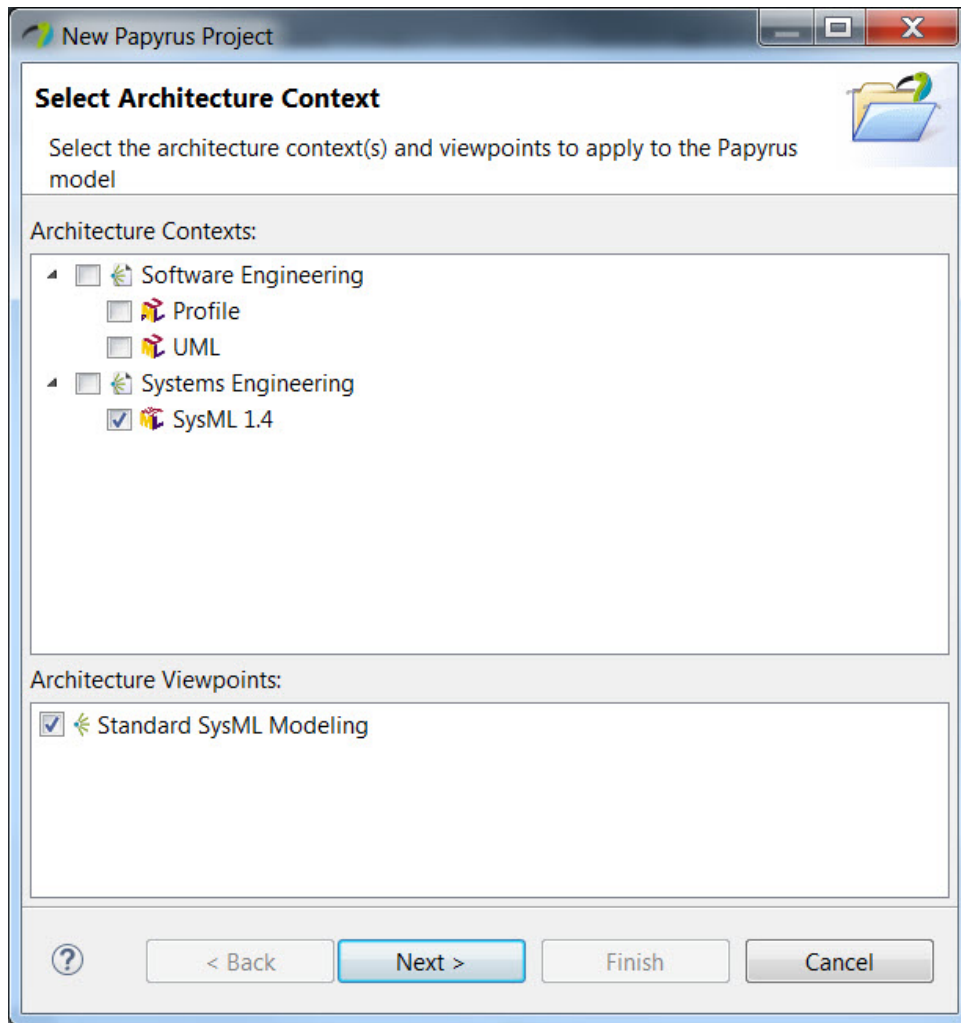


Figure 2: Select Architecture Context

3. On the next screen type *car* for the project name and click **Finish**. Your screen should look like figure 3.

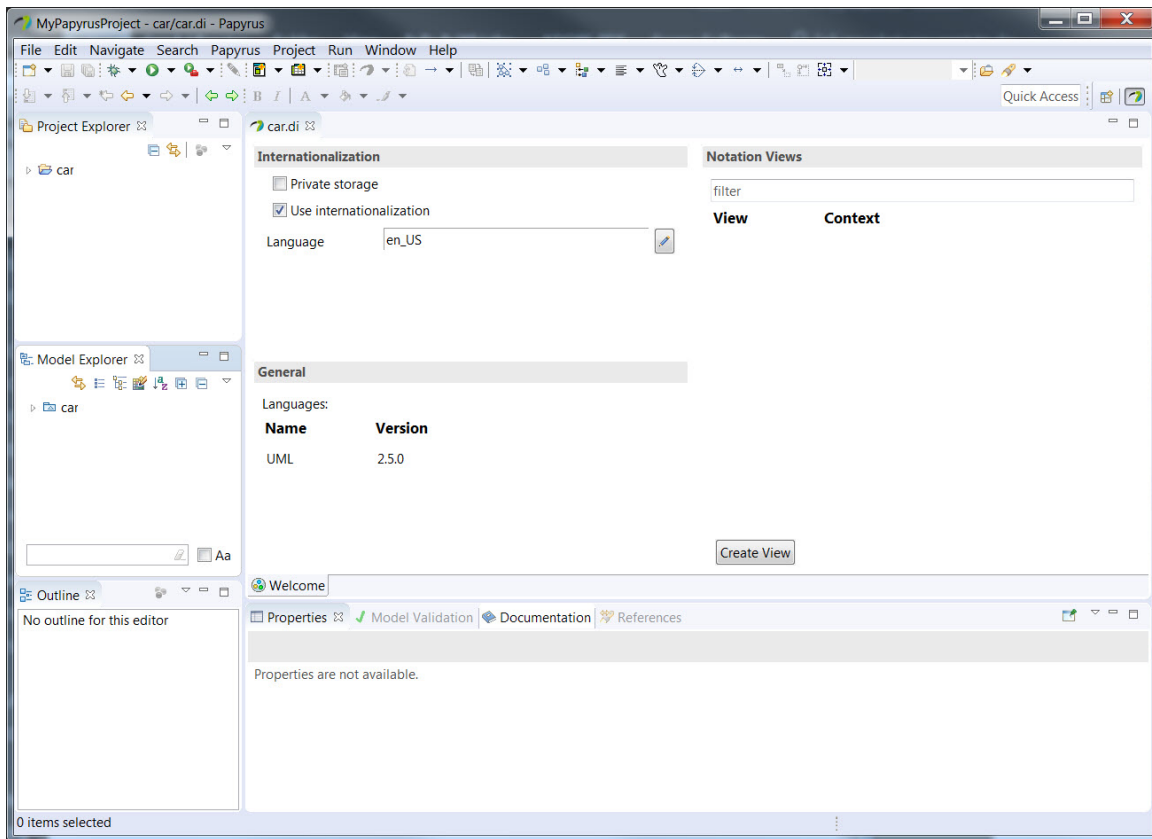


Figure 3: Project File Created

4. Now we will create our model for the vehicle climate control system problem space. Create a new *Papyrus* model by selecting **File > New > Papyrus Model** (see figure 4). Ensure that the *SysML 1.4* checkbox is selected when the *Select Architecture Context* dialog appears and then select **Next** (see figure 2).

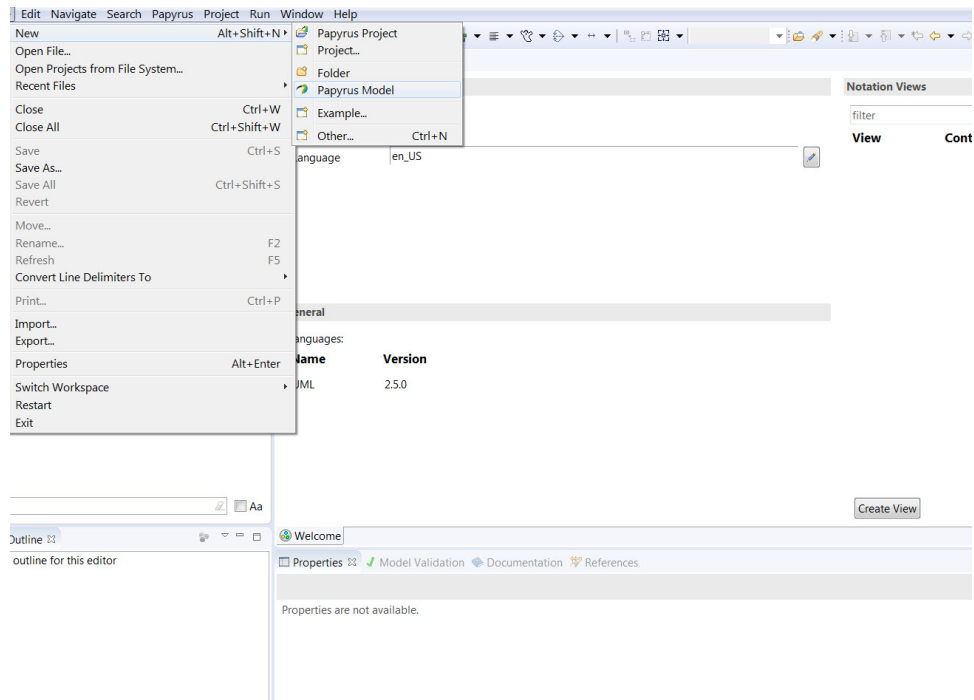


Figure 4: Create New Model

5. When the *New Papyrus Model* dialog appears, click on the *car* folder to choose this as the parent folder and enter *VehicleCCU_Problem* in the file name box and click **Finish** (see figure 5).

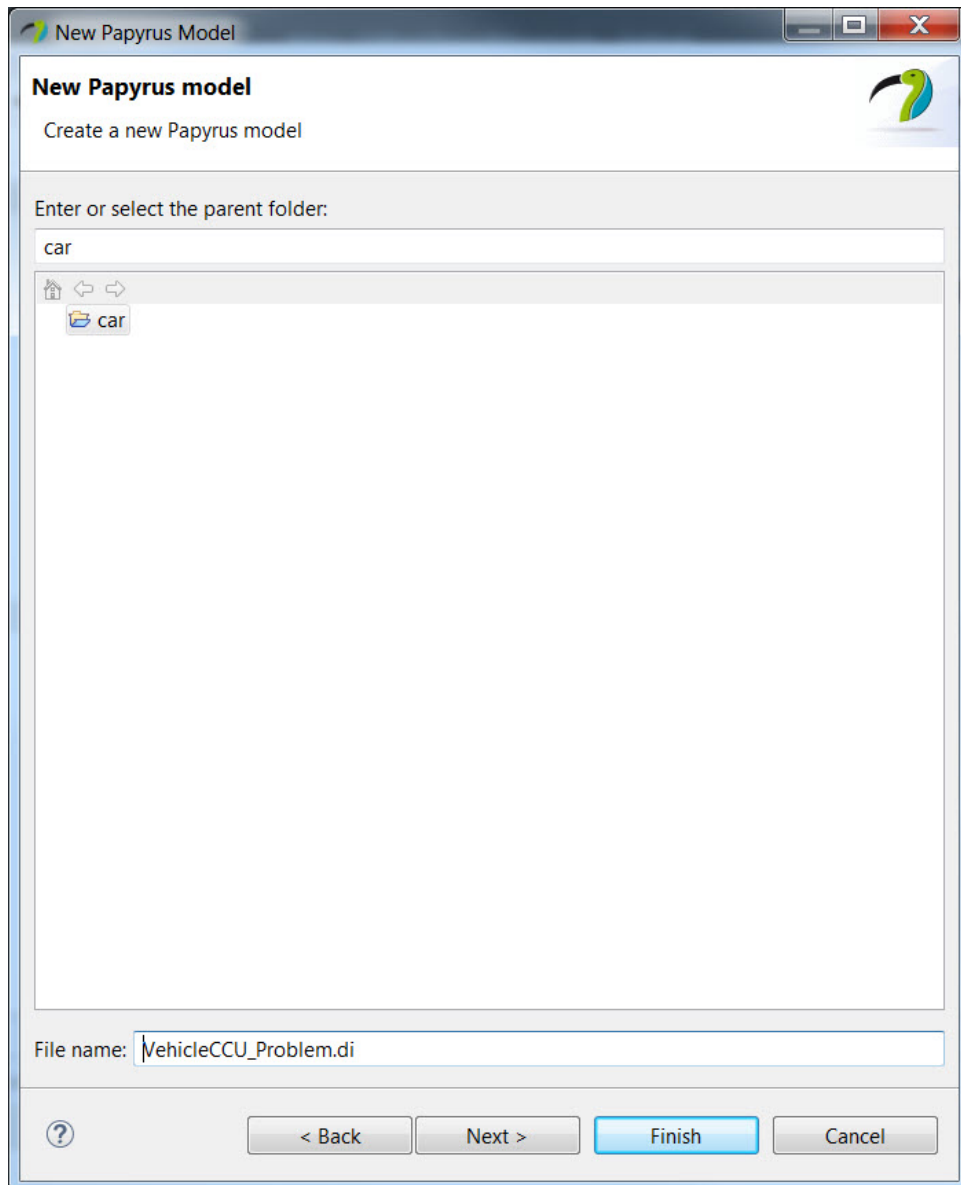


Figure 5: New Model Dialog

6. Your screen should now look like figure 6. You see that in the *Project Explorer* window we have the *car* folder with a car model and the model we created called *VehicleCCU_Problem*. The *Model Explorer* window should be showing the top level folder of our *VehicleCCU_Problem* model that we just created (figure 7). If it does not then double-click on the *VehicleCCU_Problem* in the *Project Explorer* window OR select the *VehicleCCU_Problem* tab in the main window.

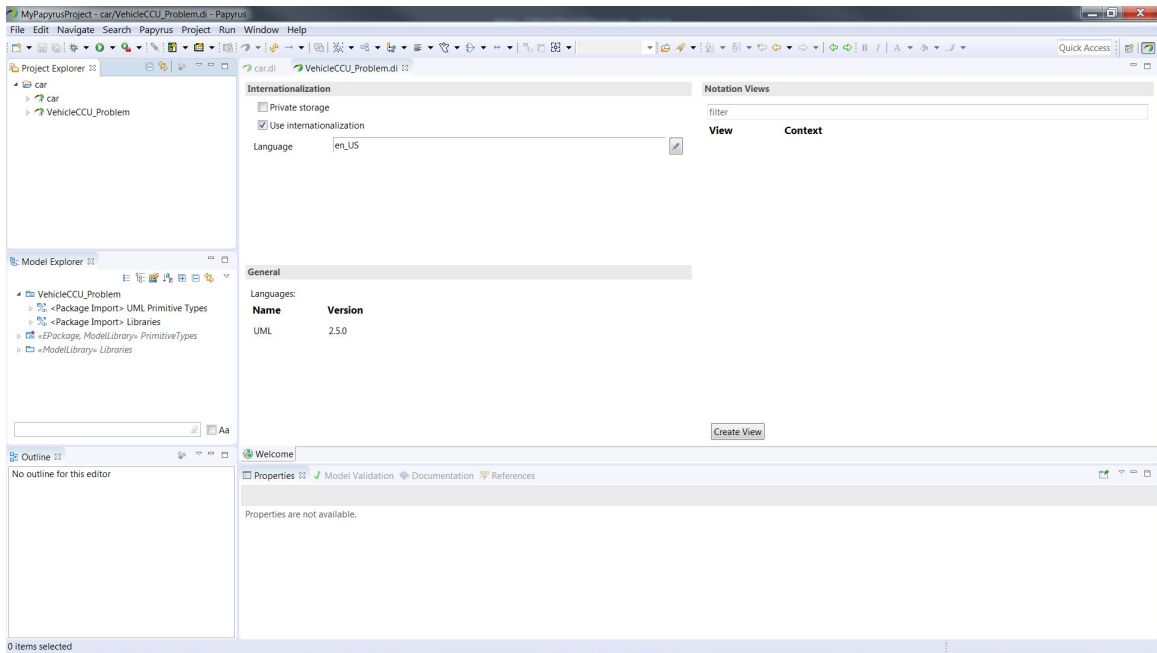


Figure 6: Vehicle CCU Problem Model

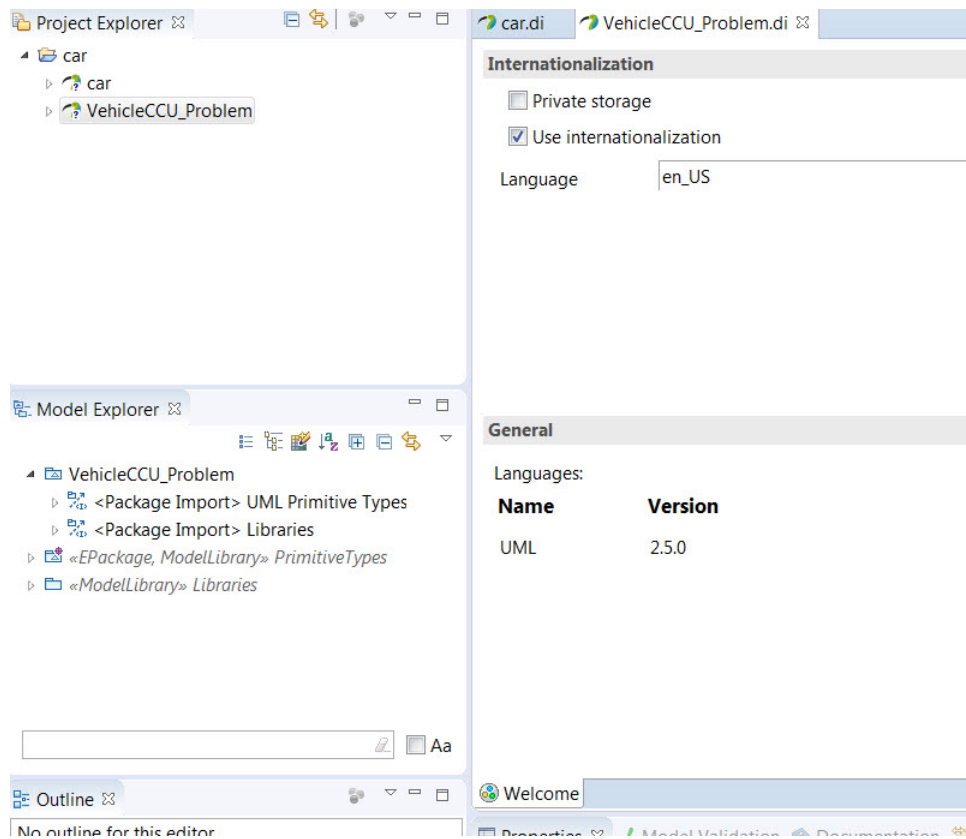


Figure 7: Vehicle CCU in Model Explorer

2.2.2 Step 2. Organize the model for Black-Box analysis.

1. We are now going to organize our model by creating three new packages in our *VehicleCCU_Problem* model and when we are finished it should look like figure 8.

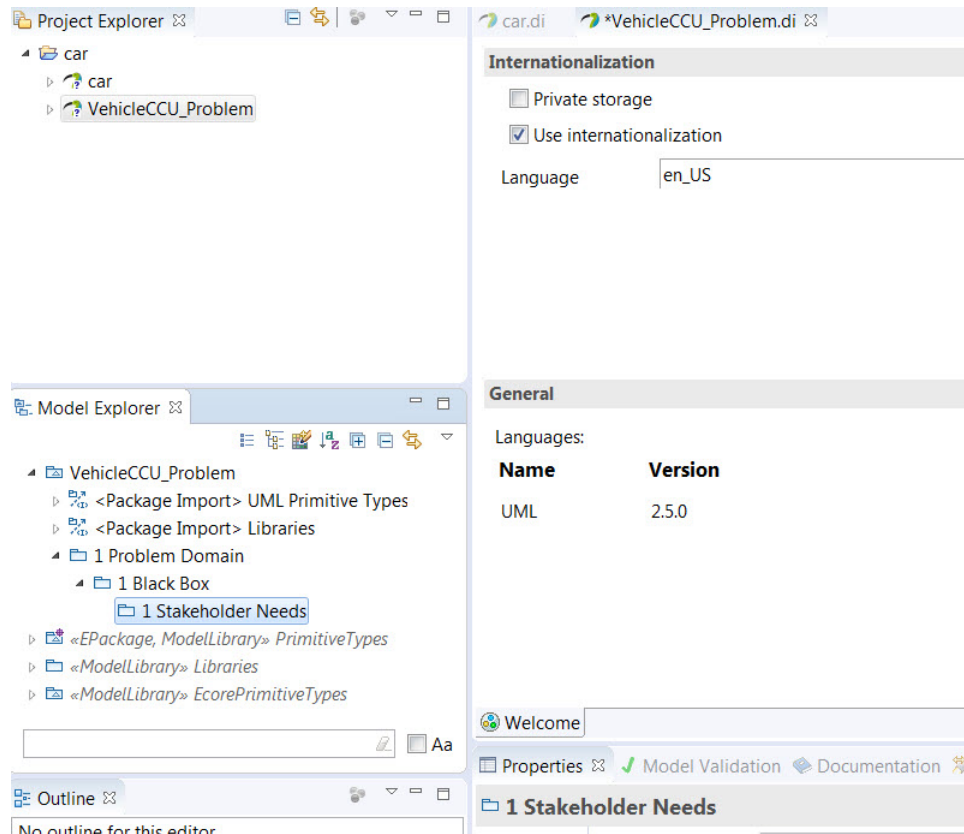


Figure 8: Black Box Packages

2. Right-click on the *VehicleCCU_Problem* top level folder in the *Model Explorer* window and then select **New Child** and then **Package** (figure 9). The new package will appear under the *VehicleCCU_Problem* folder. Name the package *1 Problem Domain*. (Note that if you have already hit enter the package is named with a default name like *Package14* and it can be renamed in the UML properties tab at the bottom of the main window.
3. Now right-click on the *1 Problem Domain* package and follow the same procedure to create a new child package named *1 Black Box*.
4. Now right-click on the *1 Black Box* package and follow the same procedure to create a new child package named *1 Stakeholder Needs*. Your result should look like figure 8. If you make a mistake along the way you can simply delete a package and try again. Ensure that the packages are nested as shown in figure 8 before continuing.

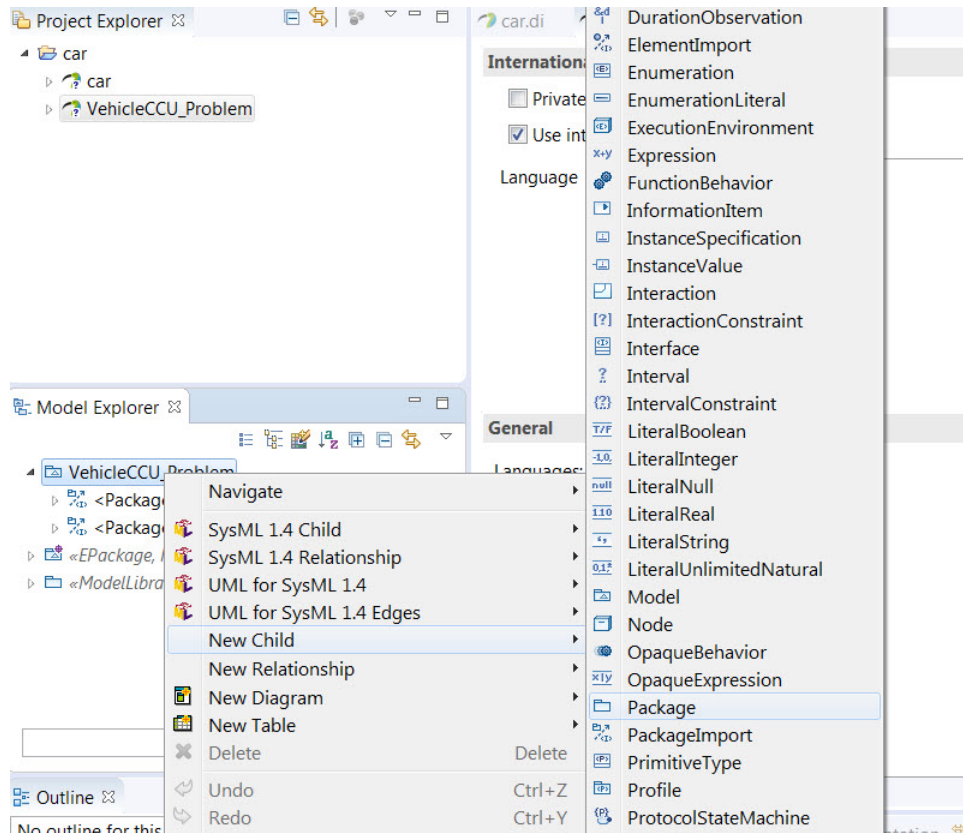


Figure 9: Create a New Package

2.2.3 Step 3. Capturing Stakeholder Needs (Requirements Table)

1. After talking with our stakeholders we will capture our initial list of stakeholder needs for the Vehicle Climate Control Unit using a SysML Requirement Table. There would of course be many requirements but we will just capture a few for illustrative purposes.
2. Right-click on the *1 Stakeholder Needs* package in the *Model Explorer* window and select **New Table > SysML 1.4 Requirement Tree Table** (figure 10). Name the table *Stakeholder Needs*. We want to use the Requirement Tree Table so we can have nested requirements in our requirements table.

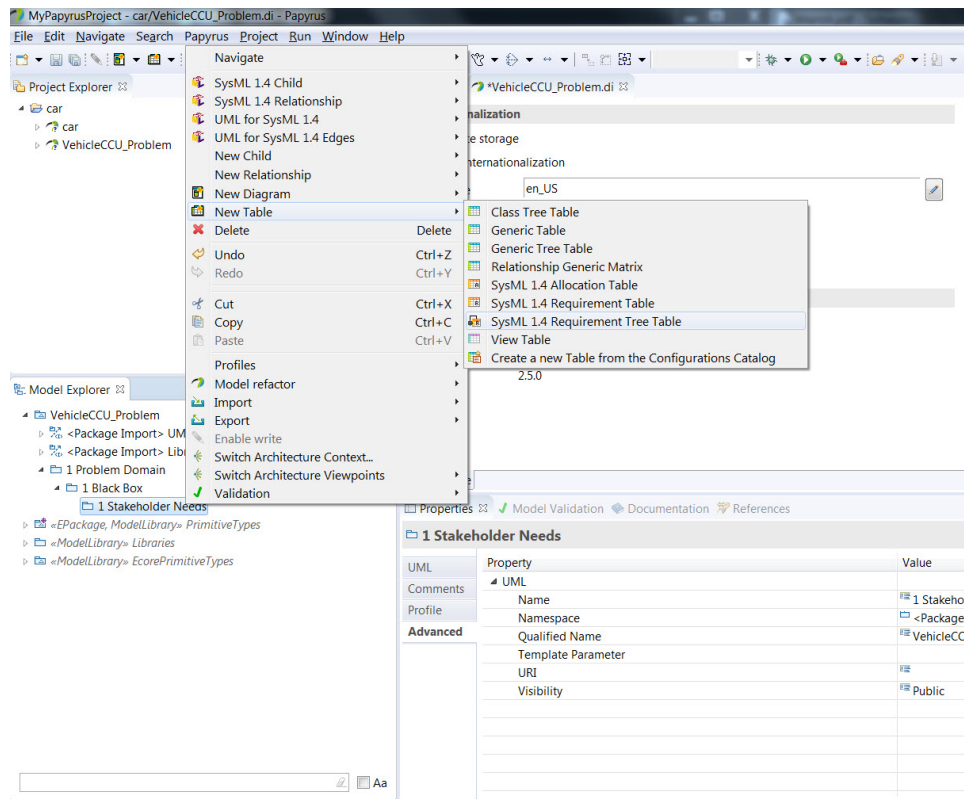


Figure 10: Creating a New Requirements Table

3. You should now see an initial table in the main window with two cells at the top labeled *id* and *text*.
4. First Left-click anywhere in the blank space in the main table window and then Right-click and select **Create SysML 1.4 Element > Create New Requirement** (see figure 11). A new requirement should appear in the table view and a requirement model element should appear underneath the *Stakeholder Needs* table element in the *Model Explorer* window (figure 12).

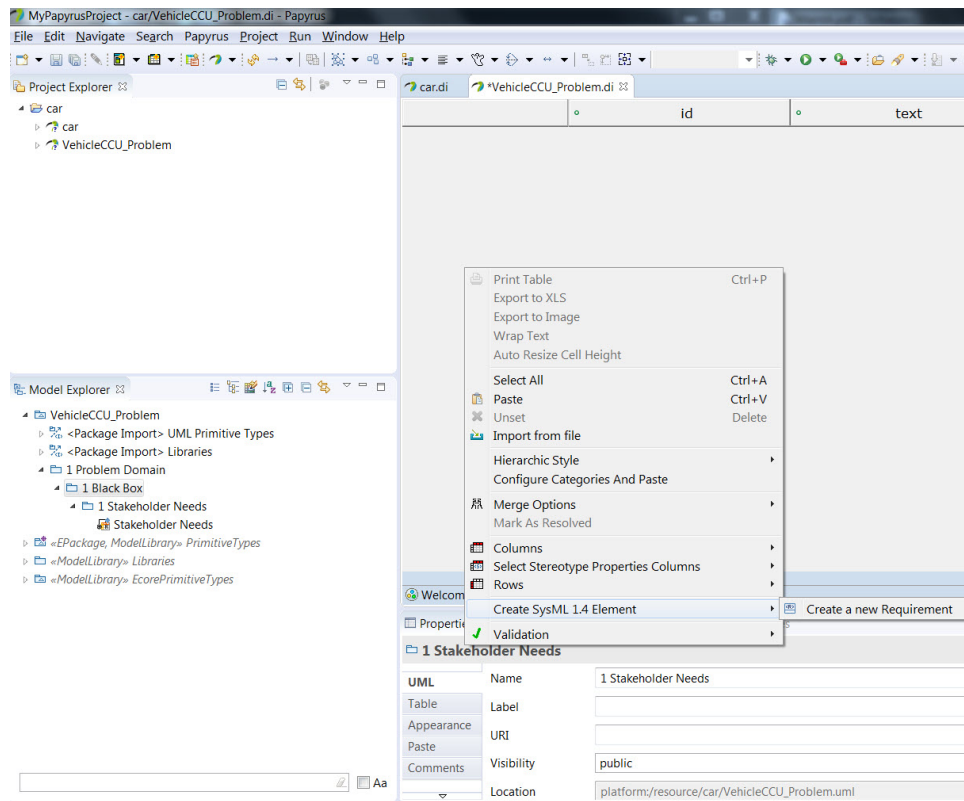


Figure 11: Create New Requirement in Requirement Table

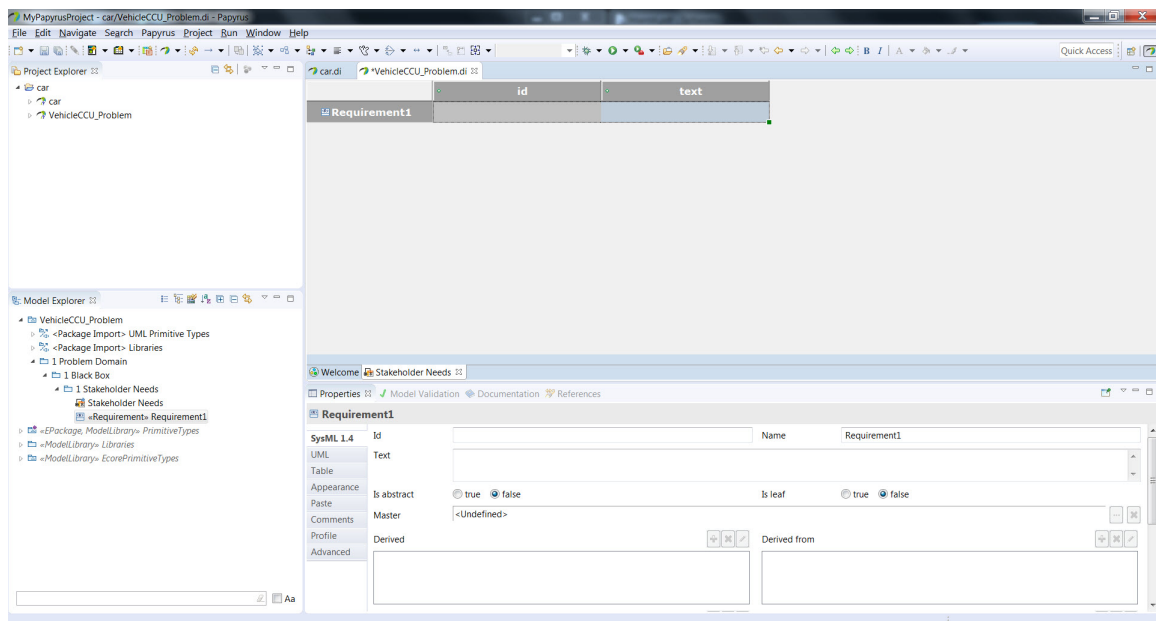


Figure 12: New Requirement in Model

5. Most software like *Papyrus* has multiple ways to accomplish the same task. We can type directly in the requirements table cells to enter the *id* and *text* or we can enter it

in the *Properties Window* in the *SysML 1.4* tab. We are going to use the *SysML 1.4* tab at the bottom of the main screen . To do this Left-click on *Requirement 1* under the *Stakeholder Needs* table element in the *Model Explorer* then select the *SysML 1.4* tab in the *Properties window* (figure 12). You can see the properties in this tab for *Id*, *Name*, and *Text*.

6. Type *SN-1* in the *Id* field and *SN-1 User Needs* in the *Name* field. We will not enter anything in the *Text* field as this will be a heading for our user needs (figure 13).

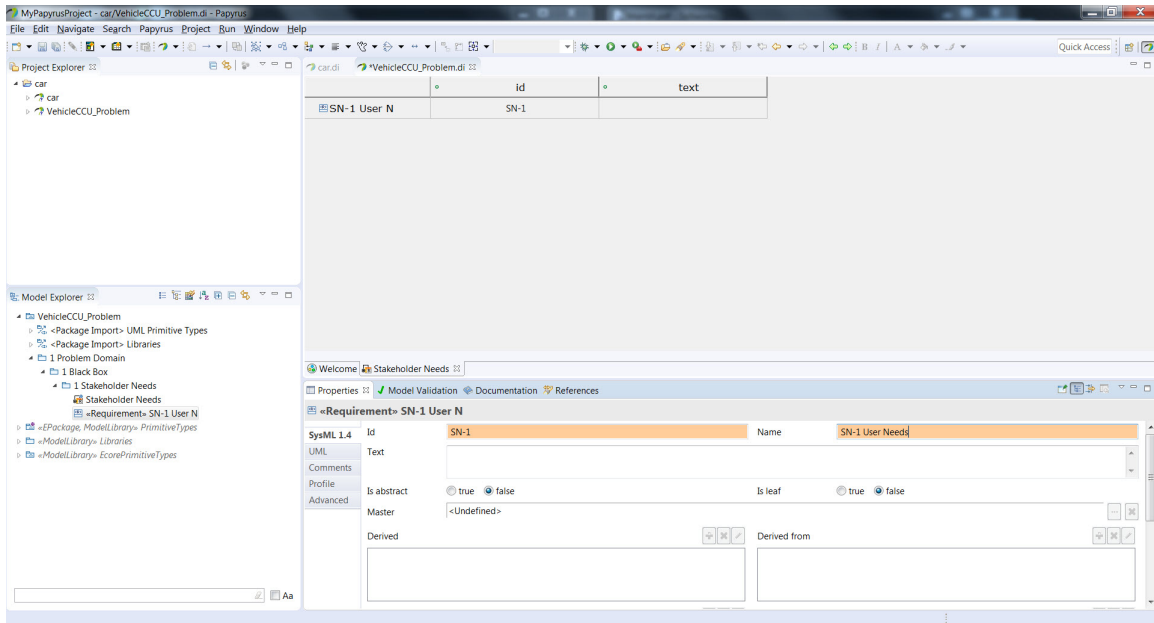


Figure 13: User Needs

7. Continue adding requirements and populating the *Id*, *Name*, and *Text* fields until the requirements table looks like figure 14.

	id	text
SN-1 User Needs	SN-1	
SN-1.1 Setting Temperature	SN-1.1	It must be possible to set and maintain desired temperature in the car.
SN-1.2 Heat and Cool Modes	SN-1.2	Unit shall heat and cool.
SN-1.3 Noise Level	SN-1.3	Climate control unit in max mode shall not be louder than the engine.
SN-1.4 Climate Control Mass	SN-1.4	Mass of the unit shall not exceed 1 percent of the total car mass.

Figure 14: User Needs Requirements Table

8. In the *Model View* window Left-click on the requirement *SN-1.1 Setting Temperature* requirement then hold down the Control key and select the last requirement *SN-1.4 Climate Control Mass*. Drag these requirements and drop them on the *SN-1 User Needs* to make these requirements child requirements of our heading requirement. Your screen should look like figure 15 with the requirements nested under *SN-1 User Needs* in the *Model View* window and a tree structure in the table view.

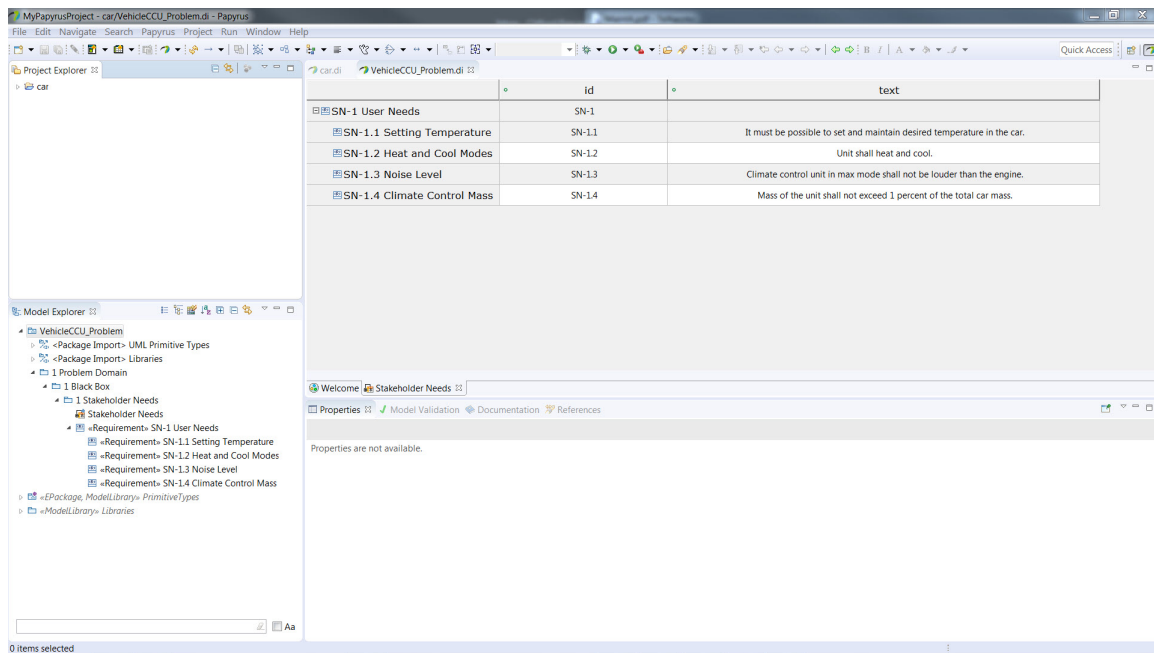


Figure 15: Requirements Tree

2.2.4 Step 4. Initial System Context

The operating environment or System Context defines the external view of the system where the Sol is treated as a black box. It introduces elements that interact with the system but are not owned by the system. During this initial phase of system context definition one or more system contexts are defined and elements that participate in each context are identified. Several sources recommend using blocks rather than actors to define humans for the following reasons:

- A block is treated as a part of the system context. An actor is not.
- A block can be marked as external. An actor cannot.
- A block can have its behavior defined. An actor cannot.

Let's start modeling the Initial System Context.

1. Set up a new package under the *1 Black Box* package named *3 System Context* by Right-clicking on the *1 Black Box* package in the *Model Explorer* window and selecting **New Child** and then **Package**. Name the new package *3 System Context* (figure 16).

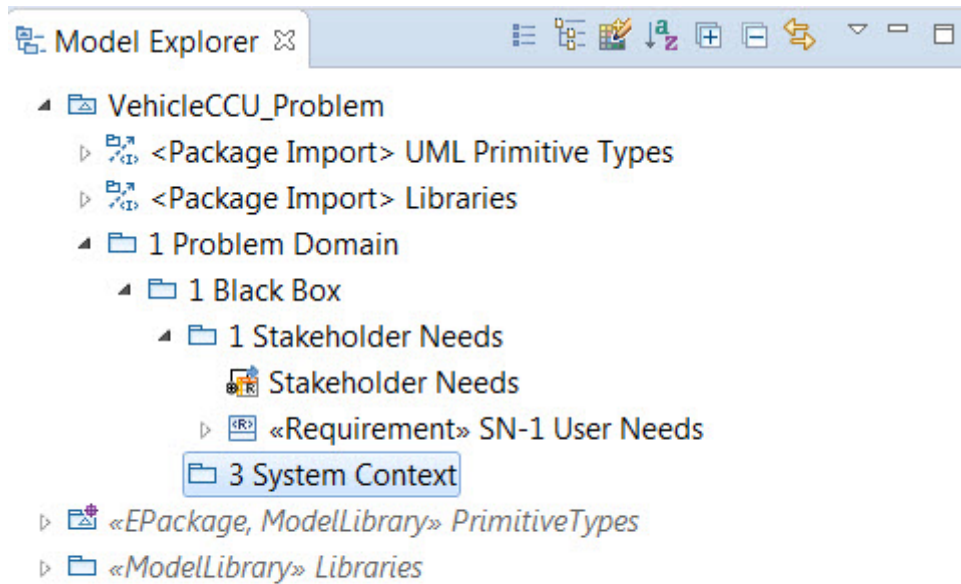


Figure 16: System Context package

2. Now we will create a block that captures the vehicle in use system context. Right-click on the 3 *System Context* package just created and select **SysML 1.4 Child** and then **Block** (figure 17). Name the block *Vehicle In Use*. Your model explorer should now look like figure 18.

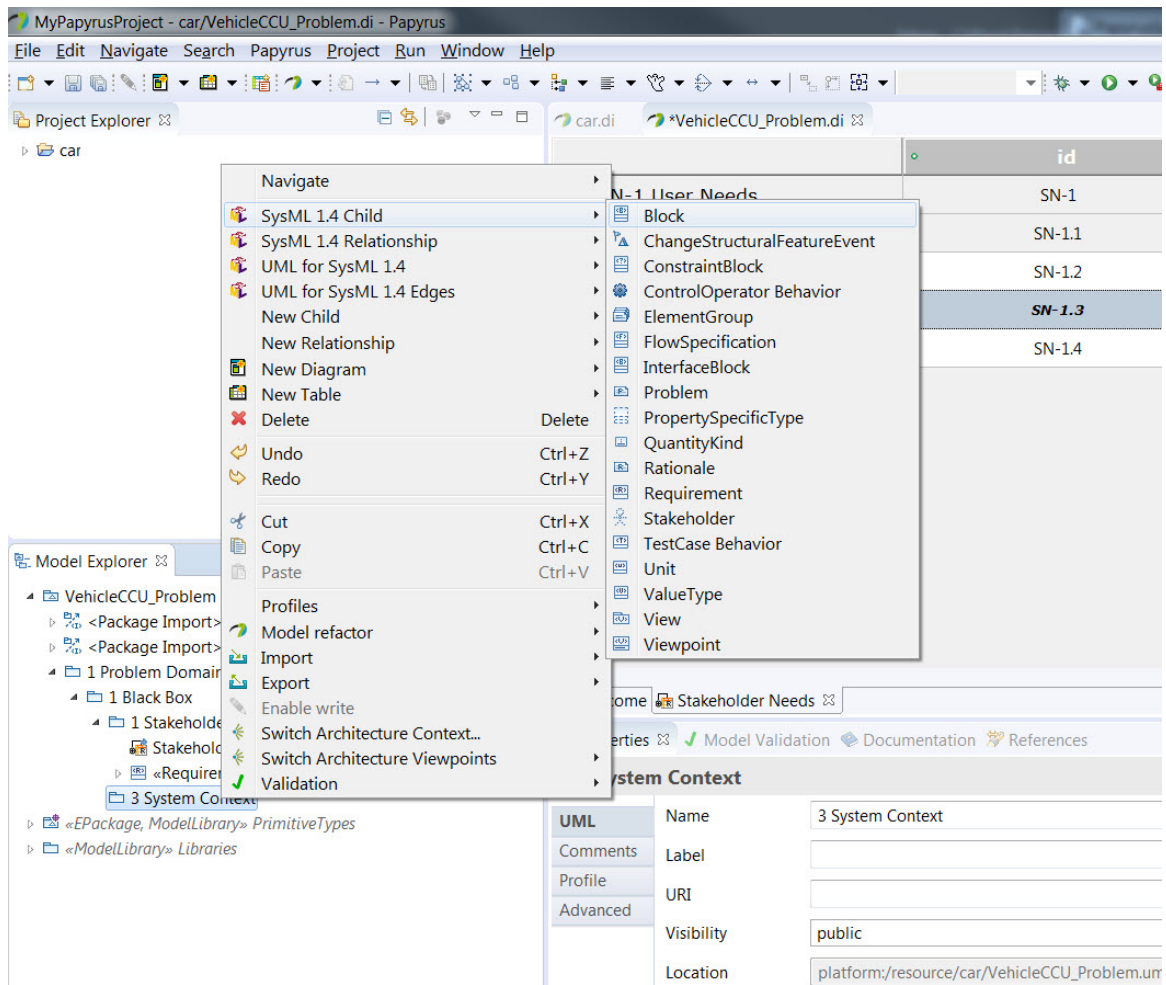


Figure 17: Create New Block

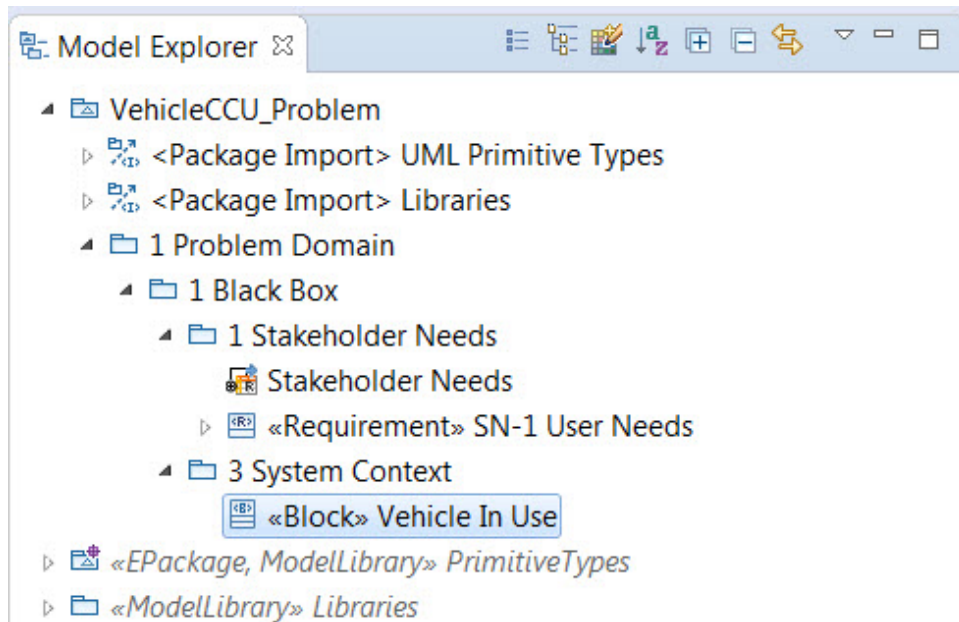


Figure 18: Vehicle In Use Block

- To specify participants of the Vehicle In Use context we will create a SysML Internal Block Diagram (IBD). Right-click on the *Vehicle In Use* block just created and select **New Diagram** then **SysML 1.4 Internal Block Diagram** (figure 19). Name the new diagram *Vehicle In Use*, the same as the block. You should see the diagram in the main window (figure 20). If you do not, simply Double-click on the *Vehicle In Use* internal block diagram in the *Model Explorer* window.

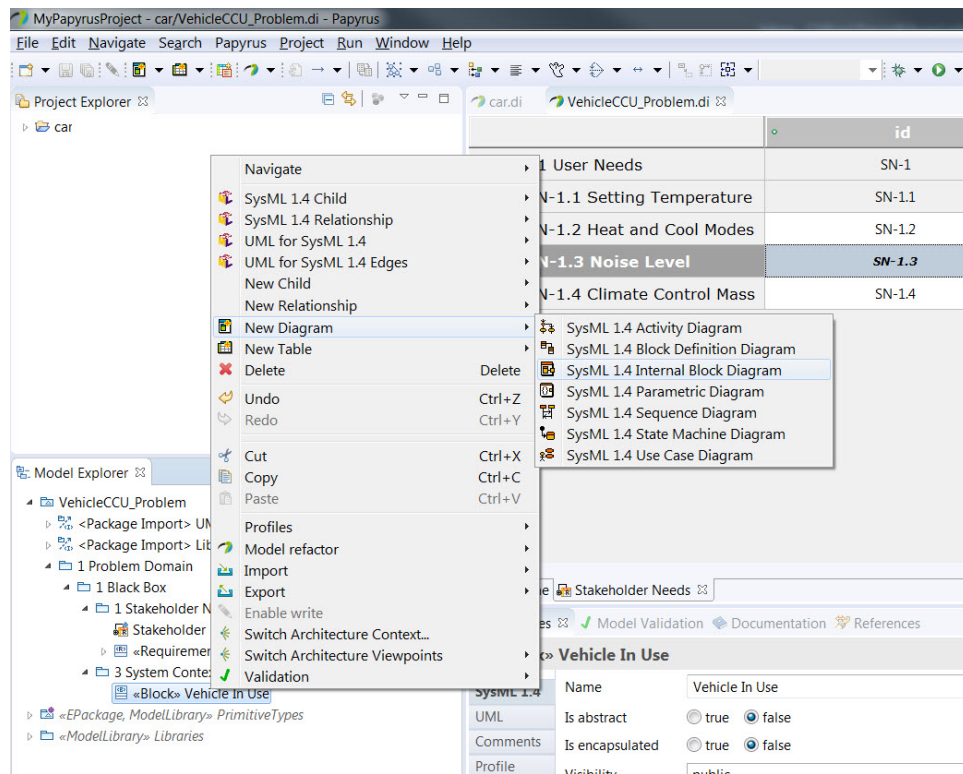


Figure 19: Create Internal Block Diagram

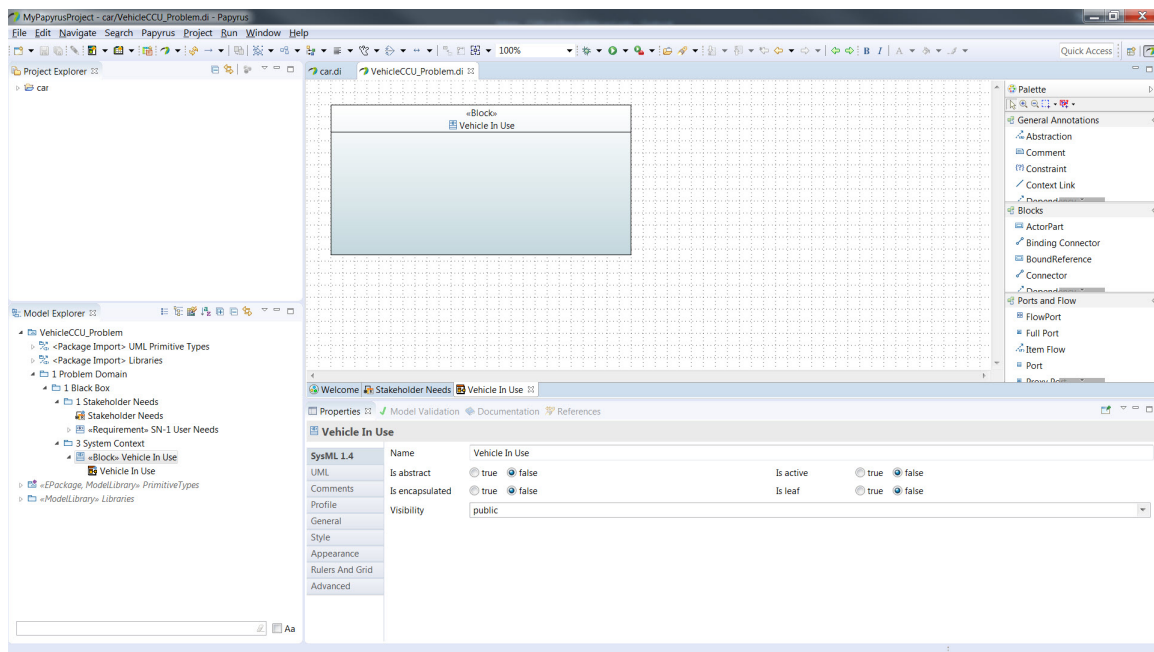


Figure 20: Vehicle In Use Internal Block Diagram

- Our analysis of stakeholder needs indicates that the Vehicle In Use system context includes the following participants: the vehicle Climate Control Unit, Vehicle Occu-

part, and an Energy Supply of some kind to power everything. The participants can be captured as part properties in the IBD. Open the *Vehicle In Use* IBD you just created if not already open. Use the tool palette (If the tool palette is not visible click on the small arrow on the right side of the main window vertical scroll bar to display it) on the right and select **Part** from the **Blocks** diagram palette . Now Left-click anywhere inside the *Vehicle In Use* diagram in the main window and the *Type Initialization* window will appear (figure 21).

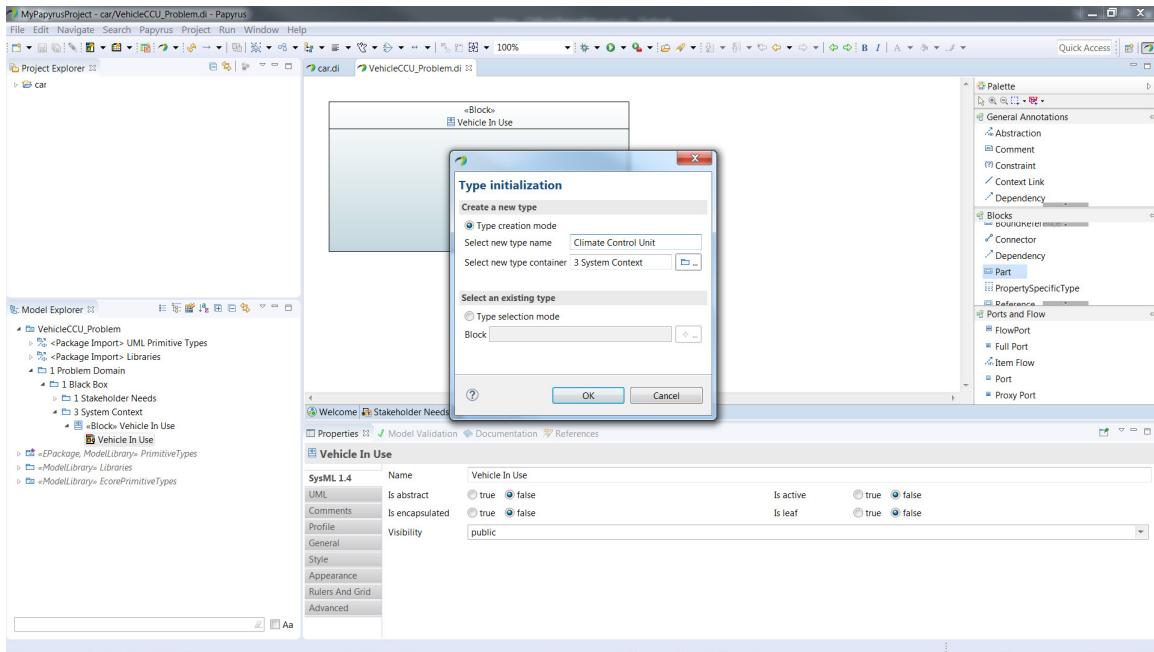


Figure 21: Part Type Initialization Window

5. In the *Type Initialization* window ensure that the *Type creation mode* radio button is selected to create a new type. In the *Select new type name box* type *Climate Control Unit* and press *Enter* twice. Your model should look like figure 22.

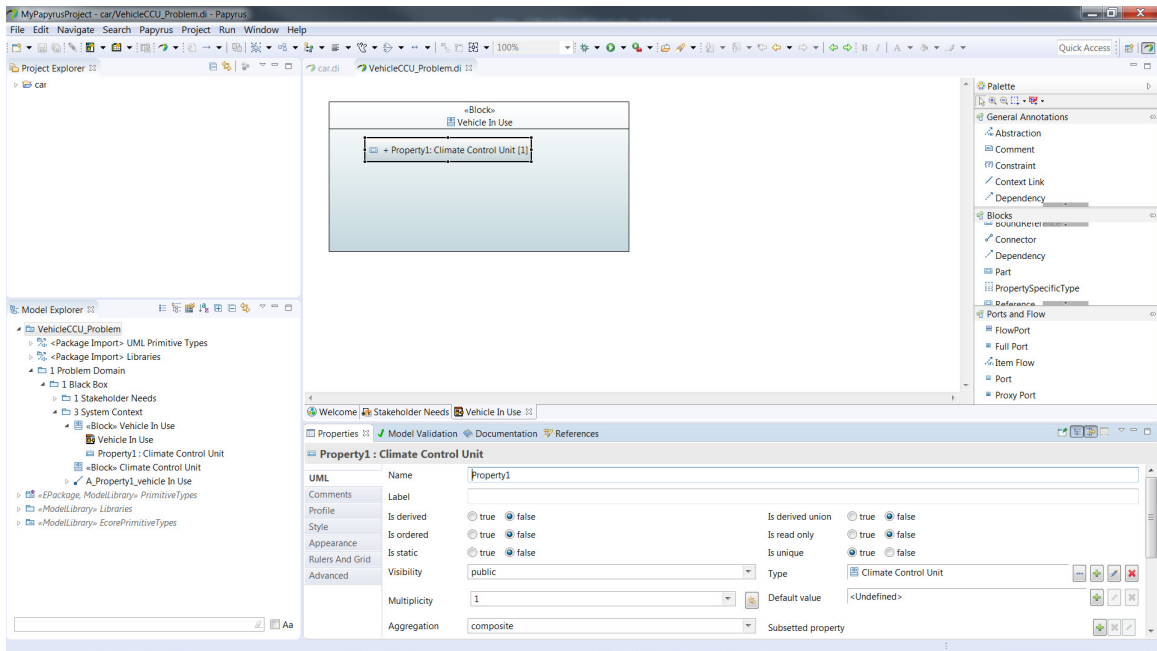


Figure 22: Part Type Initialization Window

- Let's make this new block a little cleaner by not displaying the *Property1* name in front of the colon in the *Climate Control Unit* part in the diagram. Select the *Climate Control Unit* in the diagram. In the *Properties* window at the bottom of the screen select the *UML* tab and place your cursor in the *Label* field. Type a single Space with the spacebar and hit *Enter*. Your *Climate Control Unit* should now look like figure 23.

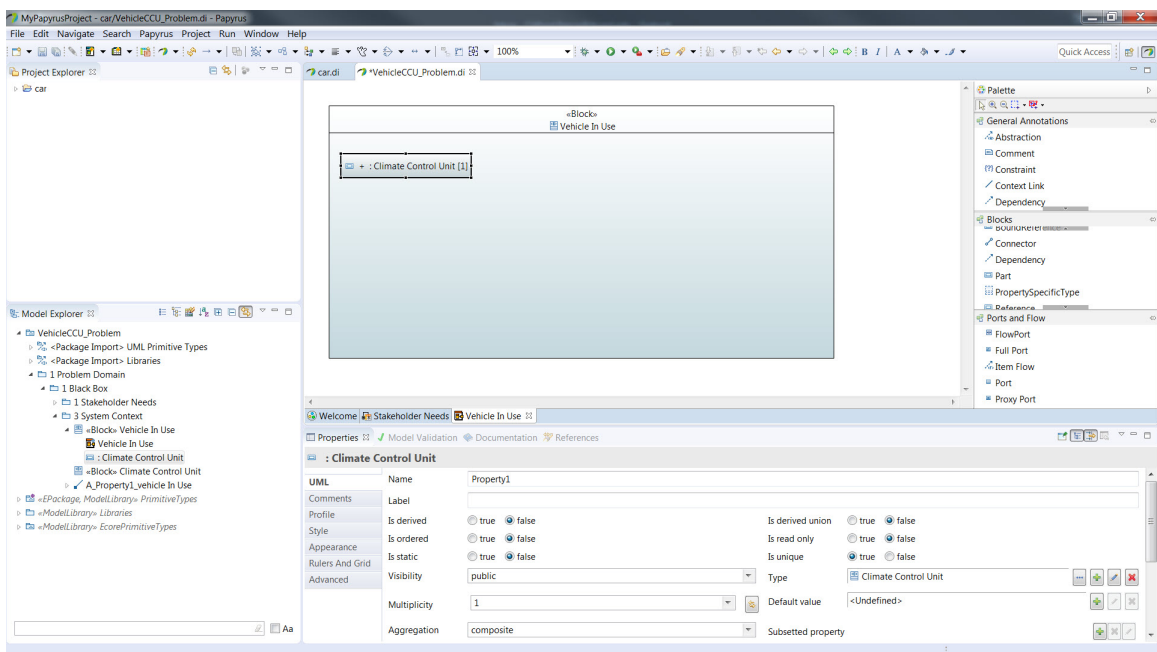


Figure 23: Climate Control Unit with Label Removed

7. Create two more parts; one called *Vehicle Occupant* and one called *Energy Supply*. When finished your diagram should look like figure 24.

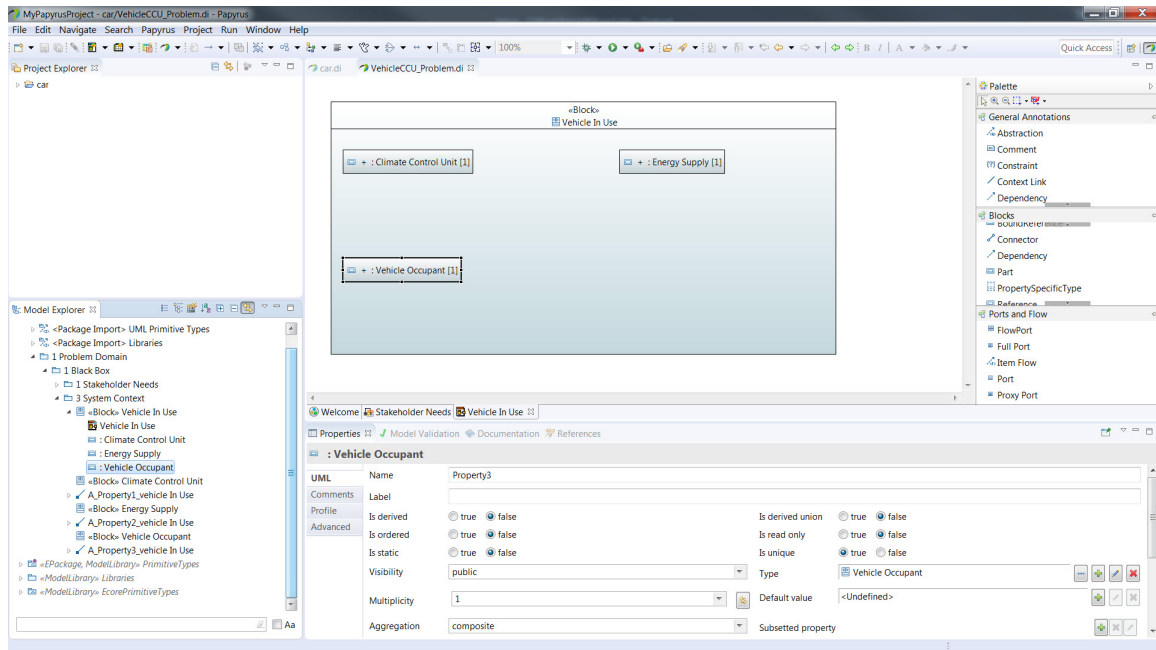


Figure 24: Vehicle In Use IBD

2.2.5 Step 5. Use Case and Activity Diagram

In this section, the functional stakeholder needs will be refined with use cases and use case scenarios. Again we will be using blocks rather than actors (see section 2.2.4).

1. Set up a new package under the *1 Black Box* package named *2 Use Cases* by Right-clicking on the *1 Black Box* package in the *Model Explorer* window and selecting **New Child** and then **Package**. Name the new package *2 Use Cases* (figure 25).

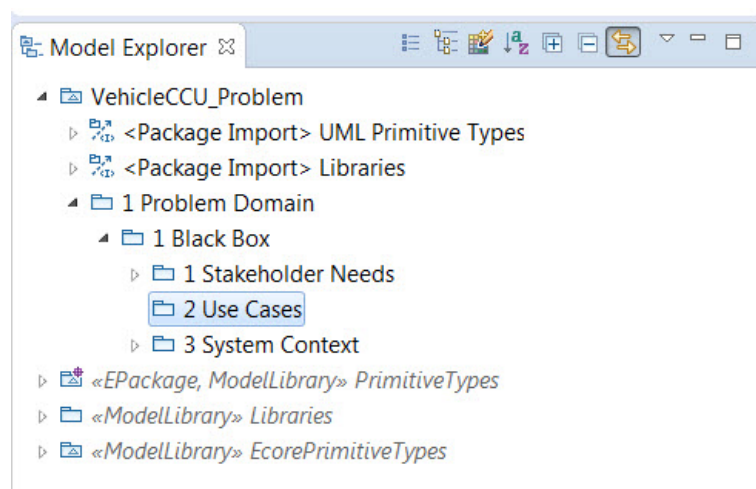


Figure 25: Use Case Package

2. Now that we have created the use case package we need to create a Use Case Diagram. In the *Model Explorer* window, Right-click on the *2 Use Cases* package that you created in the previous step and select *New Diagram* and then *SysML 1.4 Use Case Diagram* (figure 26). Name the diagram *Use Cases of Vehicle In Use SC*. The new use case diagram will appear under your *2 Use Cases* package and a new blank use case diagram should appear in the main window (figure 27).

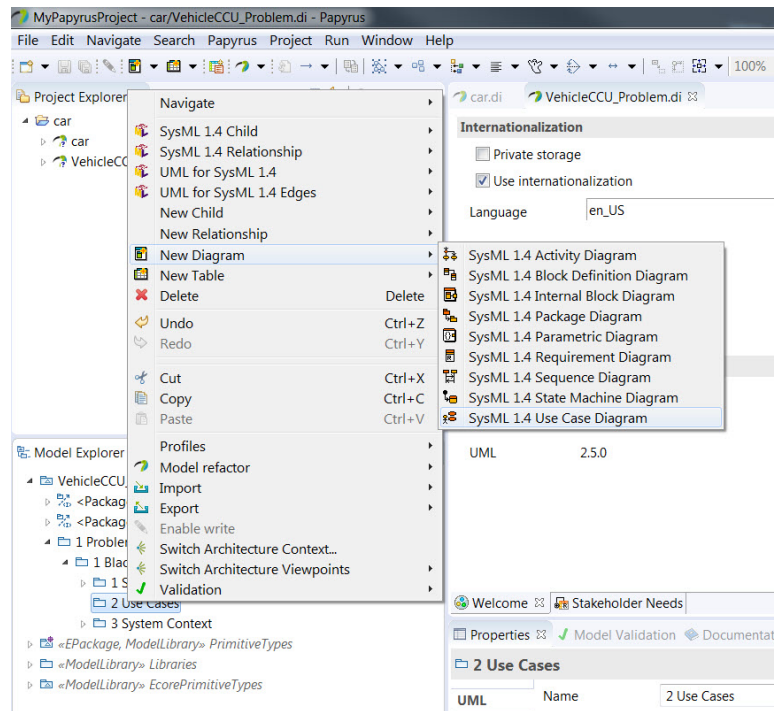


Figure 26: Use Case Diagram Creation

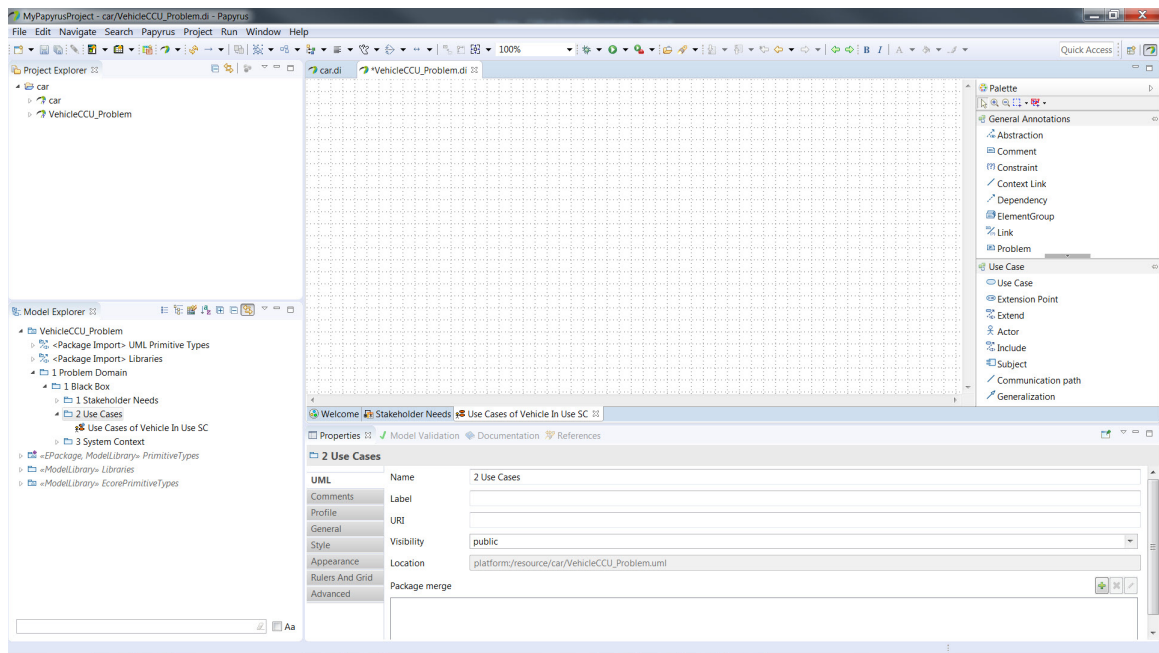


Figure 27: New Use Case Diagram

3. In the *Model Explorer* window, click on the arrow to the left of the *3 System Context* package to expand its contents (figure 28).

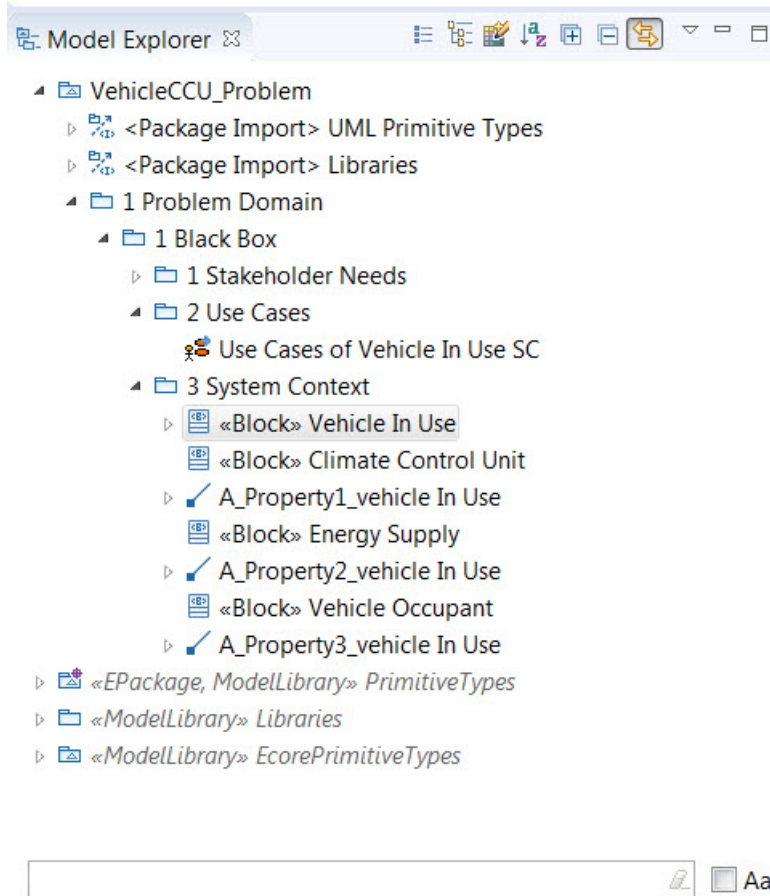


Figure 28: Expand the System Context package

4. Click and Drag the *Vehicle In Use* block onto the *Use Cases of Vehicle In Use* use case diagram in the main window. A copy of the block should appear in the use case diagram (figure 29).

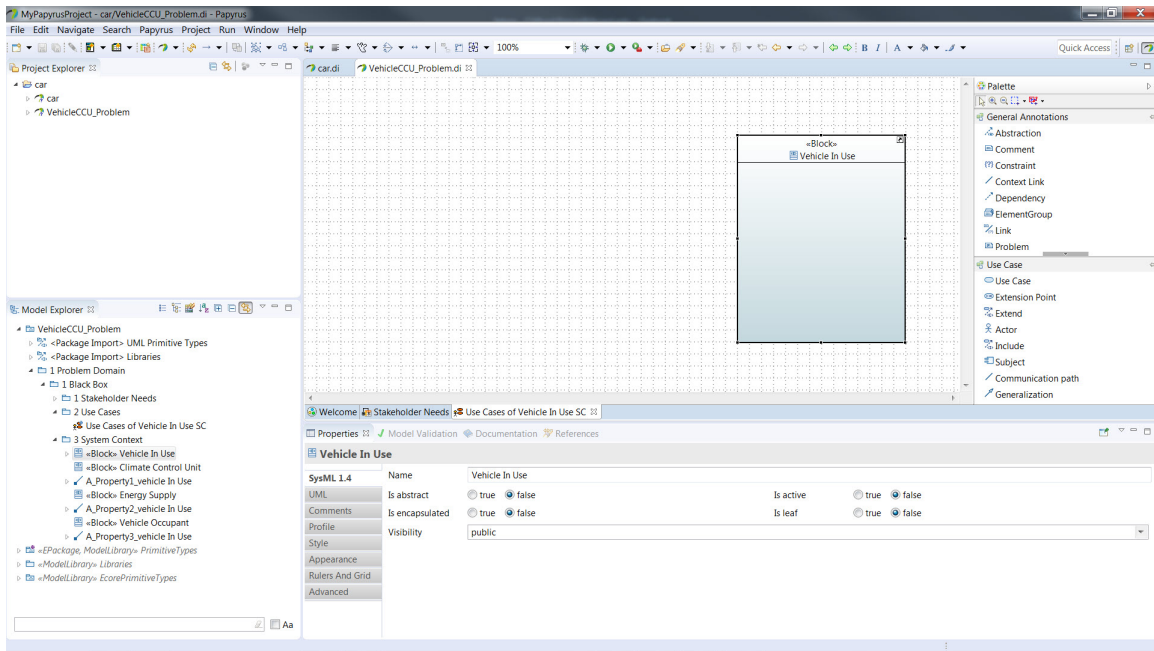


Figure 29: Vehicle In Use block in Use Case Diagram

- Click and Drag the *Vehicle Occupant* block onto the *Use Cases of Vehicle In Use* use case diagram in the main window. A copy of the block should appear in the use case diagram (figure 30).

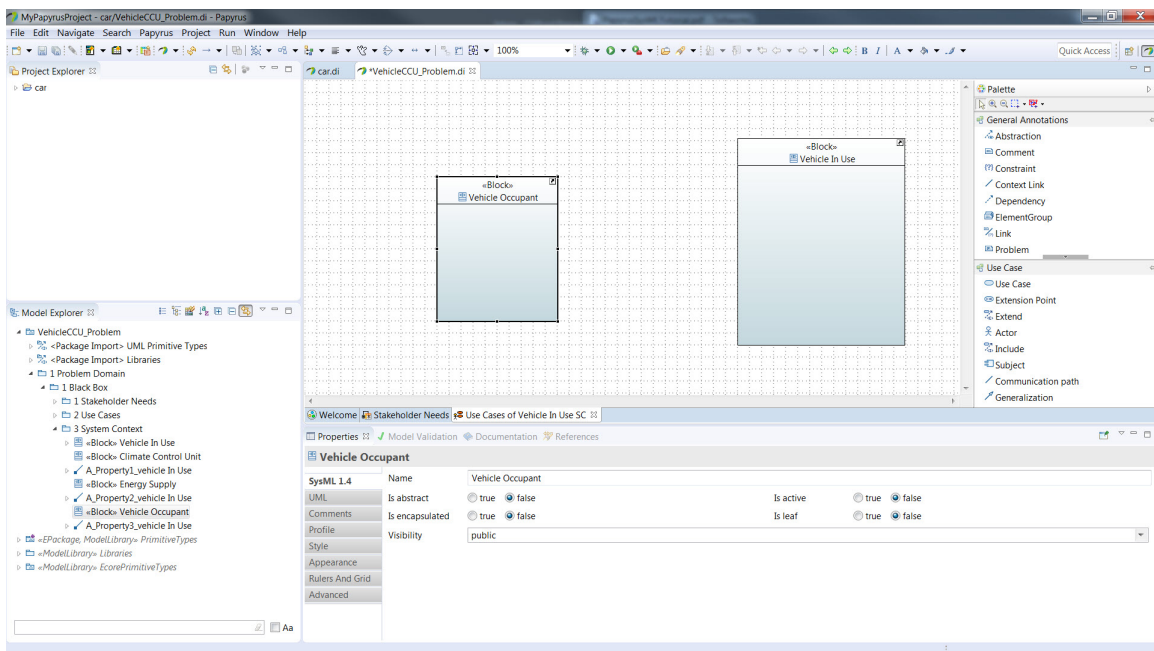


Figure 30: Vehicle Occupant block in Use Case Diagram

6. In our analysis of stakeholder needs we have identified the *Feel Comfortable Temperature* use case performed in the *Vehicle In Use* system context. We will now capture this use case in the *Use Cases of Vehicle In Use* SC diagram. The tool palette on the right should be displaying the Use Case tool palette. Left-click on the *Use Case* item (figure 31) and then Left-click anywhere in the *Vehicle In Use* block. A use case is created in this block. Name this use case *Feel Comfortable Temperature* and press Enter. Your screen should look like (figure 32).

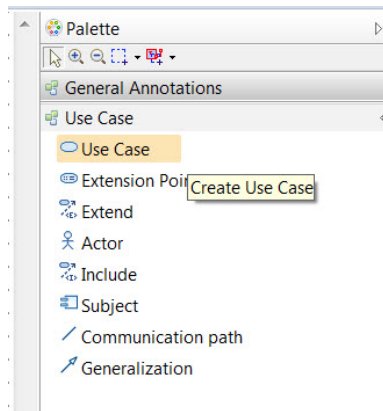


Figure 31: Use Case Tool Palette

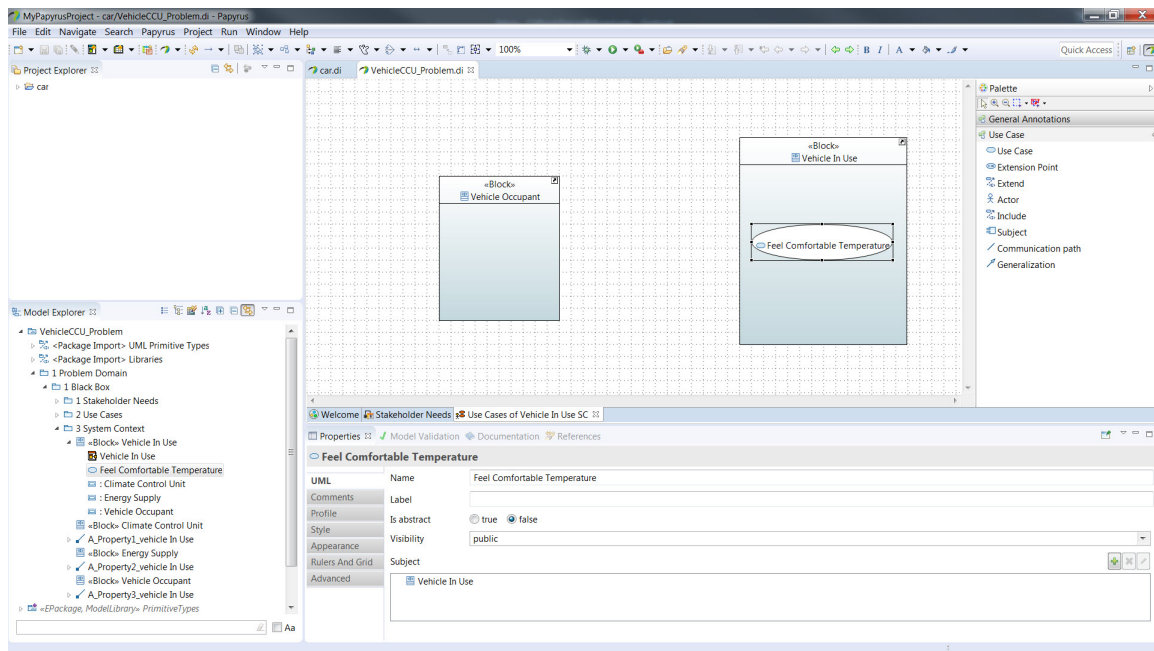


Figure 32: Vehicle Occupant block in Use Case Diagram

7. Now we will create an association between the *Feel Comfortable Temperature* use case and the *Vehicle Occupant* block (recall that we are using a block here to represent the traditional UML actor for the reasons stated earlier in section 2.2.4). Left-

click the *Feel Comfortable Temperature* use case in the diagram. Now Left-click the *Communication Path* item in the *Use Case* tool palette (figure 33). Now Left-click on the *Feel Comfortable Temperature* use case and then Left-click on the *Vehicle Occupant* block in the diagram. An association should be created and your model should look like (figure 34).

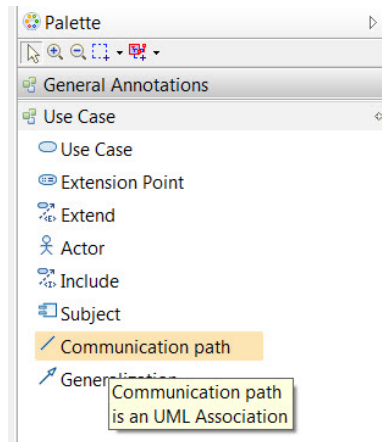


Figure 33: Use Case Tool Palette

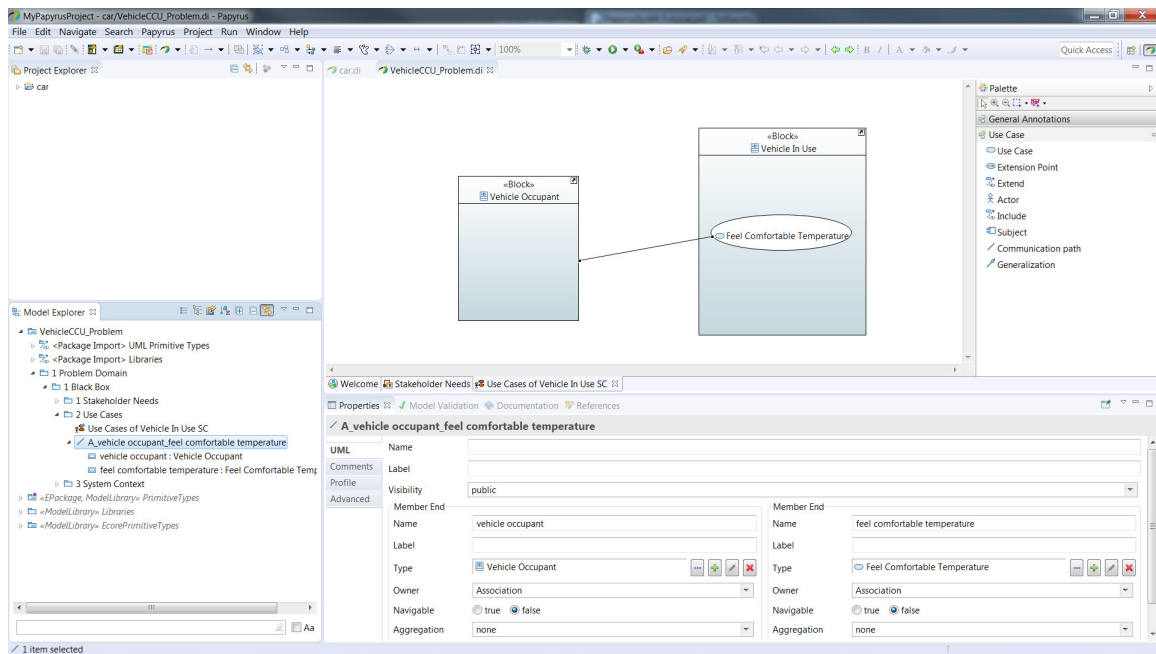


Figure 34: Association Created

8. Now we will elaborate the *Feel Comfortable Temperature* use case as an activity diagram. First let's move the *Feel Comfortable Temperature* use case from the 3 *System Context* package and place it under the 2 *Use Cases* package. To do this, expand the 3 *System Context* package view in the *Model Explorer* window. Now

Click and Drag the *Feel Comfortable Temperature* use case and Drop it on the *2 Use Cases* package in the *Model Explorer* window. The use case is now a child of the *2 Use Cases* package (figure 35).

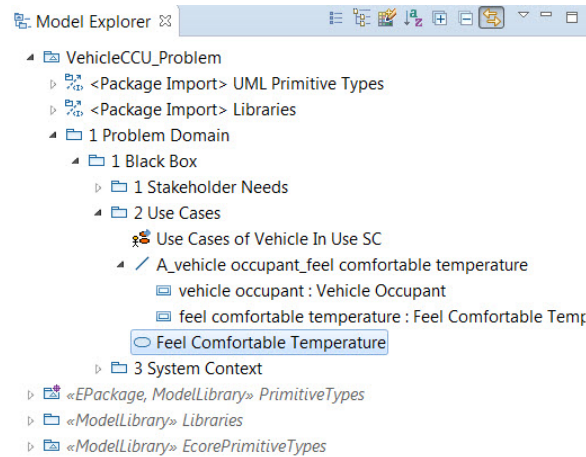


Figure 35: *Feel Comfortable Temperature* as a child of *2 Use Cases*

- Now we will create an activity diagram for our use case. Right-click on the *Feel Comfortable Temperature* use case and then select **New Diagram** and then **SysML 1.4 Activity Diagram** (figure 36). Name the diagram *Feel Comfortable Temperature* (the same name as the use case). Your screen should look like figure 37. We now have the *Feel Comfortable Temperature* activity diagram as a child of the *Feel Comfortable Temperature* use case in the *Model Explorer* window.

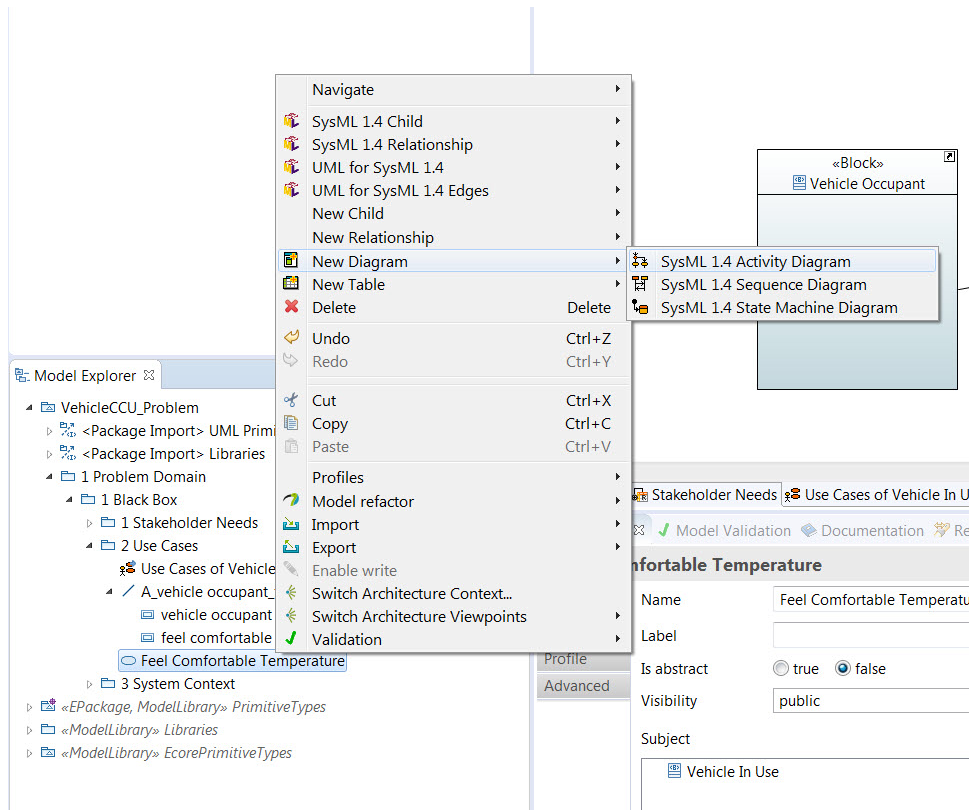


Figure 36: Create New Activity Diagram

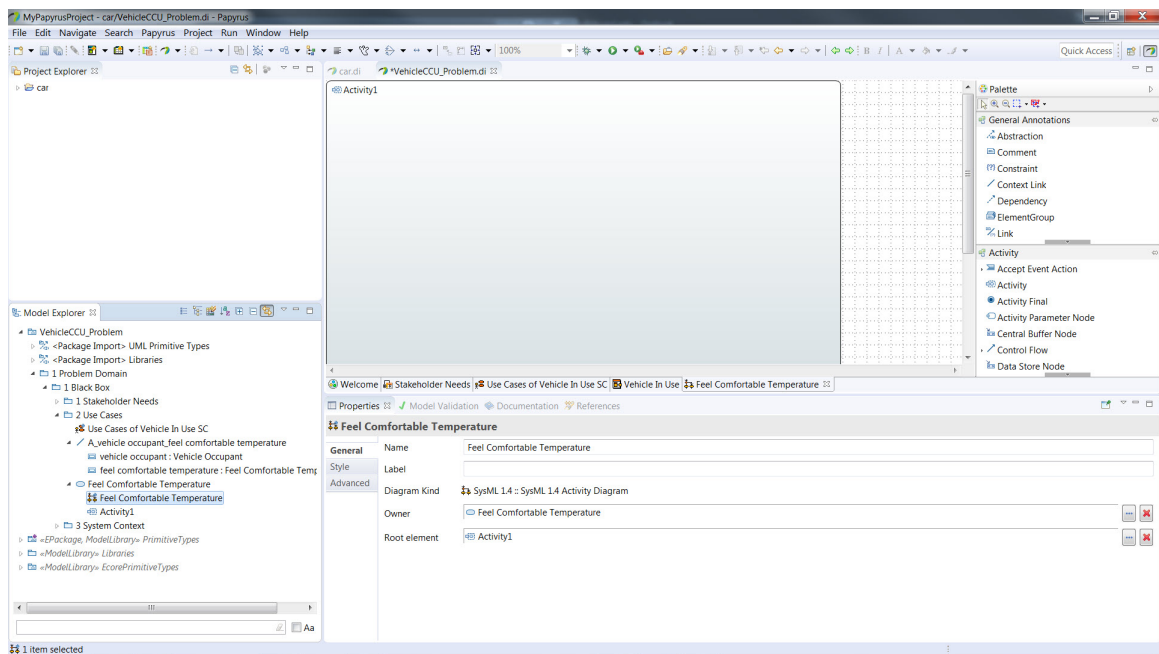


Figure 37: *Feel Comfortable Temperature* Activity Diagram

10. We could start diagramming our activities in the activity diagram first and then allocate the activities to participants using swimlanes. This is a little cumbersome in Papyrus as it involves dragging the activity boxes and connectors onto the swimlanes and some effort to make your diagram “pretty” again. If we know the participants in the activity we can go ahead and create the participant swimlanes first and then populate the activities. That is the approach we will follow. In the *Model Explorer* window Right-click on the *Activity 1* then select **SysML 1.4 Child** and then **AllocateActivityPartition** (figure 38). Name this partition *Vehicle Occupant* (figure 39).

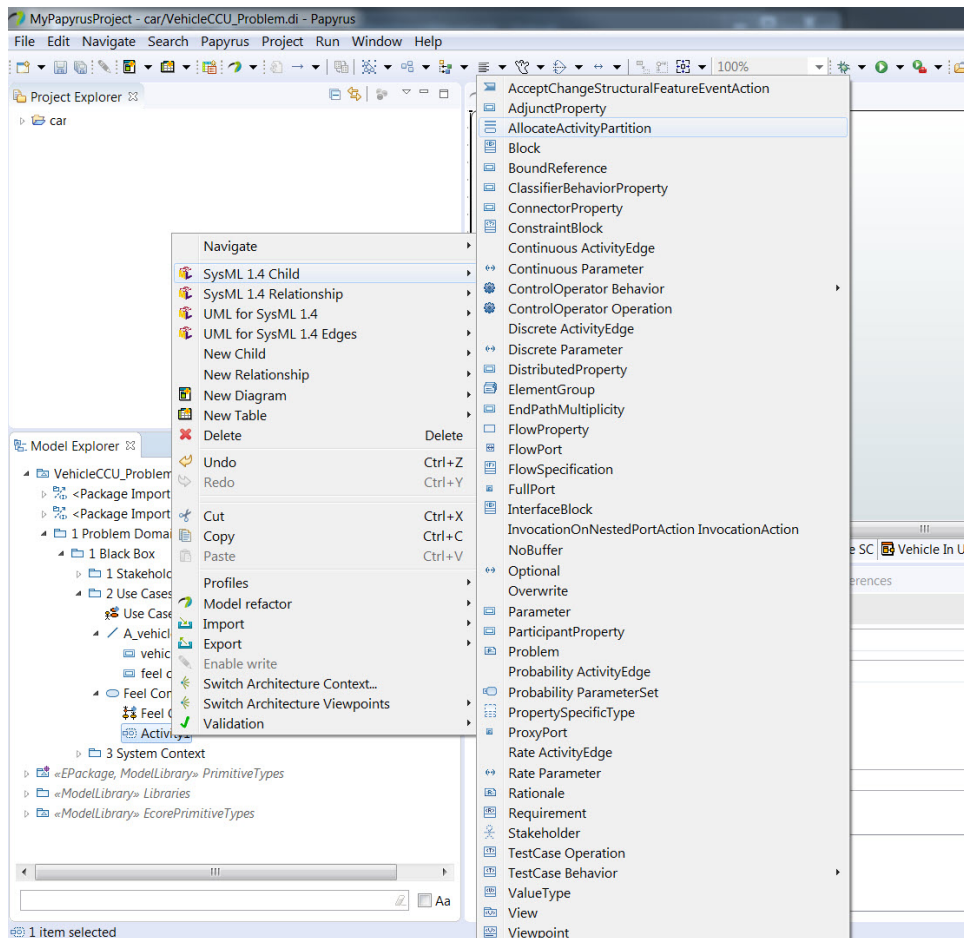


Figure 38: New Activity Partition (Swimlane)

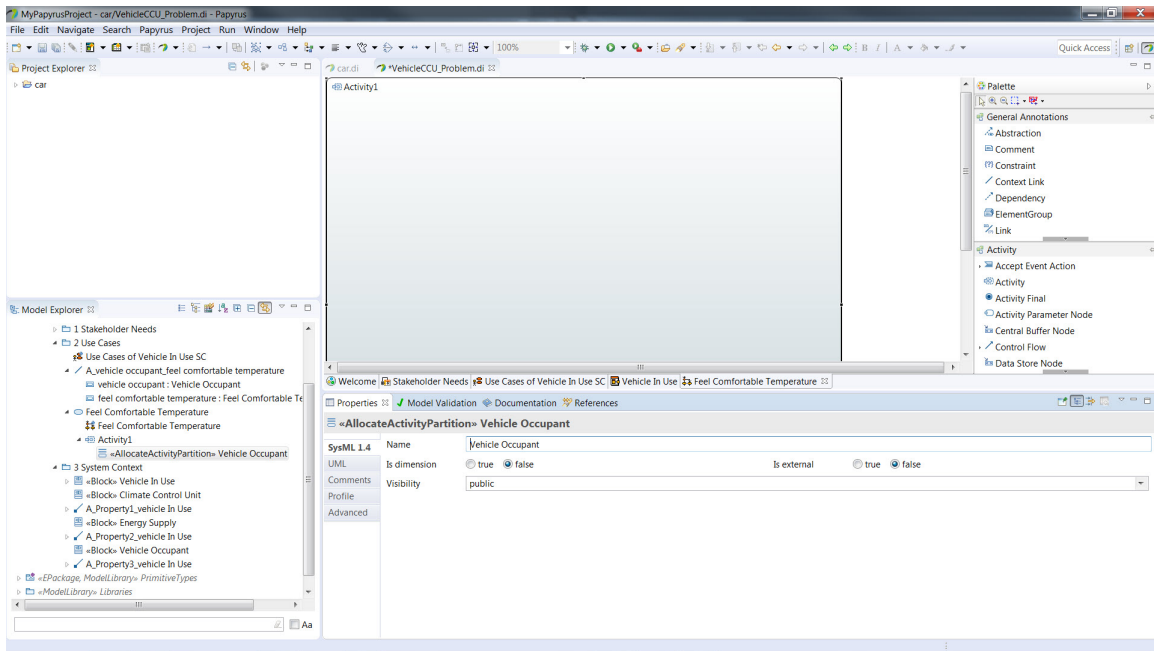


Figure 39: Vehicle Occupant Activity Partition Creation

11. Although we have named our new activity partition *Vehicle Occupant* it does not truly represent the vehicle occupant in our use case until we type it by the *Vehicle Occupant* block. We do this as follows. First ensure that the *Vehicle Occupant* activity partition is selected in the *Model Explorer* window and you see its properties in the **Properties** tab in the bottom center of the screen (figure 39). Now click on **Advanced** tab and look for the **Represents** property and Double-click in the **Value** cell and a pop-up element selection should appear (figure 40). Expand 3 *System Context* package and select the *Vehicle Occupant* and then select **OK**. The value of the **Represents** property should look like figure 41. Our activity partition is now typed by the *Vehicle Occupant* block.

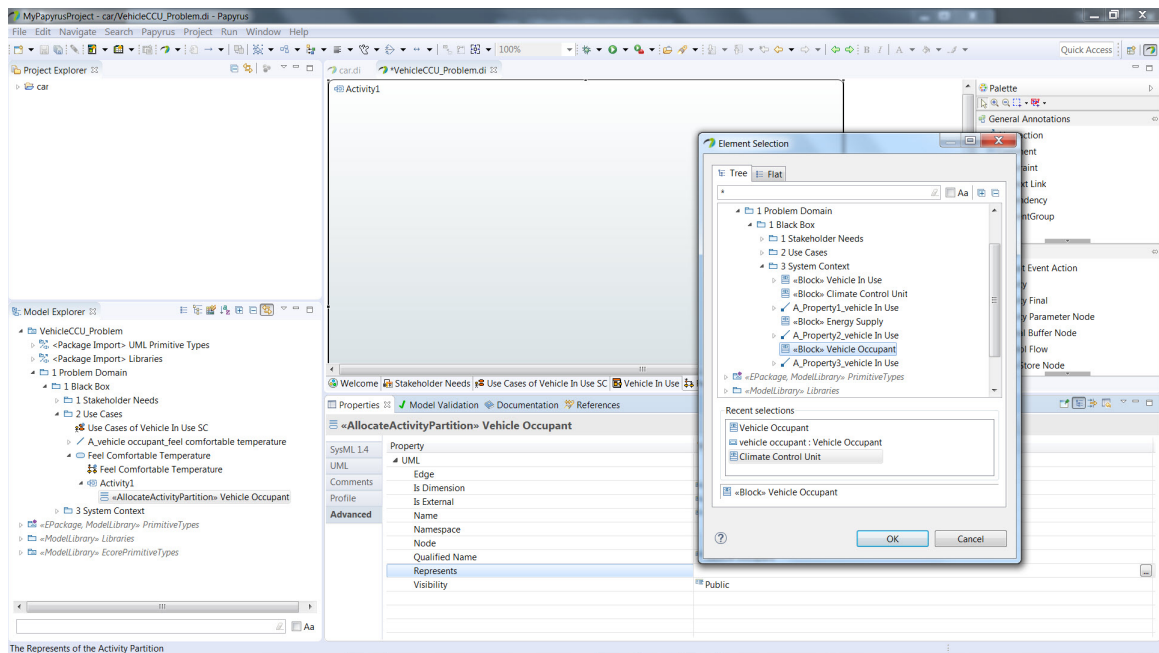


Figure 40: Vehicle Occupant Activity Partition Properties

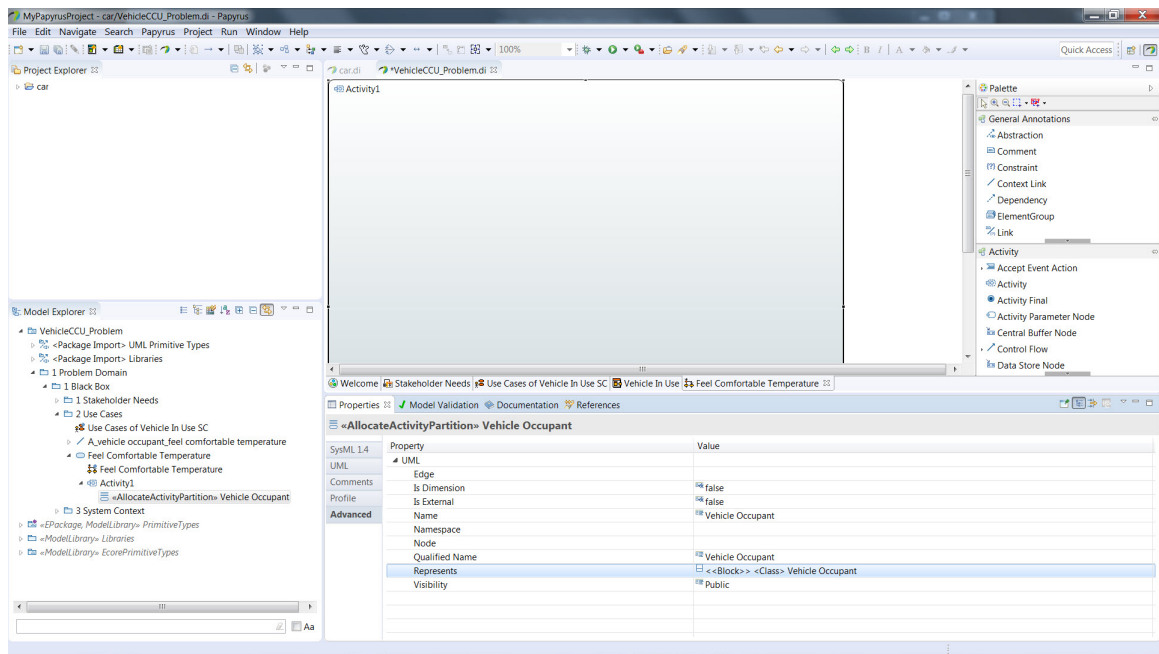


Figure 41: Vehicle Occupant Activity Partition Typed by Vehicle Occupant Block

- Now we need to add the activity partition to our activity diagram. In the *Model Explorer* window Click on the *Vehicle Occupant* activity partition that we created in the previous step and drag it to the activity diagram in the main window. You can adjust the size of the partition to make it larger (figure 42).

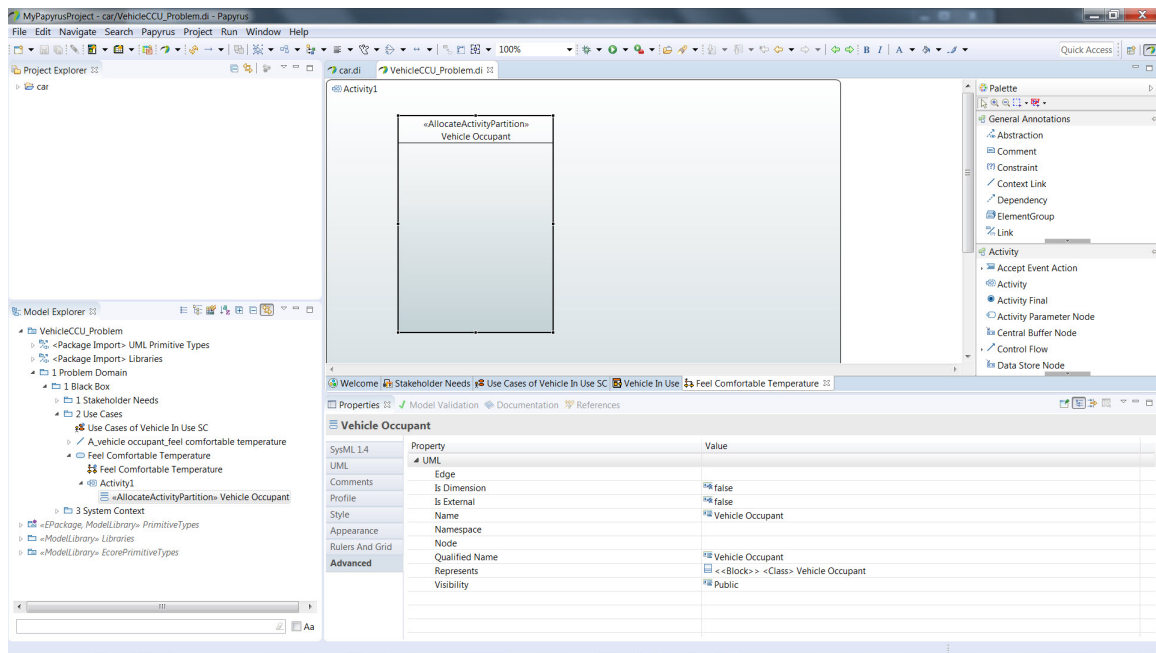


Figure 42: Vehicle Occupant Activity Partition in Activity Diagram

- Repeat steps 10 through 12 to create another activity partition named *Climate Control Unit* and for its **Represents** property value select the *Climate Control Unit* block (figure 43). Drag this partition to your activity diagram and position next to the *Vehicle Occupant* activity partition (figure 44). To help position your shapes go to the **Rulers and Grid** tab in the **Properties** window at the bottom of the screen and un-check the **Snap to Grid** item and check the **Snap to Shapes** item.

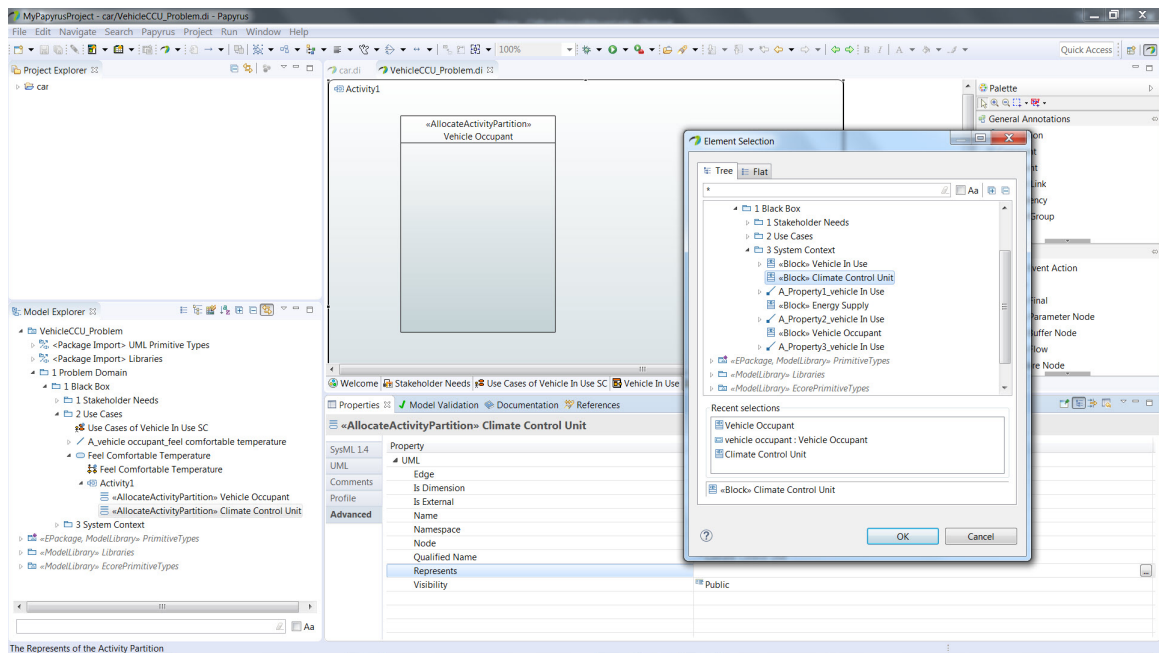


Figure 43: Create Climate Control Unit Activity Partition

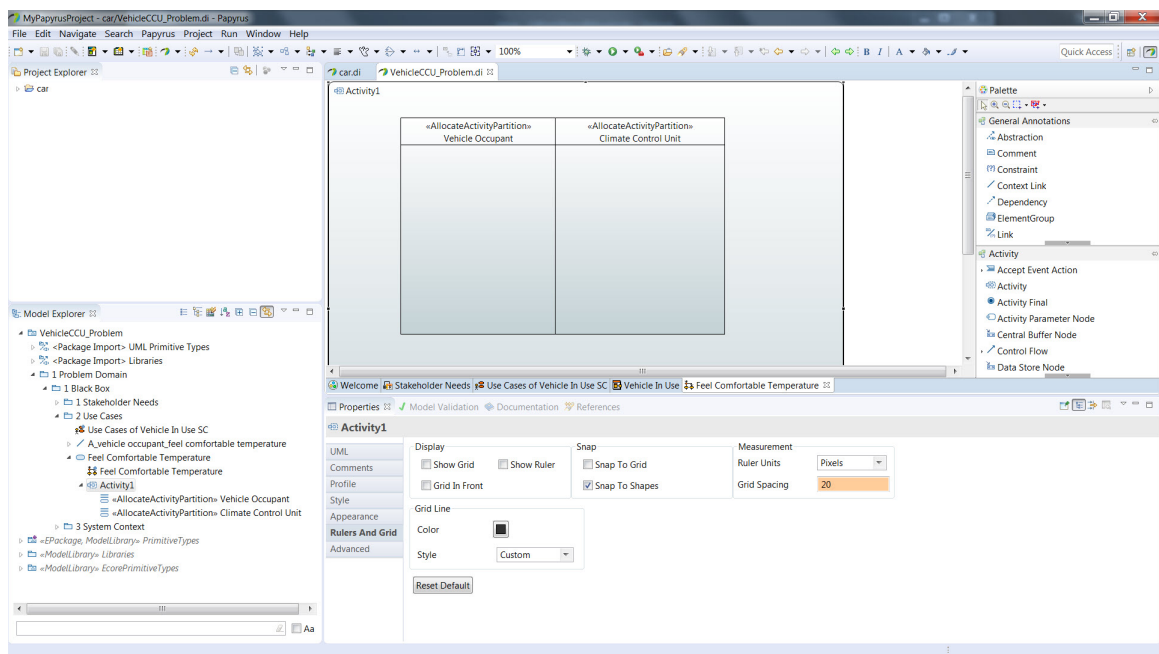


Figure 44: Climate Control Activity Partition in Activity Diagram

14. Let's say that our analysis of the *Feel Comfortable Temperature* use case scenario has resulted in the following actions:

- (a) Turn on Climate Control
- (b) Check System

- (c) Display Temperature
- (d) Start Climate Control
- (e) Set Temperature
- (f) Reach Required Temperature
- (g) Maintain Temperature
- (h) Turn off Climate Control
- (i) Stop Climate Control

Let's start creating our activity diagram.

15. Ensure that the *Feel Comfortable Temperature* activity diagram is selected in the main window. You will see the **Activity** tool palette on the right. Click on **Initial node** in the tool palette then click in the *Vehicle Occupant* activity partition and an initial node will be created (figure 45). Note that the node has a label next to it in the diagram. These labels can be cumbersome so select the *Initial node* in the activity diagram or in the *Model Explorer* window. Now select the **UML** tab in the **Properties** tab at the bottom of the screen (figure 45). Click in the **Label** field and type one blank space with the <Spacebar> and press *Enter*. The label should now be removed.

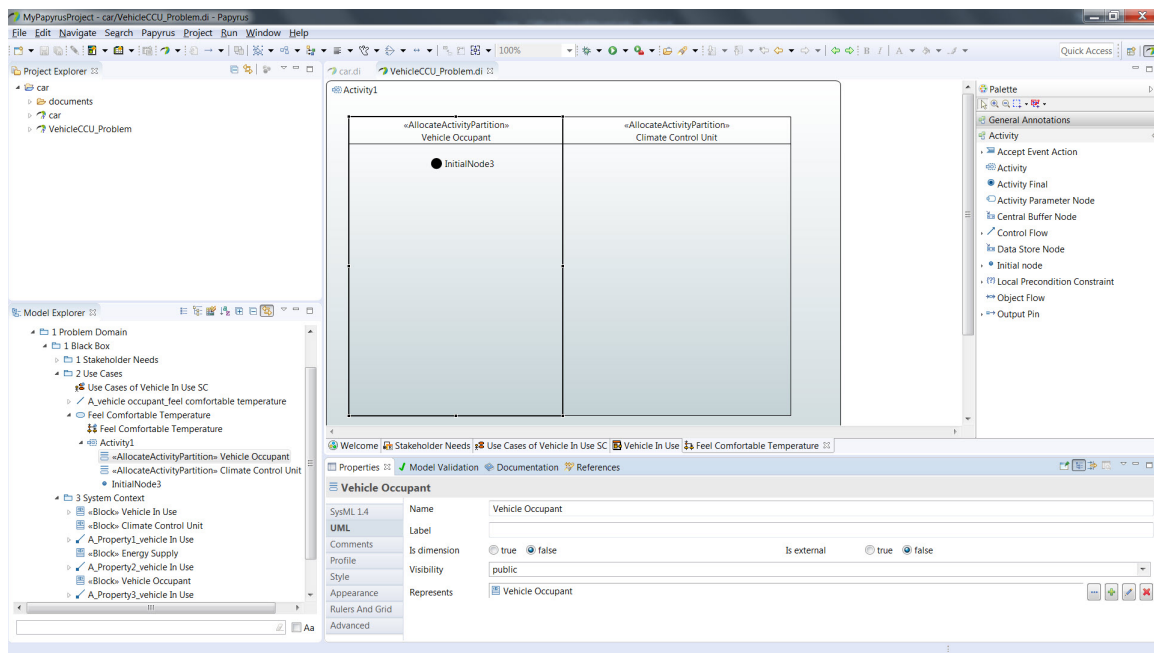


Figure 45: Activity Diagram Initial Node

16. Now we will start creating our sequence of actions. Right-click *Activity1* in the *Model Explorer* window and select **New Child** and in the pop-up window scroll down until you find **Opaque Action** and select it (figure 46). Type **Turn On Climate Control** for the name and then press *Enter*. The *Turn On Climate Control* action now appears

as a child node under *Activity1* in the *Model Explorer* (figure 47). The action does not yet appear on our activity diagram. We will take care of that next.

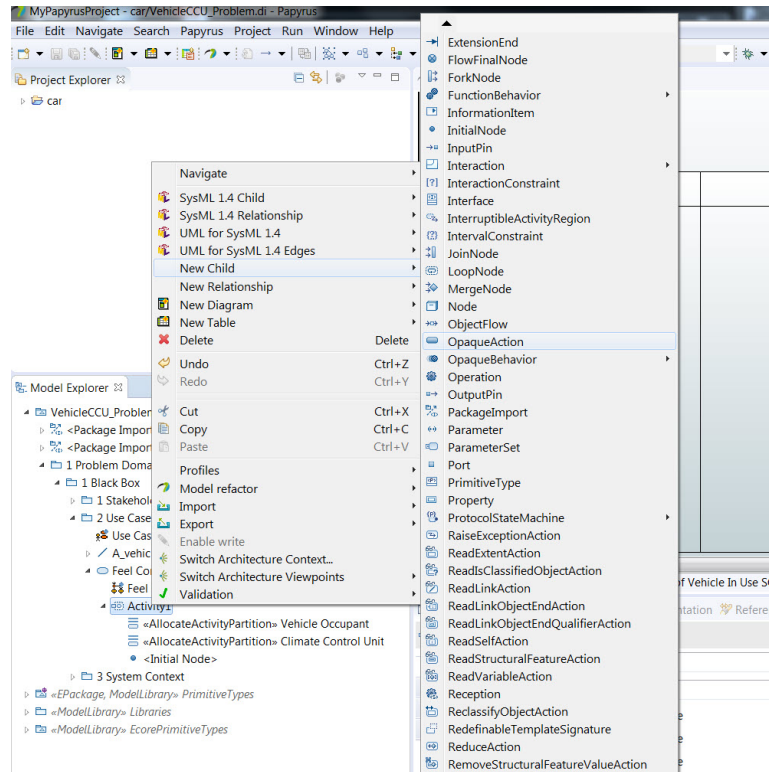


Figure 46: Create Opaque Action

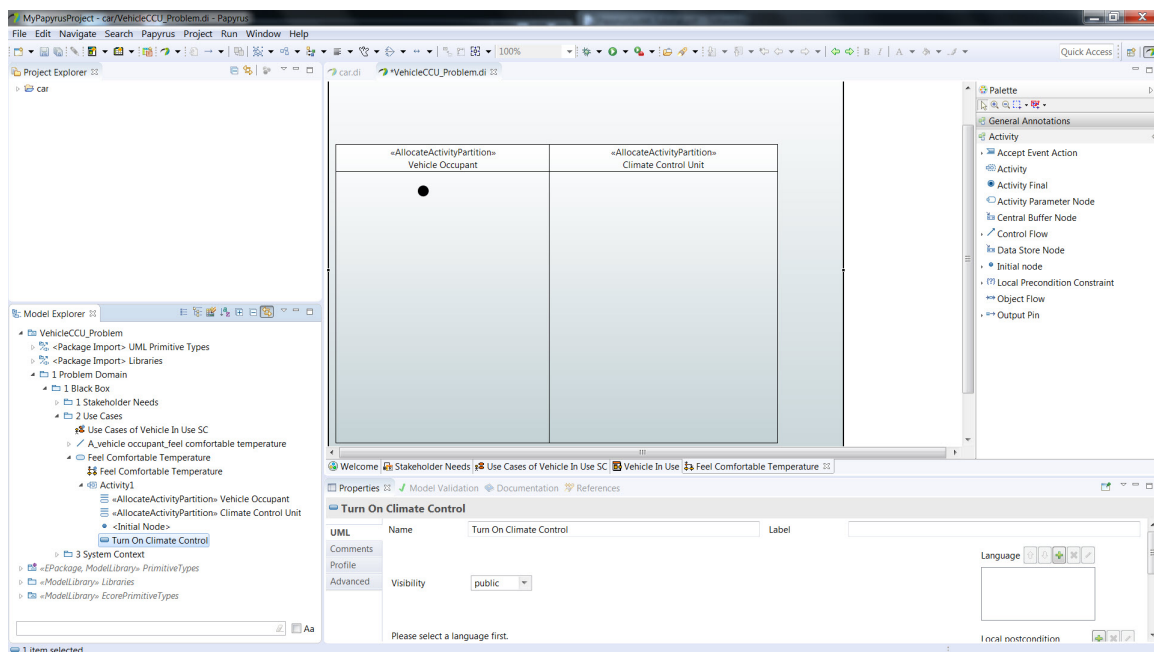


Figure 47: *Turn On Climate Control* Action Created

17. Click and Drag the *Turn On Climate Control* action from the *Model Explorer* window to the *Vehicle Occupant* activity partition underneath the initial node. The *Turn On Climate Control* action should now appear on the activity diagram (figure 48).

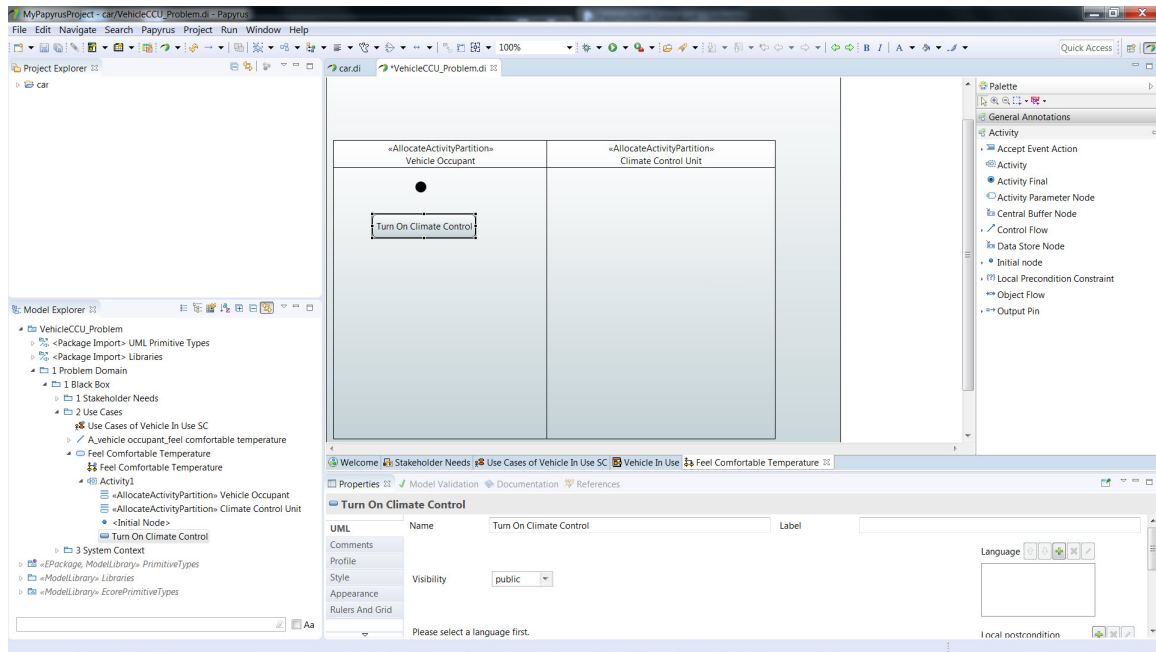


Figure 48: *Turn On Climate Control* Action in Activity Diagram

18. Repeat the procedure to create the other actions listed in step 14 and place them in the activity partitions (swimlanes). Use the activity tool palette on the right and place an **Activity Final** node underneath the *Stop Climate Control* action. Your diagram should look like (figure 49).

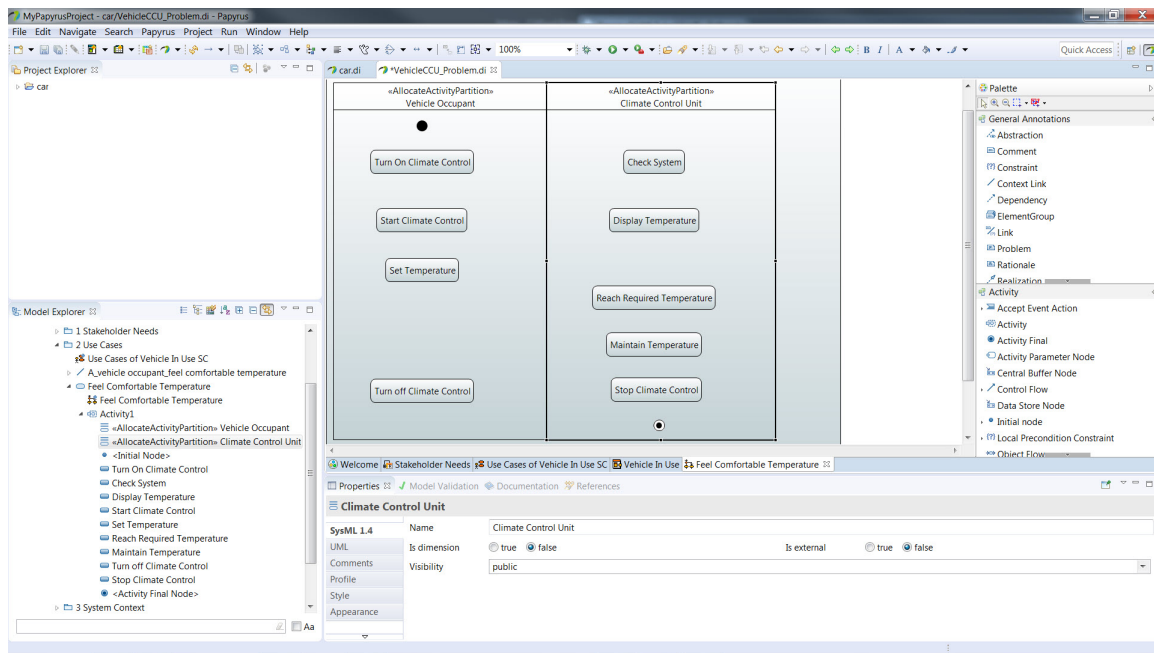


Figure 49: *Feel Comfortable Temperature* Activity Diagram

19. Now we will start adding our control flows to these actions. In the activity palette on the right Click on **Control Flow**. Now Click on the *Initial Node* in the activity diagram and then Click on the *Turn On Climate Control* action. Just hit <Enter> for the name as we will not name this control flow (I will call this an unnamed control flow from now on). A control flow will be added between these items (figure 50). If you would like the control flow line to be a little thicker you can select the **Appearance** tab in the *Properties* window and increase the value of the **Line width** parameter. Create another control flow between the *Turn On Climate Control* action and the *Check System* action (figure 50).

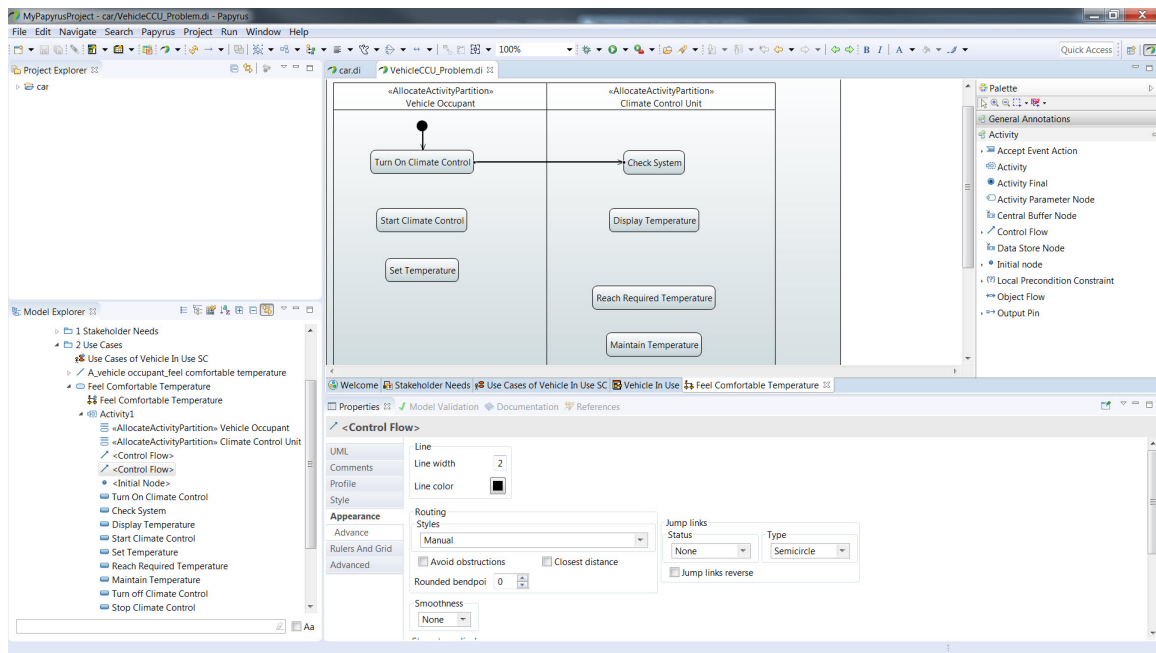


Figure 50: Control Flow

20. Now when we analyze the *Check System* action we come up with two possible cases. If the system is OK then we proceed to the next action which is *Display Temperature*, but if the system has some kind of error we want to proceed to the *Stop Climate Control* action. So, we need a decision node with a guard condition on the output of the *Display System* action. In the *Activity* palette on the right Click on the arrow next to **Initial node** to expand its contents and Click on **Decision node**. Now, in the activity diagram Click on the space between the *Check System* action and the *Display Temperature* action. A decision node will be created on the activity diagram (figure 51).

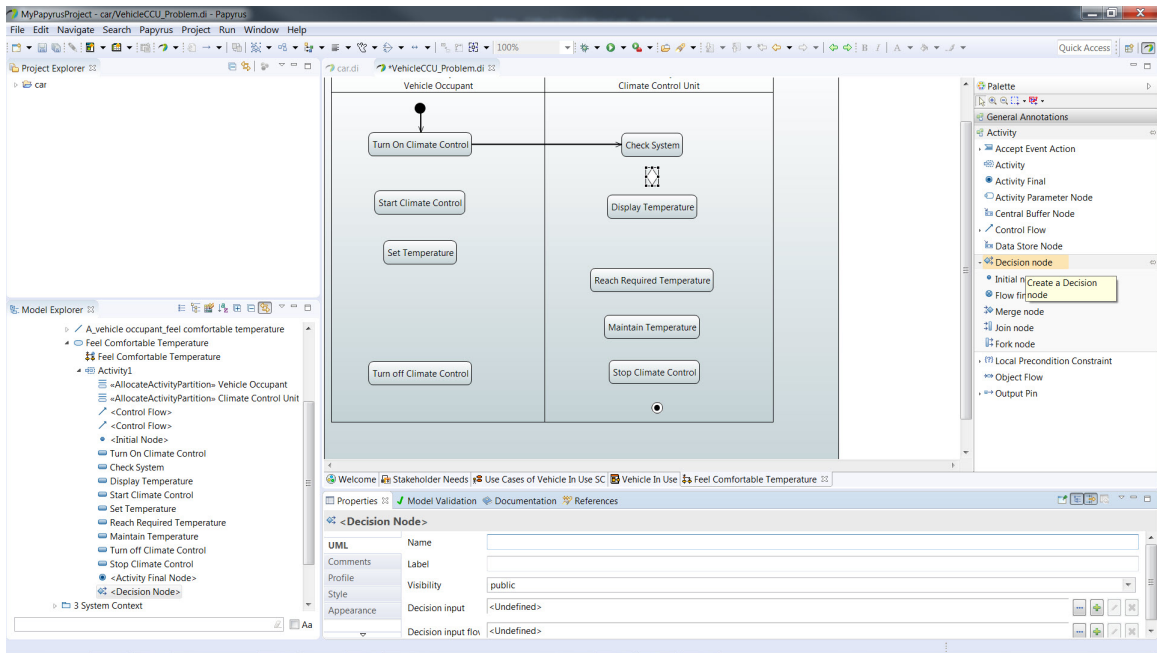


Figure 51: Control Flow

21. Add a control flow between the *Check System* action and the decision node you just created (figure 52).

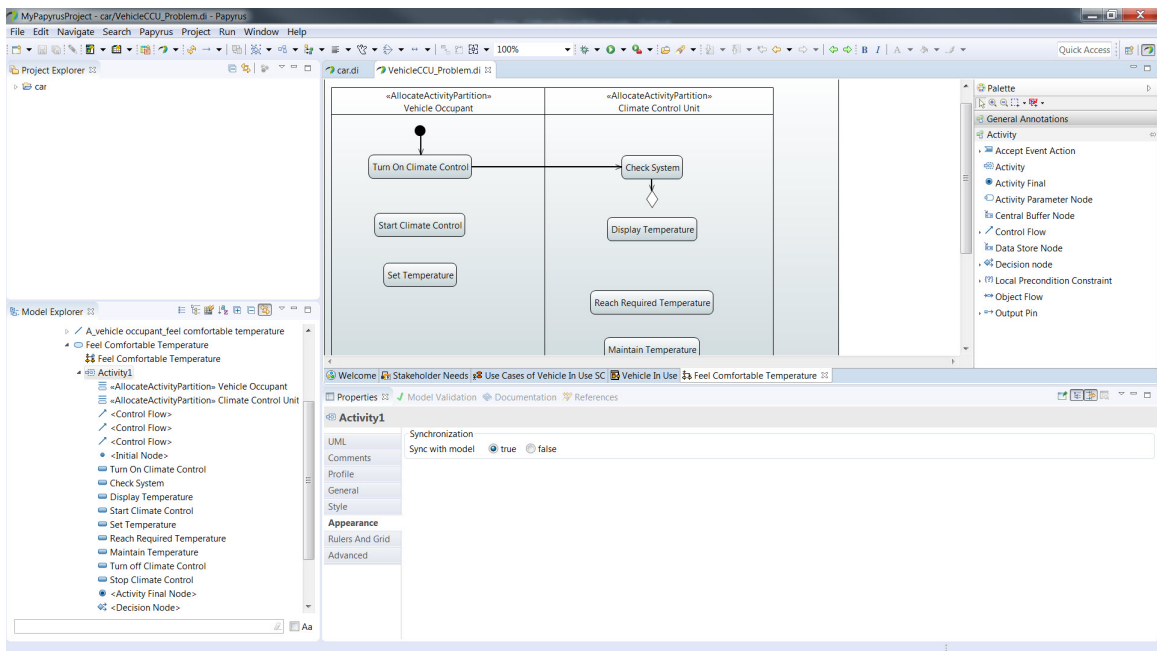


Figure 52: Control Flow at Decision Node

22. Now create a control flow between the decision node and the *Display Temperature* action and type *[OK]* for the name. This represents our guard condition and therefore

this path is to be executed if the *Check System* action results show that the system is functioning normally (figure 53).

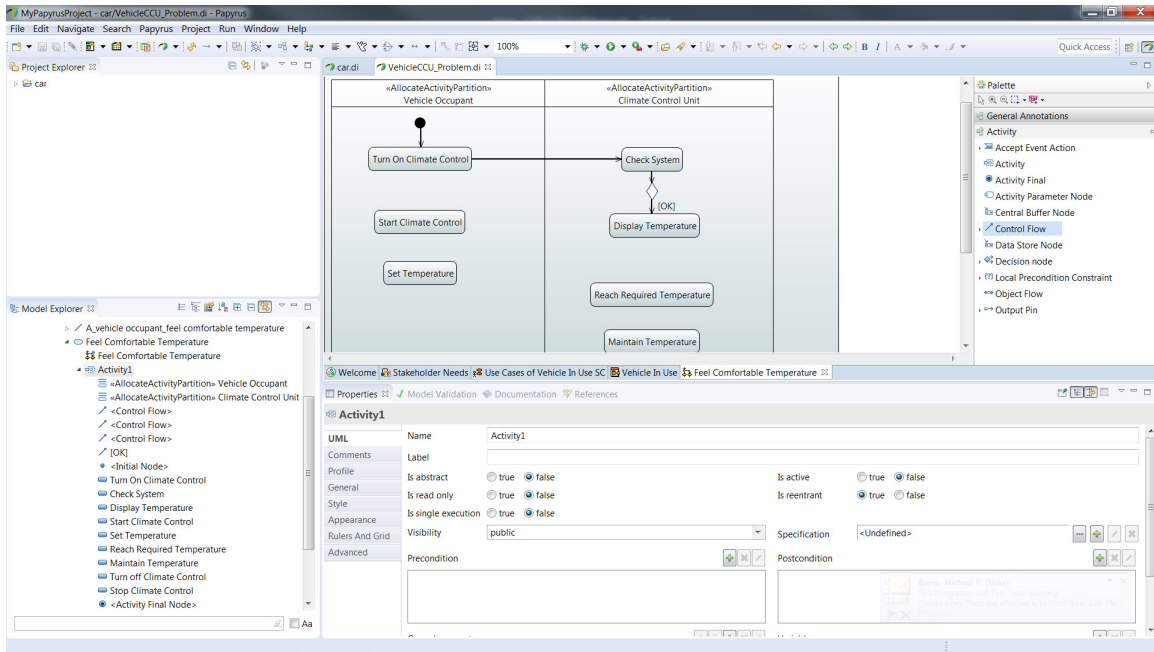


Figure 53: Creating the [OK] Guard Condition

23. Now create a control flow between the decision node and the *Stop Climate Control* action and type *Not OK* for the name. This represents our other guard condition to be executed if the *Check System* action results show that there is some kind of fault in the system. In this case we will go to the *Stop Climate Control* action. Note that you can Click on the control flow line to break it into segments and then drag these control points to reposition the segments. Add an unnamed control flow between the *Stop Climate Control* action and the final node. Your activity diagram should look like (figure 54).

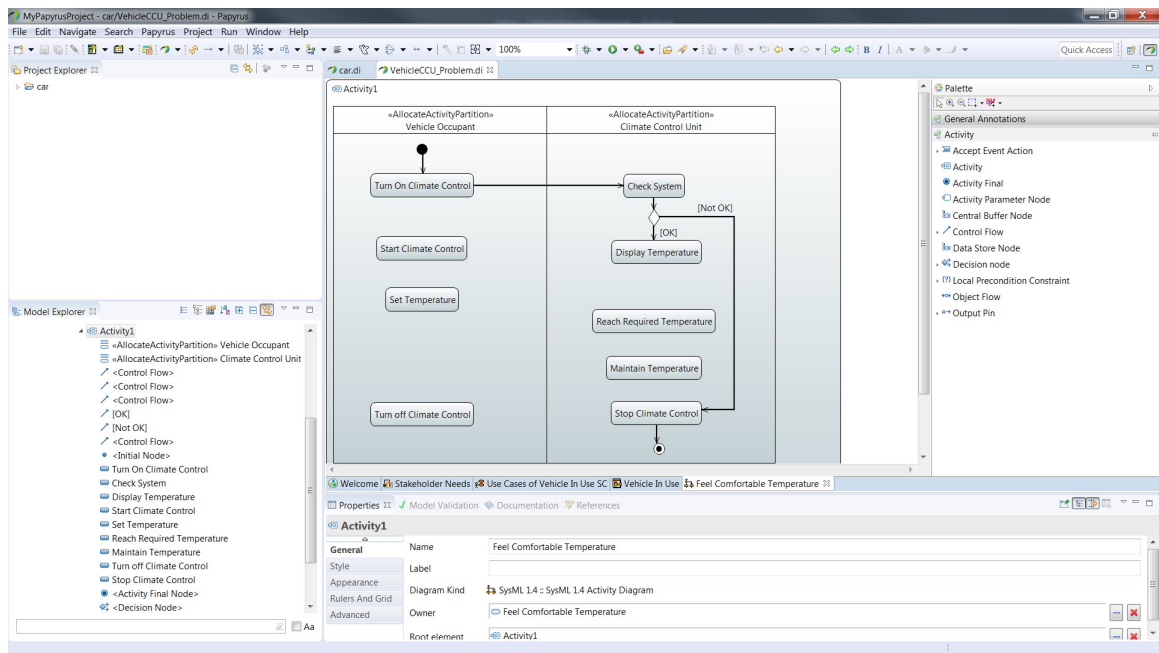


Figure 54: Creating the *[Not OK]* Guard Condition

24. The next action after displaying the temperature is to start the climate control so create an unnamed control flow from the *Display Temperature* action to the *Start Climate Control* action. After the climate control is started the next action is to set the temperature so create an unnamed control flow between the *Start Climate Control* action and the *Set Temperature* action. Add an unnamed control flow between the *Set Temperature* action and the *Reach Required Temperature* action. Add an unnamed control flow between the *Turn off Climate Control* action and the *Stop Climate Control* action. Your activity diagram should look like (figure 55).

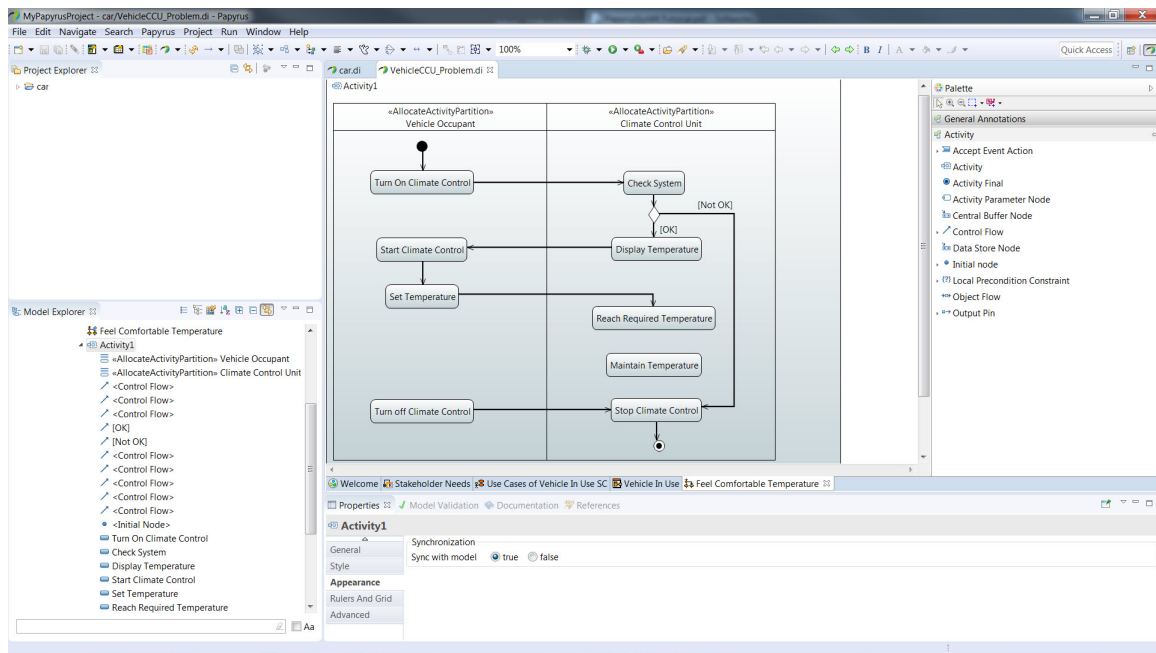


Figure 55: Creating More Control Flows

25. To complete our control flows for this simple example we need to add a couple more decision nodes; One between the *Reach Required Temperature* action and the *Maintain Temperature* action and another below the *Maintain Temperature* action. We also need to add the remaining control flows. Use your skills from the previous steps to make your diagram look like (figure 56).

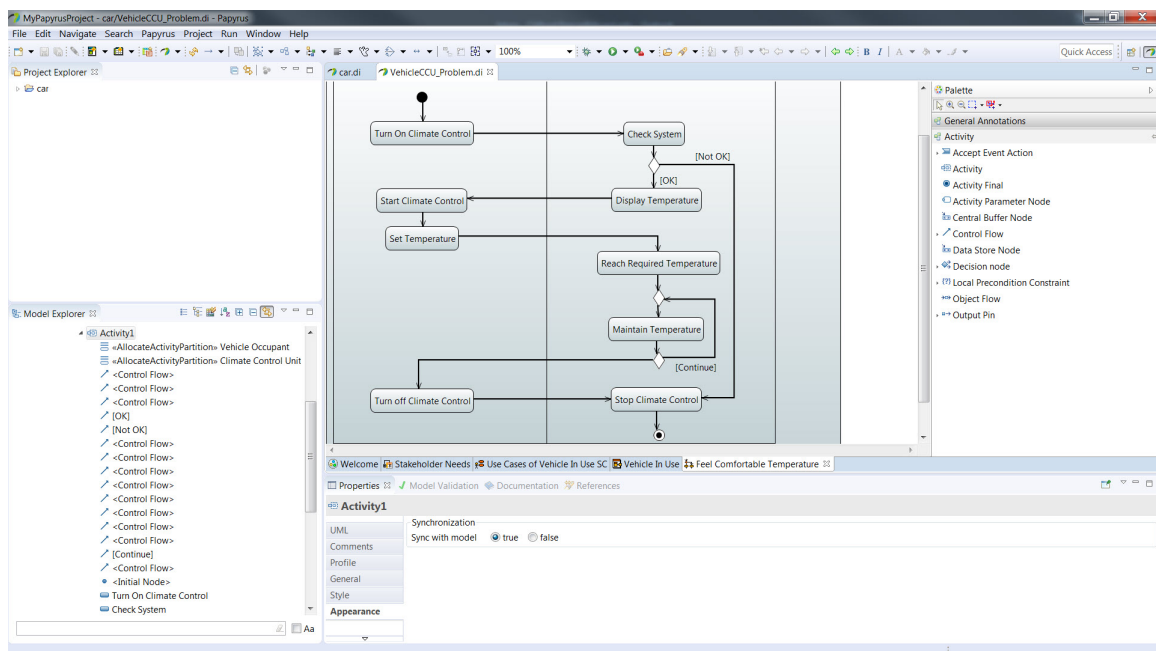


Figure 56: Final Control Flows

26. Now we have the actions and the control flow in our activity diagram but we need to pass some data as well. The required temperature should be an output from the *Set Temperature* action and this is provided as an input to both the *Reach Required Temperature* action and the *Maintain Temperature* action. In this way the system knows the desired temperature that it is to reach and maintain once it is set by the user. First create a fork node by Clicking on **Fork node** from the *Activity* palette and then Click in the space underneath the *Set Temperature* action. Size the fork as shown in (figure 57).

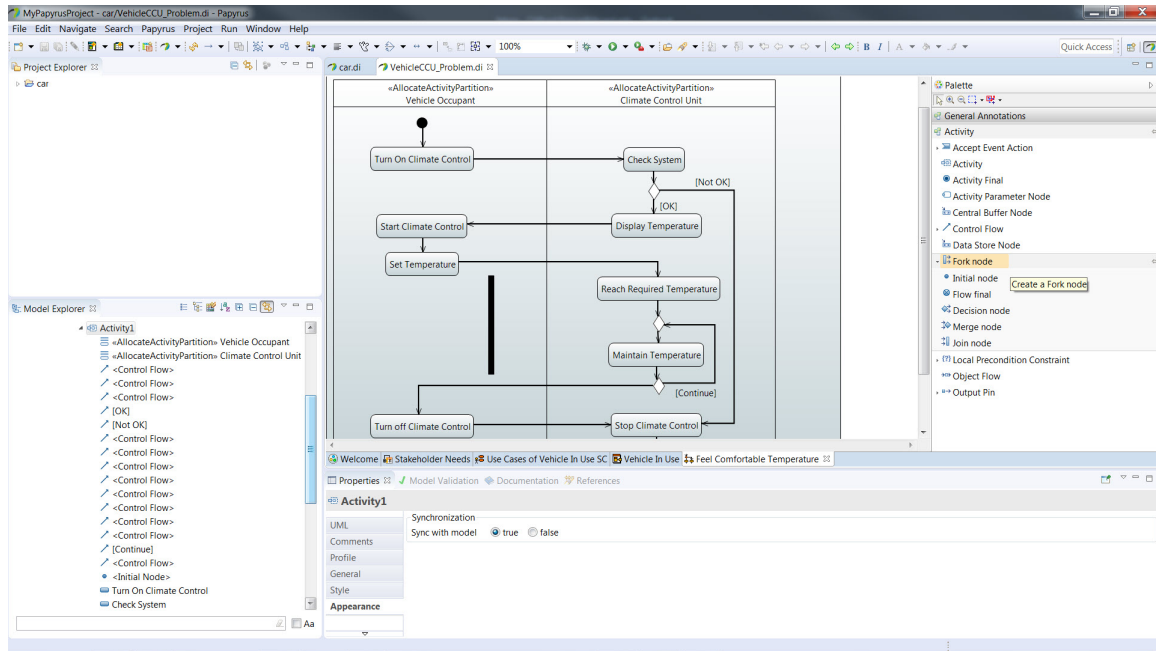


Figure 57: Create Fork Node

27. Now we will create an output pin on the *Set Temperature* action to pass the *Temperature* as an output. In the *Activity* palette on the right find the **Output Pin** item and Click on it. Now in the activity diagram Click on the *Set Temperature* action and the output pin will be attached to the action. You can Click and Drag the output pin to the bottom of the *Set Temperature* action. Now create a control flow from the output pin to the fork node and name the control flow *Temperature*. Your activity diagram should look like (figure 58).

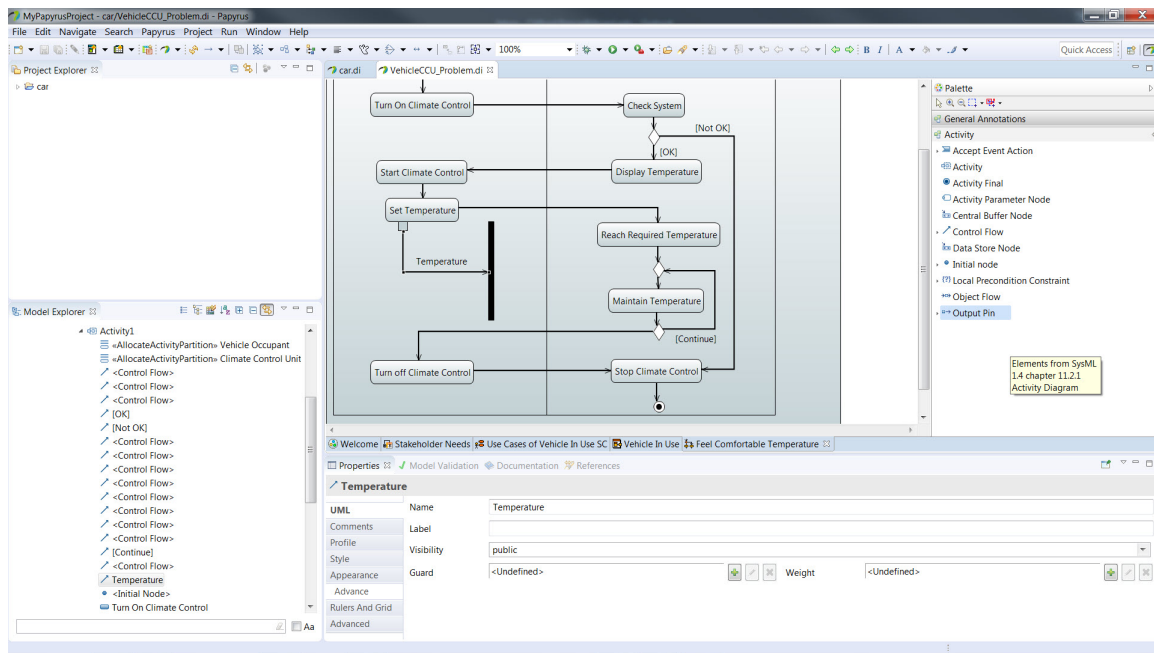


Figure 58: Temperature Output

28. Now we need to provide the *Temperature* as input to both the *Reach Required Temperature* action and the *Maintain Temperature* action. To do this go to the *Activity* palette on the right and Click on **Input Pin** then Click on the *Reach Required Temperature* action and an input pin will be created. Position this pin on the left side of the action if not already there. Repeat the procedure to create an input pin on the *Maintain Temperature* action. Now create control flows between the fork node and each input pin you just created. Your activity diagram should look like (figure 59) and our example activity diagram is complete.

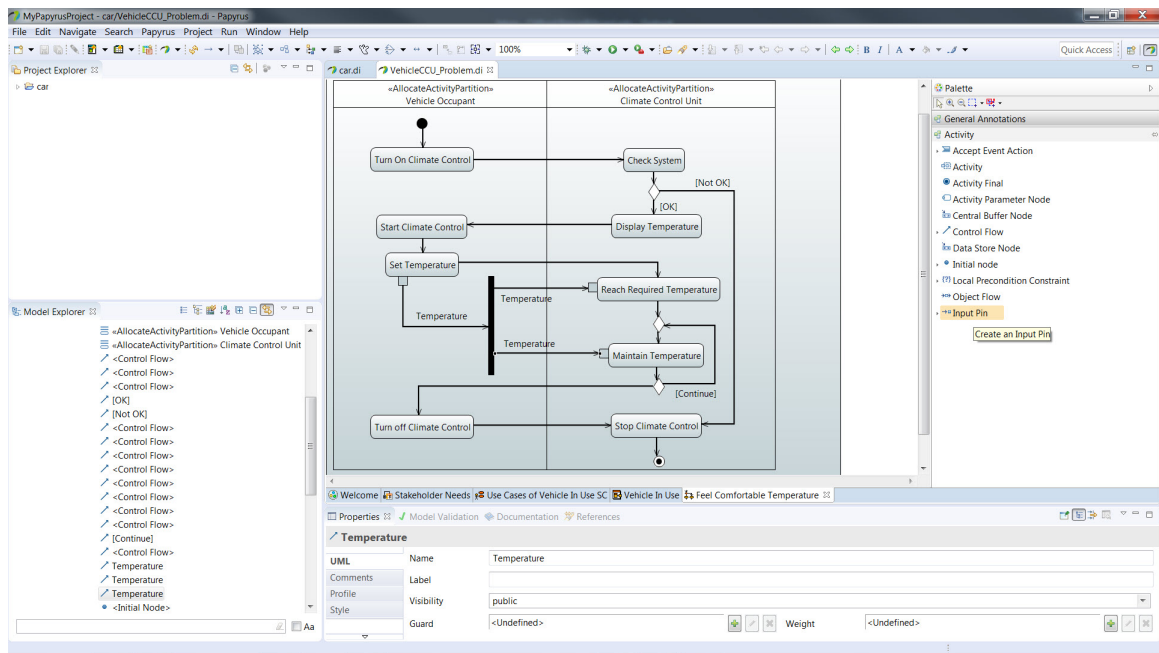


Figure 59: Temperature Input