Tutorial 6 — SysML for DSE

Overview

In the previous tutorials, if you wanted to explore the effects of changing the parameter values of a multi-model, you would have had to manually run each co-simulation and inspect the results for each parameter value. This tutorial will show you how to do this automatically. In particular, it will show you how to:

1. Declare the objective functions that will be used to assess the system under test
2. Define the parameters that will be varied during DSE along with their allowed values
3. Define which objectives will be used for ranking along with the preference for higher or lower values
4. Export the configuration for use during DSE

Requirements

- INTO-CPS Application
- COE (Co-simulation Orchestration Engine) – accessible through the INTO-CPS App Download Manager
- Modelio – accessible through the INTO-CPS App Download Manager
  - If you use linux please make sure your installation meets the dependencies listed in: `https://www.modelio.org/downloads-links/requirements.html`
  - For debian based distributions the following command may be usefull: dpkg-query -l libatk1.0-0 libc6 libcairo2 libgtk-3-0 libglib2.0-0 libwebkitgtk-1.0-0 libxtst6 libstdc++6

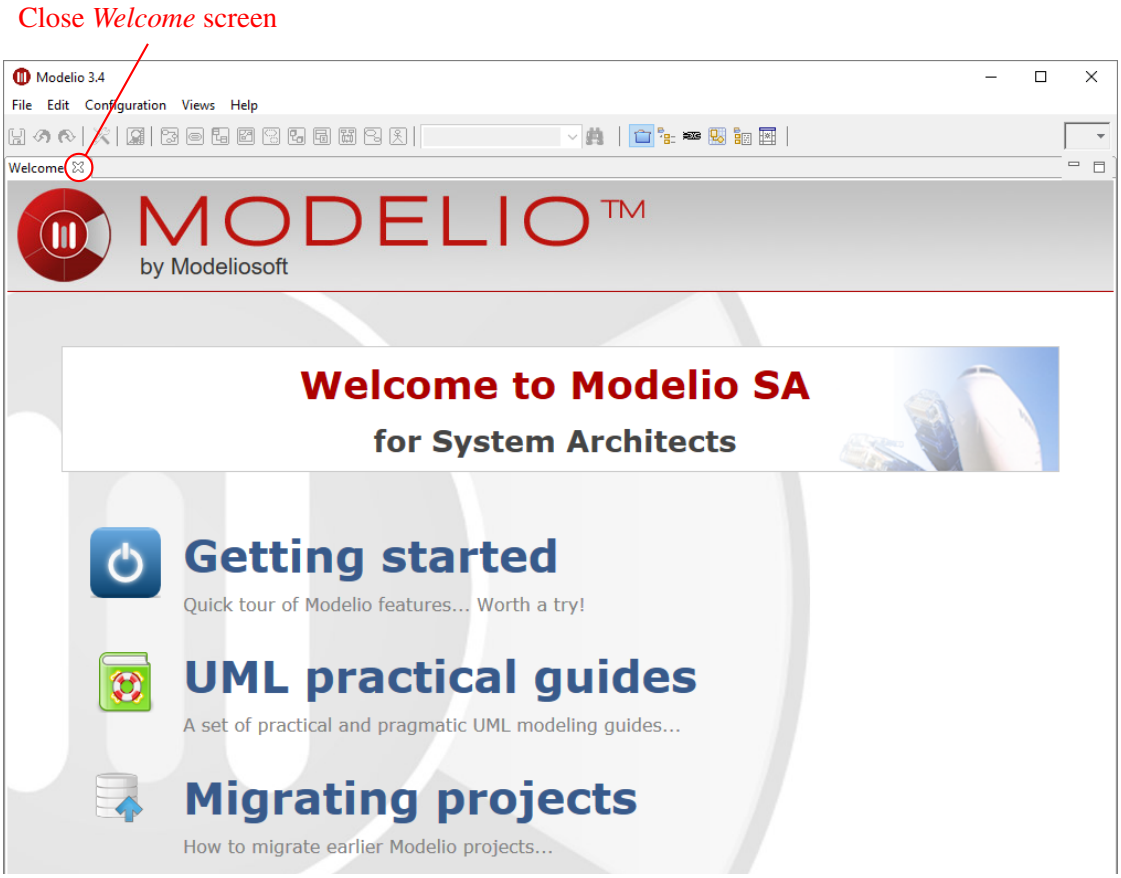
You may have been provided with tools and tutorials on a USB drive at your training session. Otherwise:

- Follow Tutorial 0 with the guidelines to install the INTO-CPS Application and COE.
- Ask your instructor for the tutorial materials. These are available for students and members of the INTO-CPS Association[1].

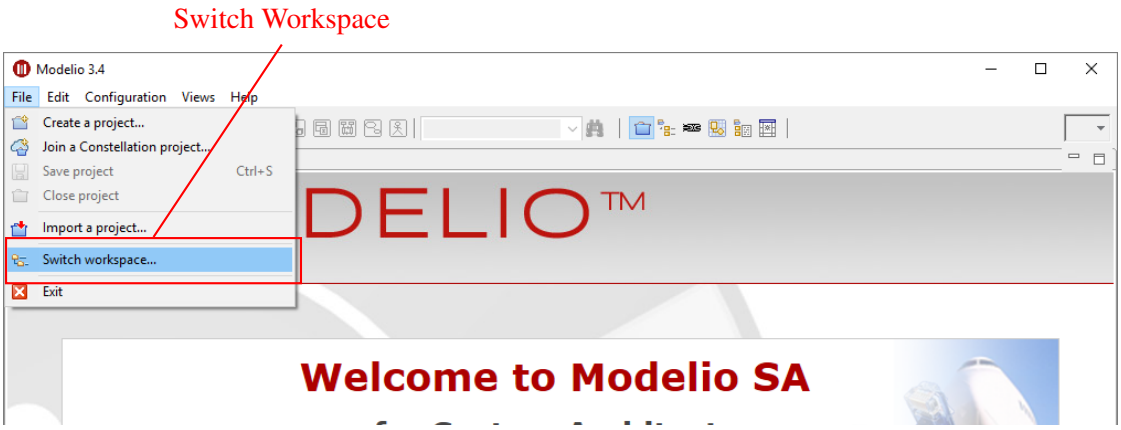
[1] `https://into-cps.org/`

1 Opening a Project

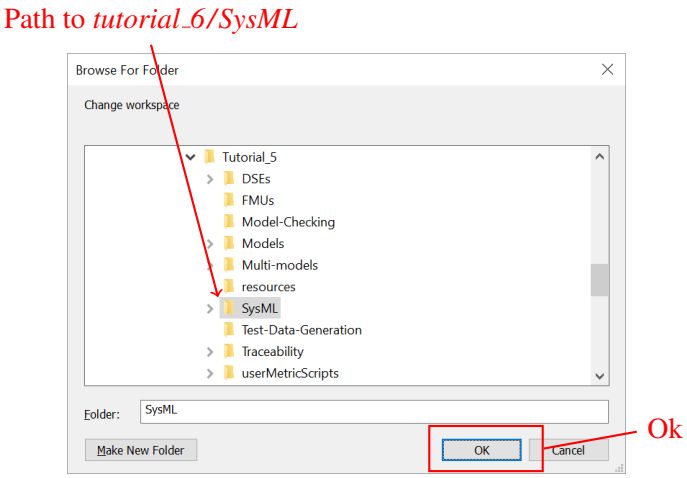
Step 1. Launch *Modelio*. On first loading, you may have to close the *Welcome* screen (you can bring it back with 'Help > Welcome' if you need)



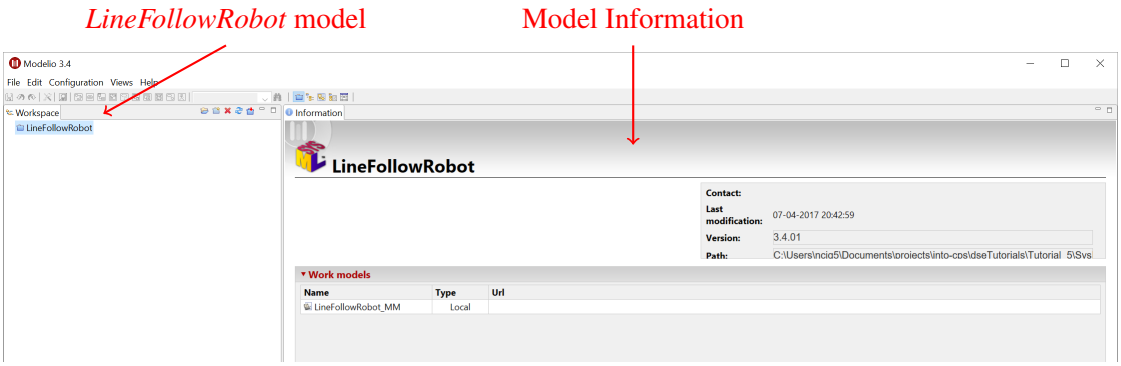
Step 2. A workspace must be chosen, select 'File > Switch Workspace'.



Step 3. Set the *Workspace* to the location of *tutorial_6/SysML* and click *Ok*.



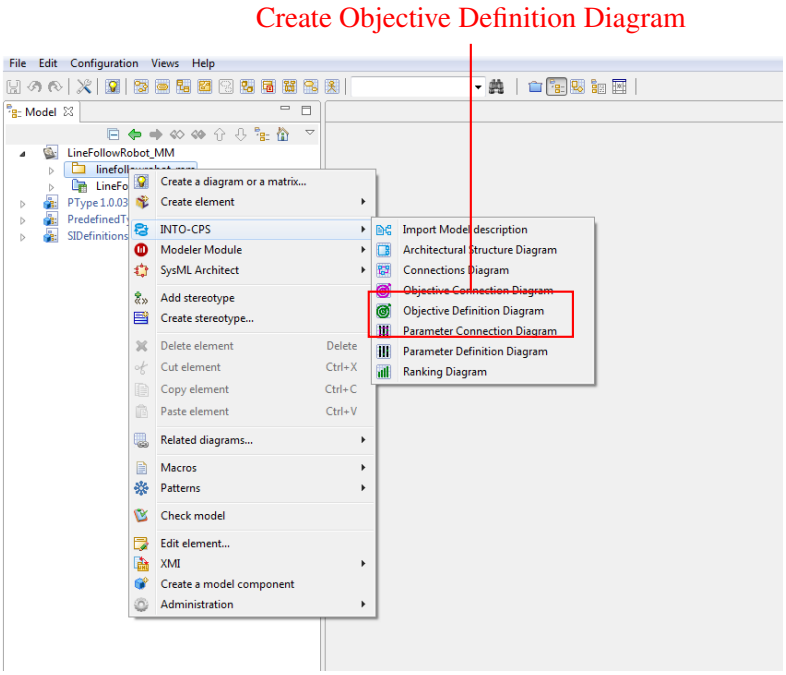
Step 4. Left-click on the *LineFollowRobot* model once on the left to see details of the model. Double-click the *LineFollowRobot* model to open the model.



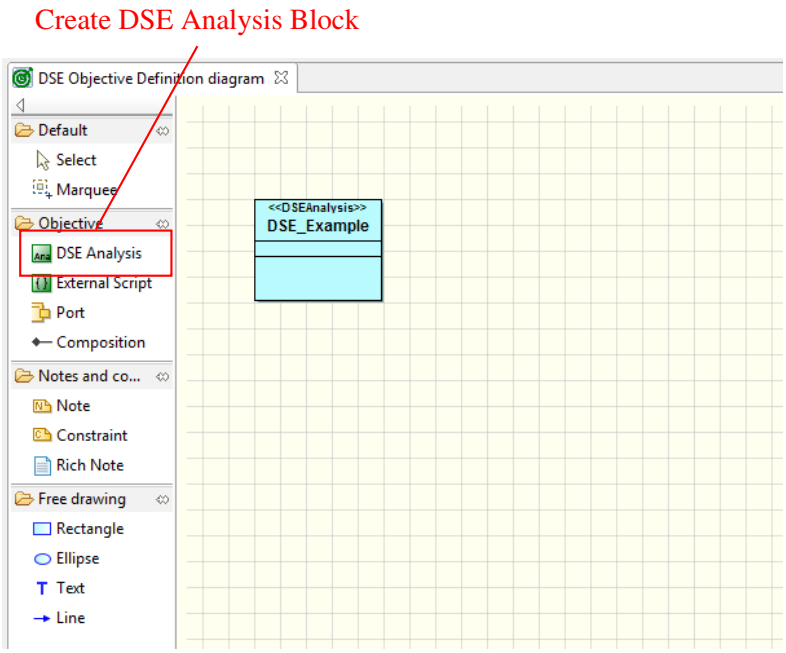
2 Defining Objectives

This section will describe the definition of objectives for use by DSE. Objectives are characterising measures of performance that may be used to determine the relative benefits of competing designs. Examples include, as we shall see, the time a robot takes to complete a circuit or the accuracy with which the robot is able to follow a path.
When defining the objectives we first describe them in terms of their name, the script file that will be used to compute them and the ports that will provide the data they require. Instances of these definitions are then created and the ports either filled with a static value or connected to data exchanged in the multi-model.

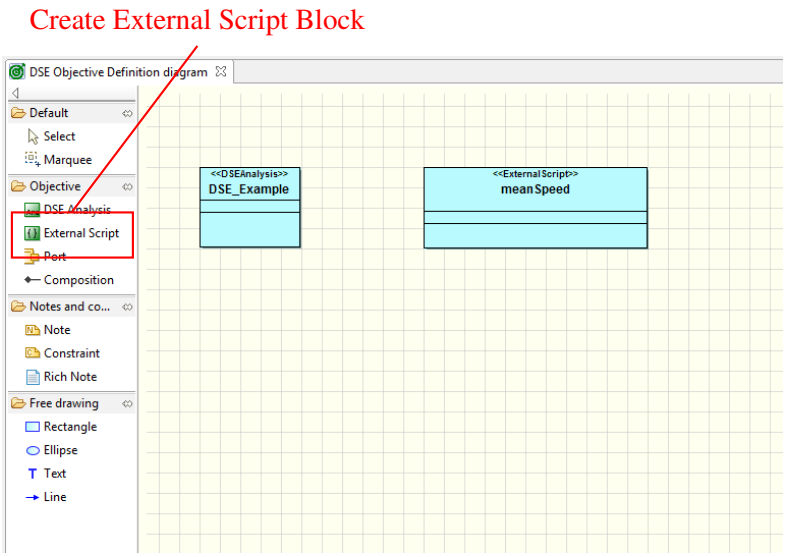
Step 5. Right click `linefollowrobot_mm` in model outline. Select 'INTO-CPS > Objective Definition Diagram'.



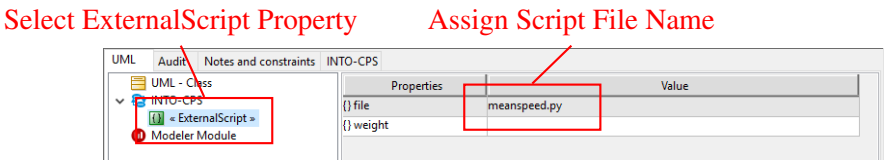
Step 6. From the *Toolbox*, select *DSE Analysis* and click on the empty diagram to create the DSE block. Double click the block and rename it '$DSE\_Example$'.



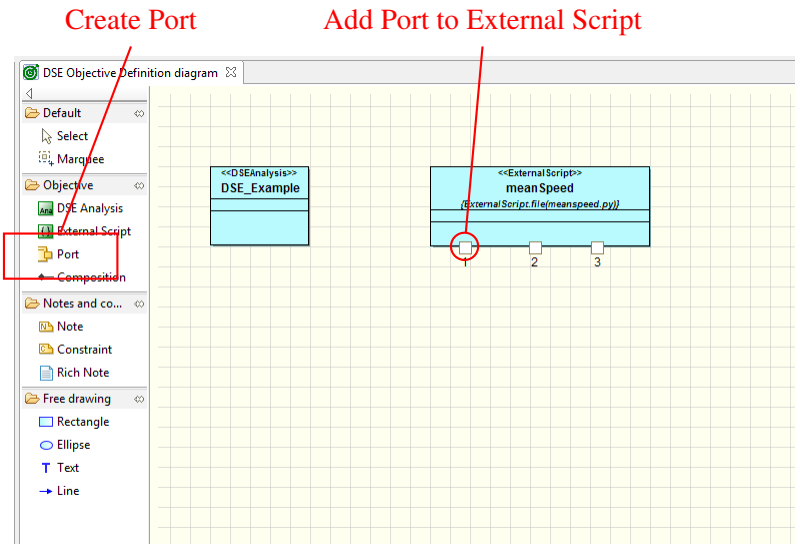
Step 7. From the *Toolbox*, select *External Script* and add it to the diagram. Double click the block that this creates and change its name to '$meanSpeed$'. Do not close the editor window.



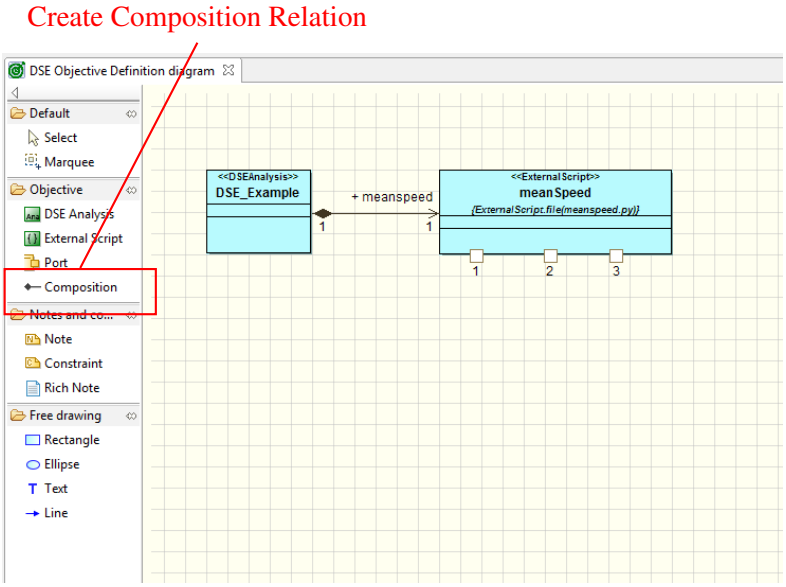
Step 8. On the top of the editor window, select the "Properties" tab to show a window similar to the one below, and change {}file to '$meanspeed.py$', and close the editor window. Note: you must use the "Properties" tab to make this change. The "INTO-CPS" tab only allows you to view the file property.



Step 9. So that objective scripts can later be connected, select the *Port* tool from the toolbox, and click on the External Script block to add a port to it. Click on the new port and change its name to '1' in the *Element* editor. Repeat this to add two more ports to the block, named '2' and '3'.



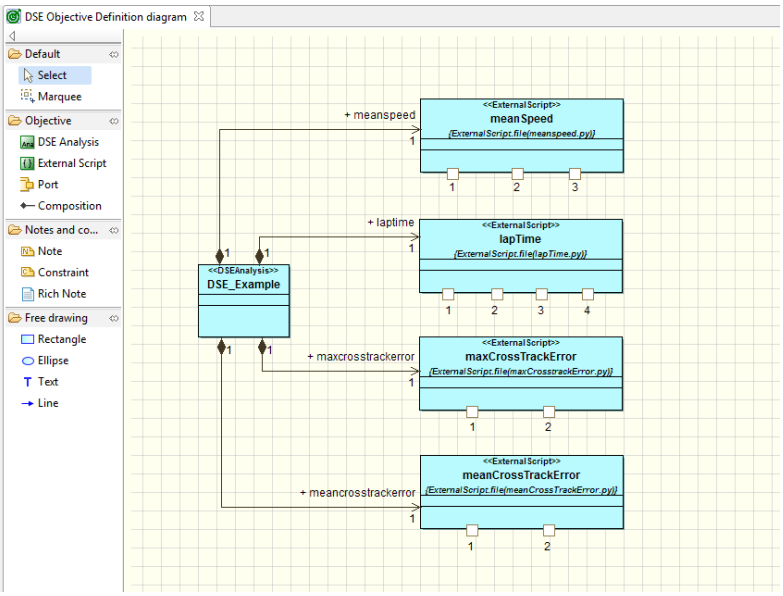
Step 10. To associate the external script with the DSE, select the *Composition* relation from the *Toolbox*, and click first on the '$DSE\_Example$' block, and then on the new '$meanSpeed$' block.



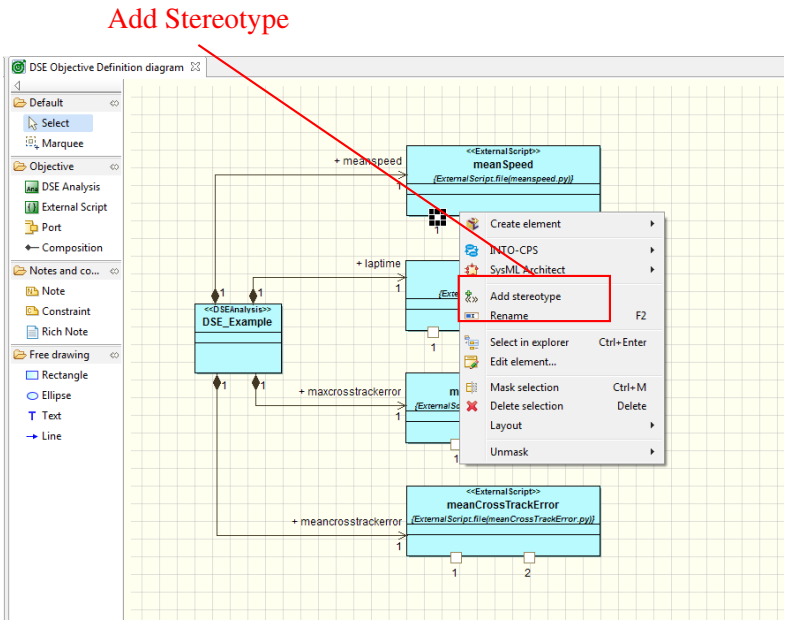
Step 11. Repeat Step 7. – Step 10. to associate the following scripts with the DSE:

- Name: '$lapTime$', Script: '$lapTime.py$', Ports: '$1, 2, 3, 4$';
- Name: '$maxCrossTrackError$', Script: '$maxCrosstrackError.py$', Ports: '$1, 2$';
- Name: '$meanCrossTrackError$', Script: '$meanCrosstrackError.py$', Ports: '$1, 2$';

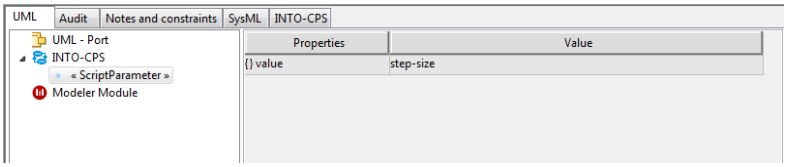
The model should now look like this:



Step 12. Some objective ports are associated with named values instead of model parts. An example of this is Port '1' of '$meanSpeed$'. Right click on the port and select *Add Stereotype*. Double click 'ScriptParameter' nested under 'INTO-CPS'.



Step 13. Double click the port and select 'INTO-CPS > ScriptParameter'. Change {}value to 'step-size' and close the editor window.

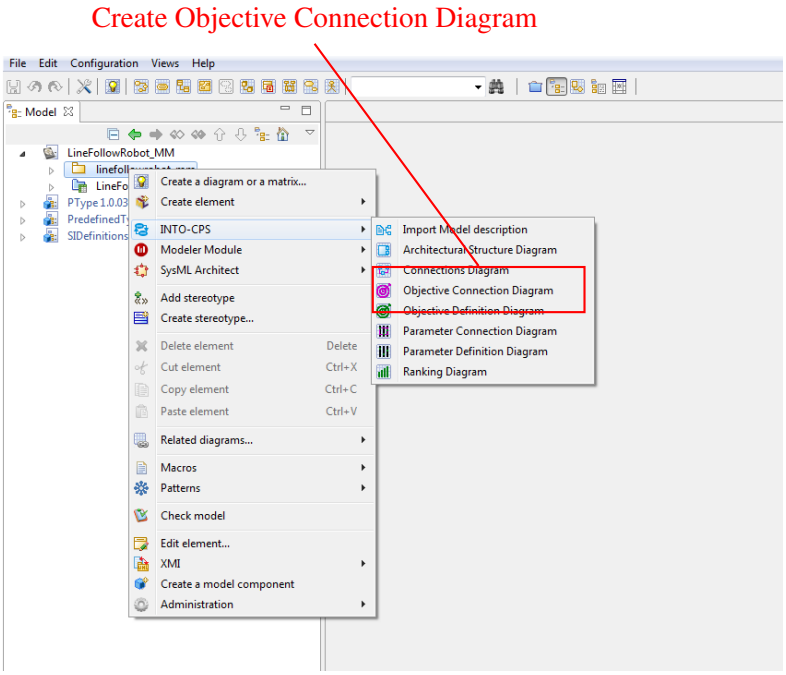


Step 14. Repeat Step 12. & Step 13. to assign the following script parameters:

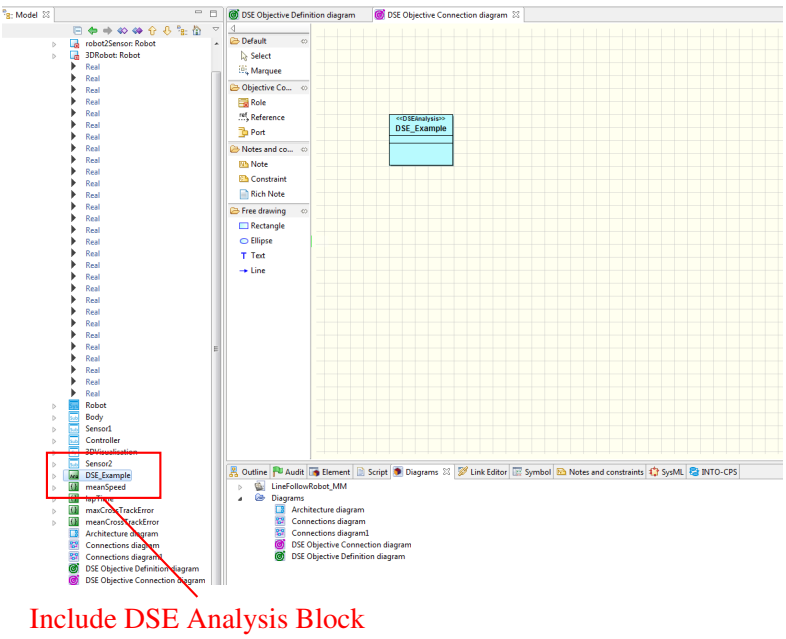
- Port '1' of '$lapTime$': Value '$time$';
- Port '4' of '$lapTime$': Value '$studentMap$';

3 Connecting Objectives

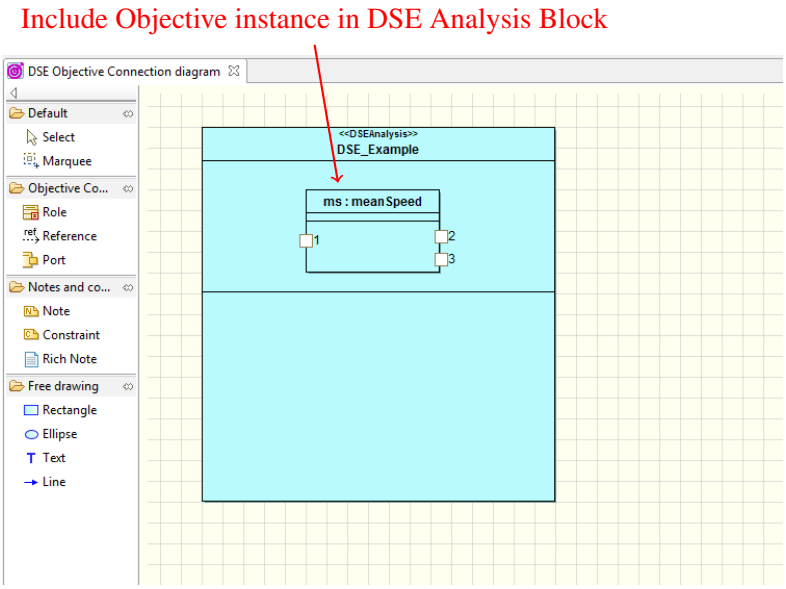
Step 15. Right click '$linefollowrobot\_mm$' in model outline. Select 'INTO-CPS > Objective Connection Diagram'.



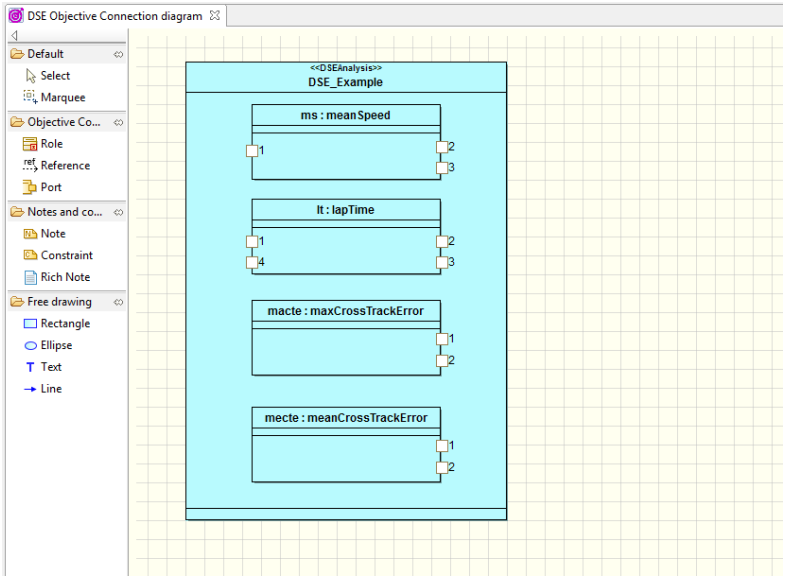
Step 16. Find the '$DSE\_Example$' block in the model outline and drag it onto the empty diagram to add the DSE block created in Part 2.



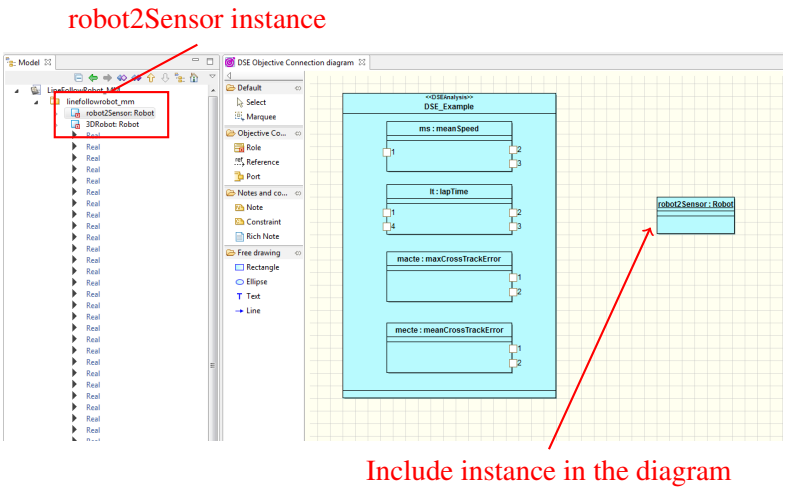
Step 17. Find the '$meanSpeed$' block in the model outline and drag it onto the DSE example block now visible on the diagram. Double click the block that this creates and name the instance 'ms'.



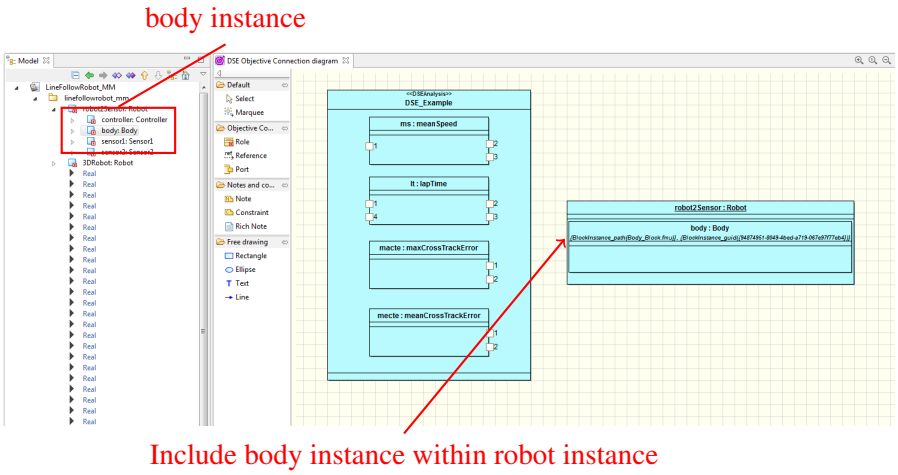
Step 18. Repeat Step 17. for '$lapTime$', '$maxCrossTrackError$', and '$meanCrossTrackError$', using the instance labels 'lt', '$mecte$', and '$macte$' respectively.



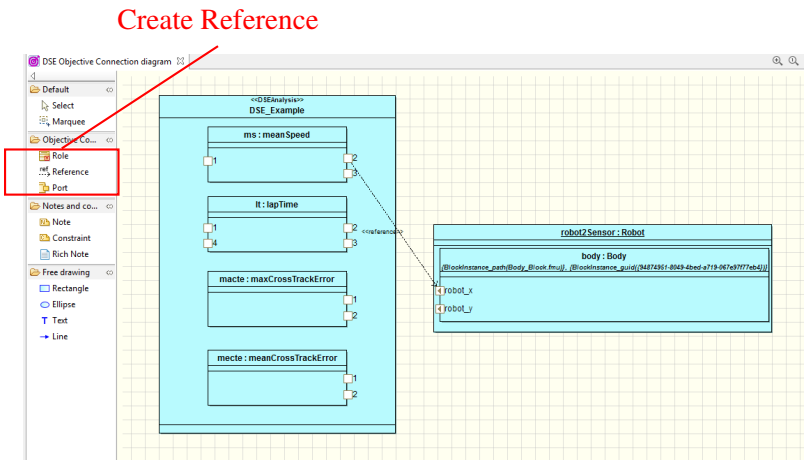
Step 19. To link Objectives to the multi-model, find the '$robot2Sensor$' instance in the model overview (under Architecture) and drag it onto the diagram to include it in the DSE.
Note: drag the block onto empty space, not onto the $DSE\_Example$ block.



Step 20. Expand '$robot2Sensor$' in the model overview and drag the nested instance of '$body$' onto the '$robot2sensor$' instance just created.



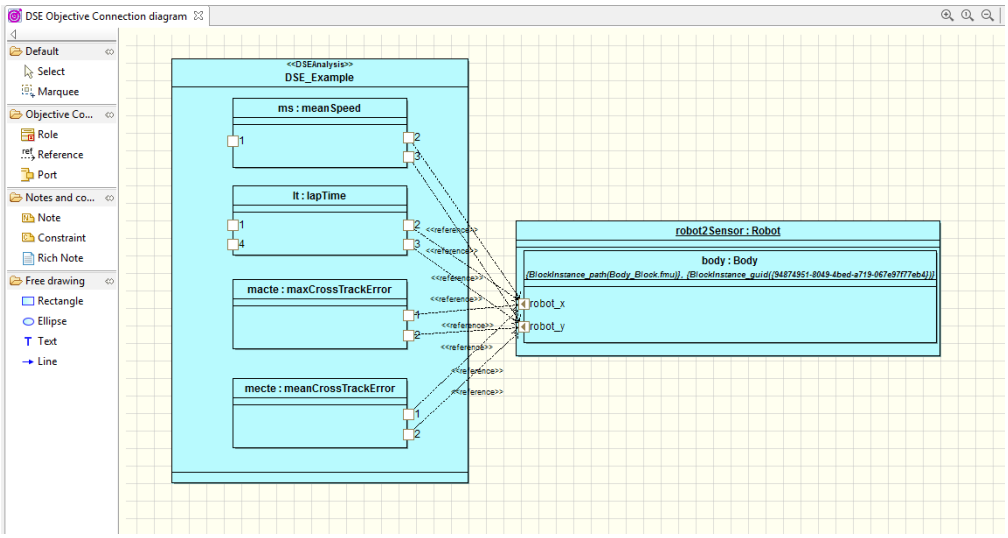
Step 21. Finally, select the *Reference* tool from the *Toolbox* and link port '2' from 'ms' to port '$robot\_x$' from '$body$'.



Step 22. Repeat Step 21. to link:

- '$ms > 3$' to '$body > robot\_y$';
- '$lt > 2$' to '$body > robot\_x$';
- '$lt > 3$' to '$body > robot\_y$';
- '$mecte > 1$' to '$body > robot\_x$';
- '$mecte > 2$' to '$body > robot\_y$';
- '$macte > 1$' to '$body > robot\_x$';
- '$macte > 2$' to '$body > robot\_y$';

The model should now look like this:



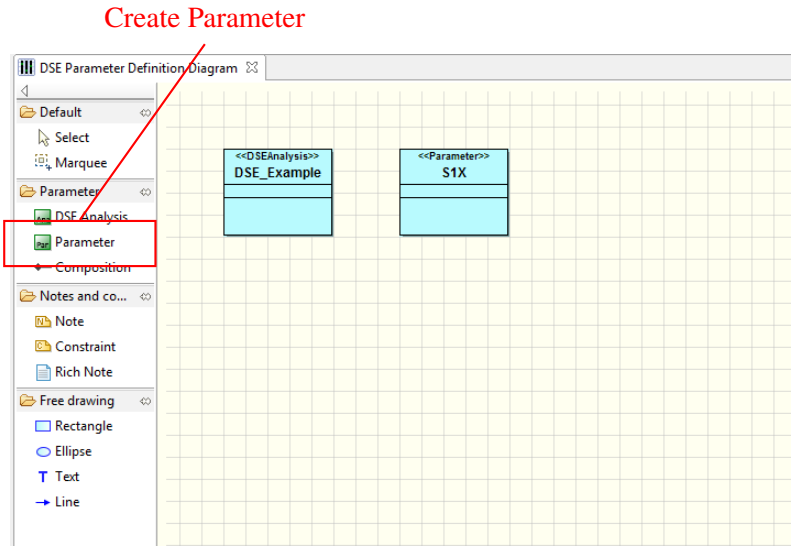
4 Defining Parameters

This section will present how to define the parameters that will be varied during a DSE. In the first part we define the parameters in terms of their name and a set of values that we wish to test. The second step here is to connect the defined parameters to the parameters of the FMUs in the multi-model that we actually want to vary.
If we multiply the cardinality of the set of values for each parameter then we can find the size of the design space. It is important to keep the design space size in mind since this, along with the time required to run each simulation, may be used to determine if a DSE will complete in a reasonable time.

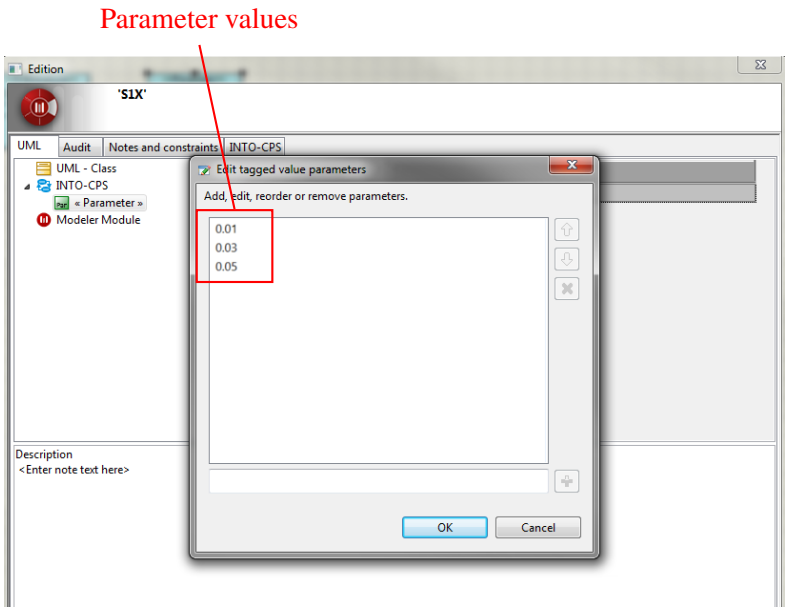
Step 23. Right click '$linefollowrobot\_mm$' in model outline. Select 'INTO-CPS > Parameter Definition Diagram'.

Step 24. Find the '$DSE\_Example$' block in the model outline and drag it onto the empty diagram to add the DSE block created in Part 2.

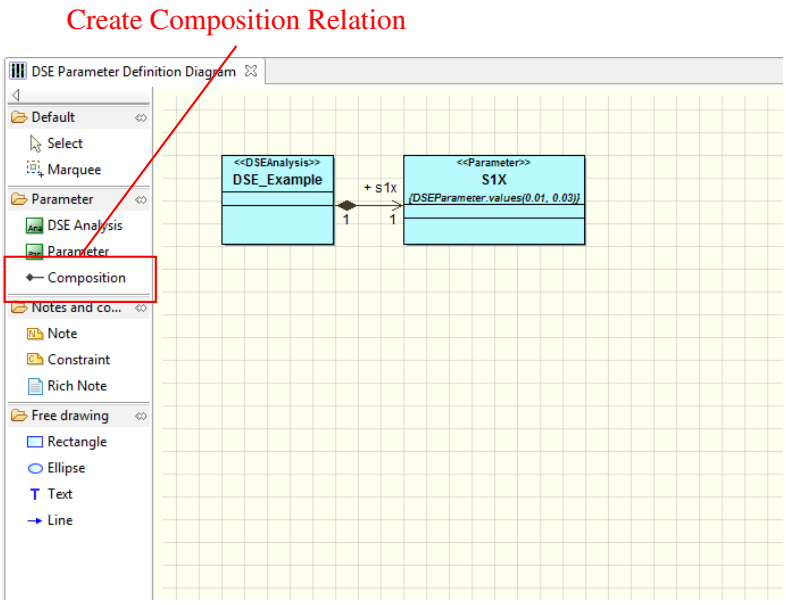
Step 25. From the *Toolbox*, select *Parameter* and add it to the diagram. Rename the new Parameter block '$S1X$'.



Step 26. Double click the Parameter block and select ‘INTO-CPS > Parameter’. Click next to {}values to add parameter values. Enter the value ‘0.01’ and click ‘+’ to submit the value. Repeat this to add the values ‘0.03’ and ‘0.05’, then click *OK*. Click *Close* to return to the diagram.



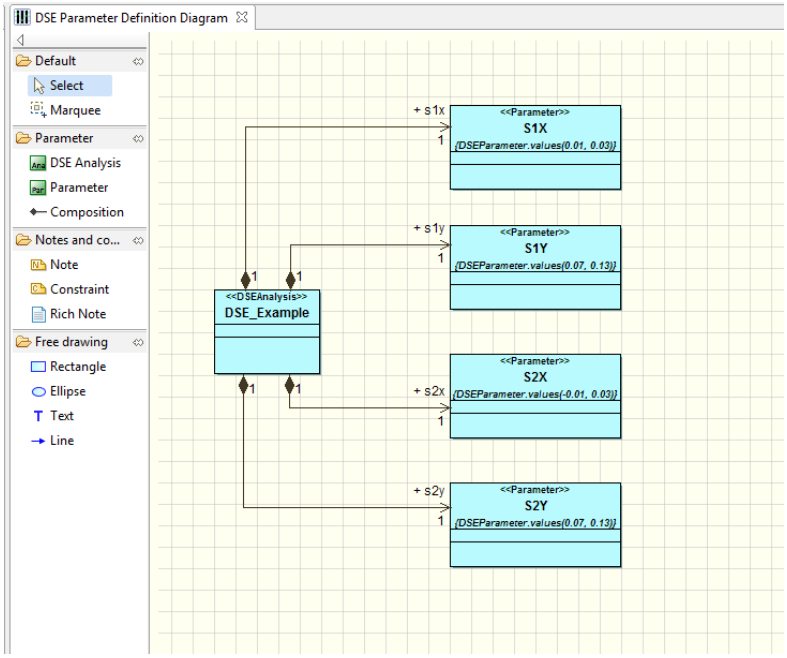
Step 27. To associate the parameter with the DSE, select the *Composition* tool from the *Toolbox*, click first on the ‘$DSE\_Exampleblock$’, and then on the new ‘$S1X$’ block.



Step 28. Repeat Step 25. – Step 27. to associate the following parameters with the DSE:

- Name: '$S1Y$', Values: '$0.01, 0.07, 0.13$';
- Name: '$S2X$', Values: '$-0.01, -0.03, -0.05$';
- Name: '$S2Y$', Values: '$0.01, 0.07, 0.13$';

The model should now look like this:

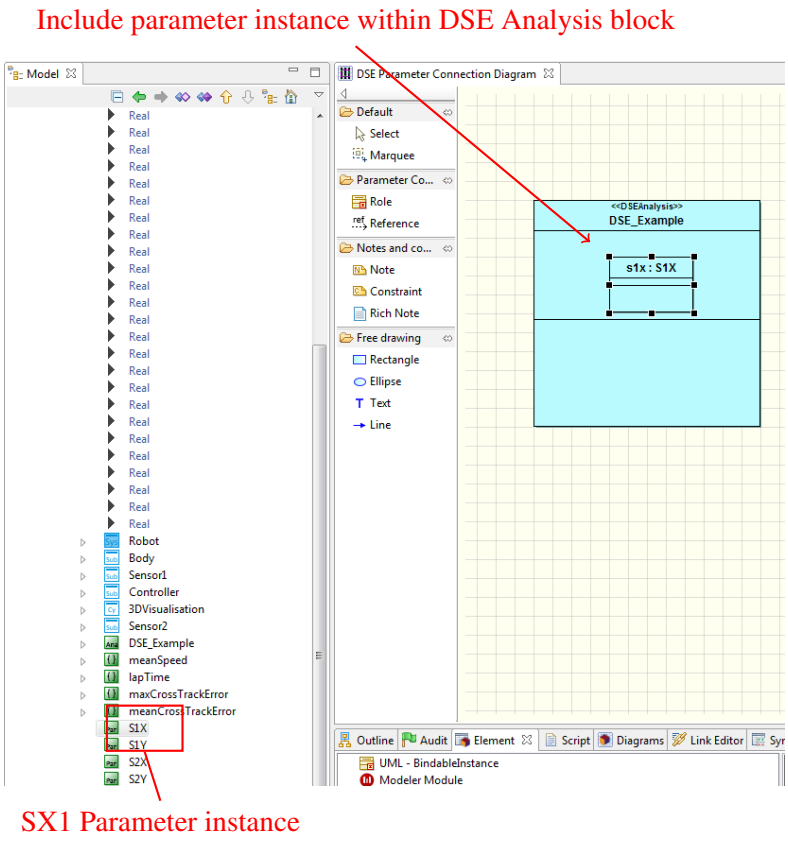


5 Connecting Parameters

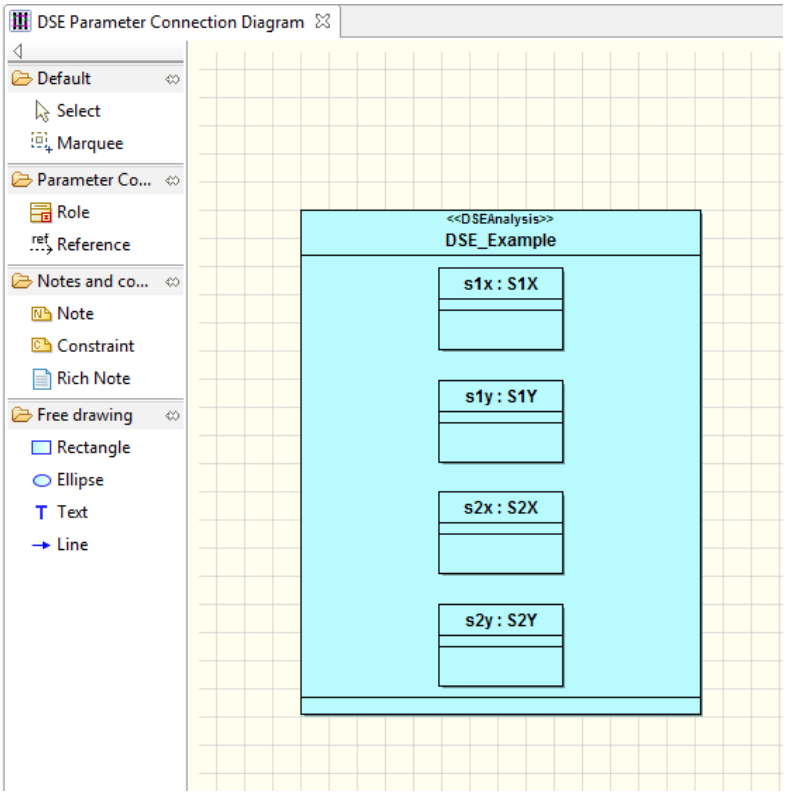
Step 29. Right click '$linefollowrobot\_mm$' in model outline. Select 'INTO-CPS > Parameter Connection Diagram'.

Step 30. Find the '$DSE\_Example$' block in the model outline and drag it onto the empty diagram to add the DSE block created in Part 2.

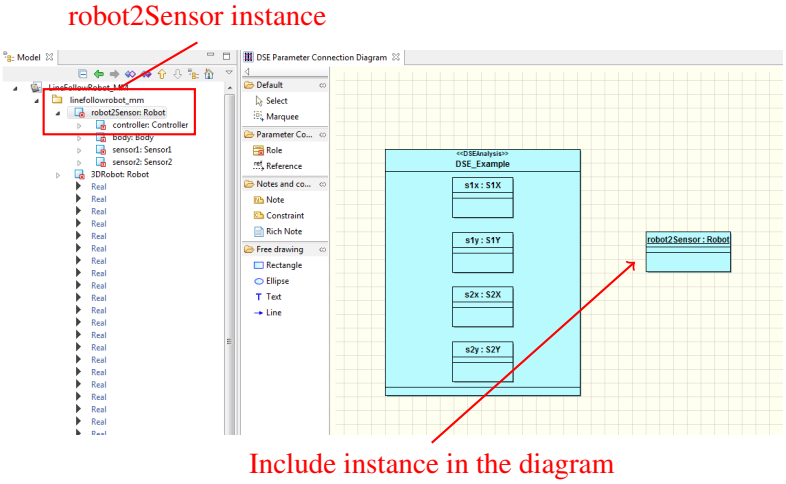
Step 31. Find the '$S1X$' block in the model outline and drag it onto the DSE example block now visible on the diagram. Double click the block that this creates and name the instance '$s1x$'.



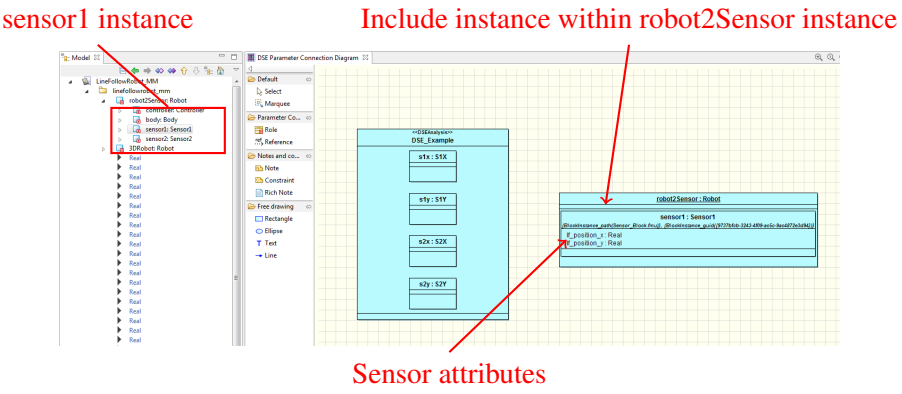
Step 32. Repeat Step 31. for '$S1Y$', '$S2X$', and '$S2Y$', using the instance names '$s1y$', '$s2x$', and '$s2y$' respectively.



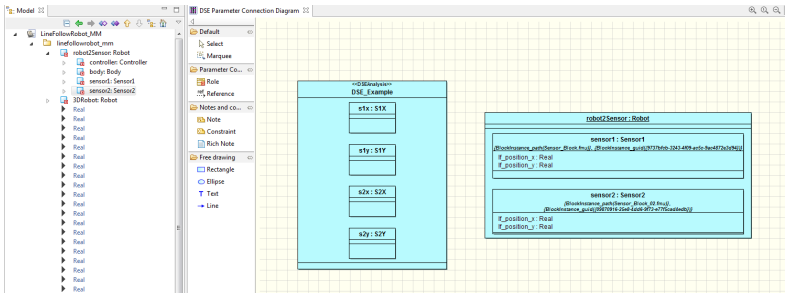
Step 33. To link Parameters to the multi-model, find the '$robot2Sensor$' instance in the model overview and drag it onto the diagram to include it in the DSE.
(Note drag the block onto empty space, not onto the DSE_Example block).



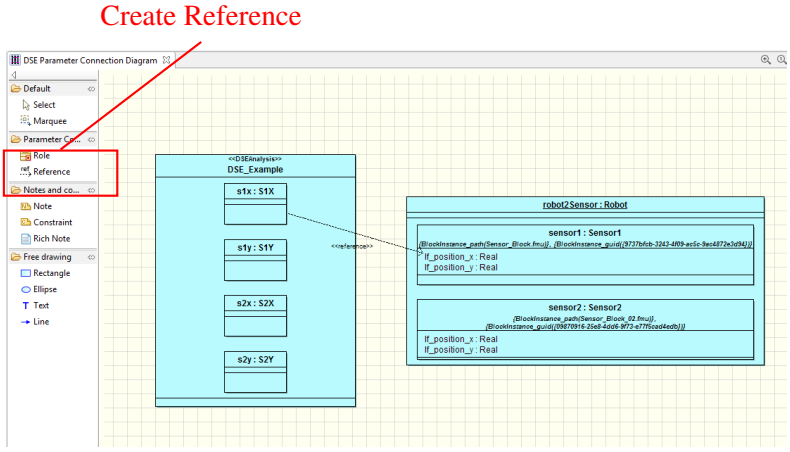
Step 34. Expand '$robot2Sensor$' in the model overview and drag the nested instance of '$sensor1$' onto the '$robot2sensor$' instance just created. This block should include the attributes '$lf\_position\_x$' and '$lf\_position\_y$'.
(NB depending on the model version, $sensor1$ may be labelled $sensor1FMU$).



Step 35. Repeat Step 34. to include '$sensor2$' in the '$robot2sensor$' block.



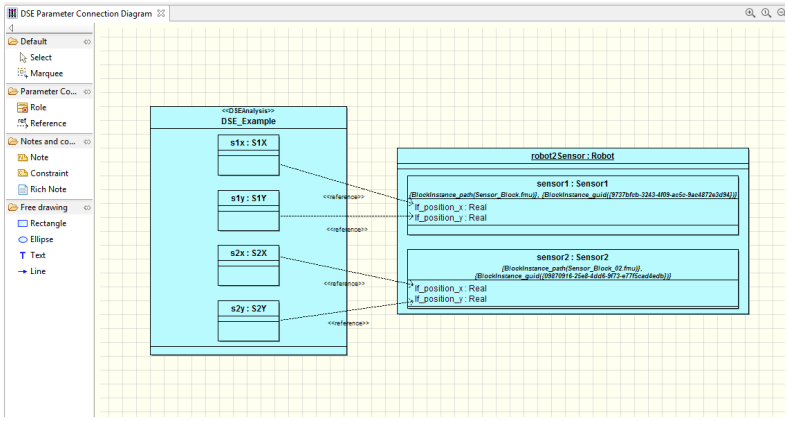
Step 36. Finally, select the *Reference* tool from the *Toolbox* and link parameter '$s1x$' to attribute '$lf\_position\_x$' from '$sensor1$'.



Step 37. Repeat Step 36. to link:

- '$s1y$' to '$sensor1 > lf\_position\_y$';
- '$s2x$' to '$sensor2 > lf\_position\_x$';
- '$s2y$' to '$sensor2 > lf\_position\_y$';

The model should now look like this:



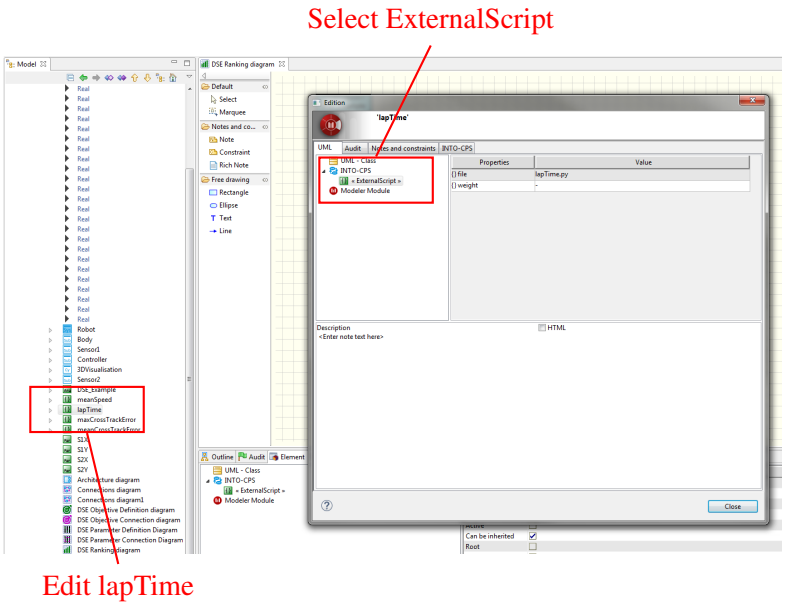
6 Ranking Results

In DSE all the designs are essentially competing to be considered the best and in this section will present how to define the way in which competing designs are compared. Here we are going to declare which of the objectives should be used to compare competing designs and whether there is a preference for lower or higher values for each of the objectives.

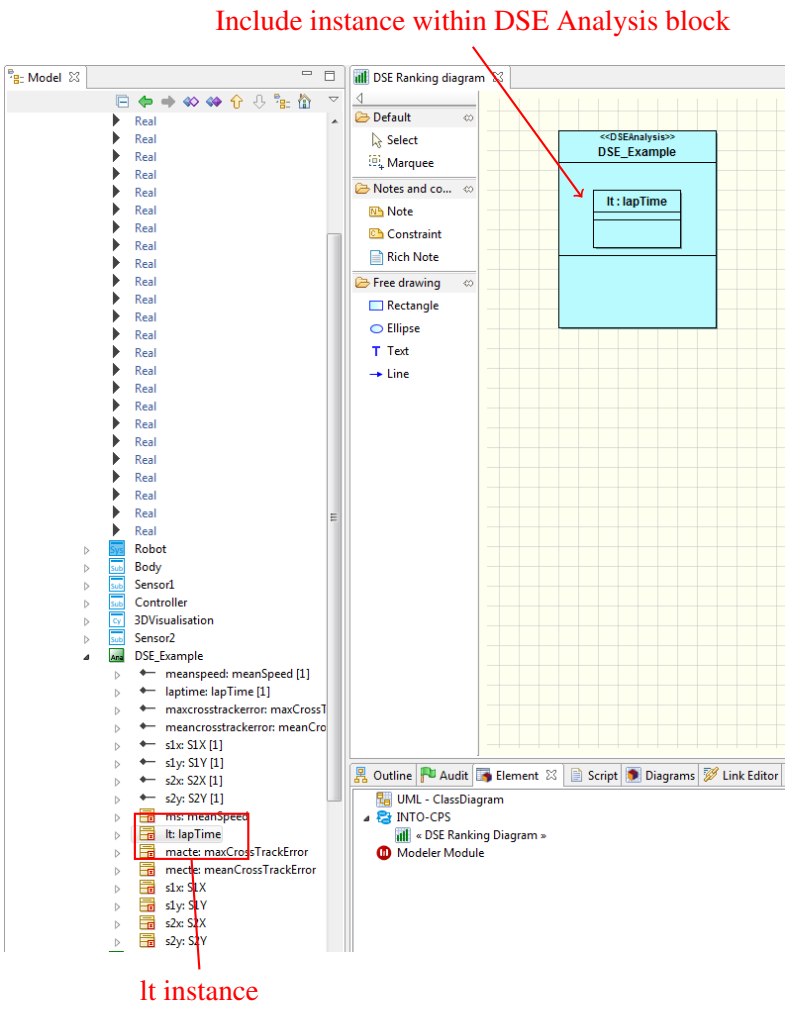
Step 38. Right click '$linefollowrobot\_mm$' in model outline. Select 'INTO-CPS > Ranking Diagram'.

Step 39. Find the '$DSE\_Example$' block in the model outline and drag it onto the empty diagram to add the DSE block created in Part 2.

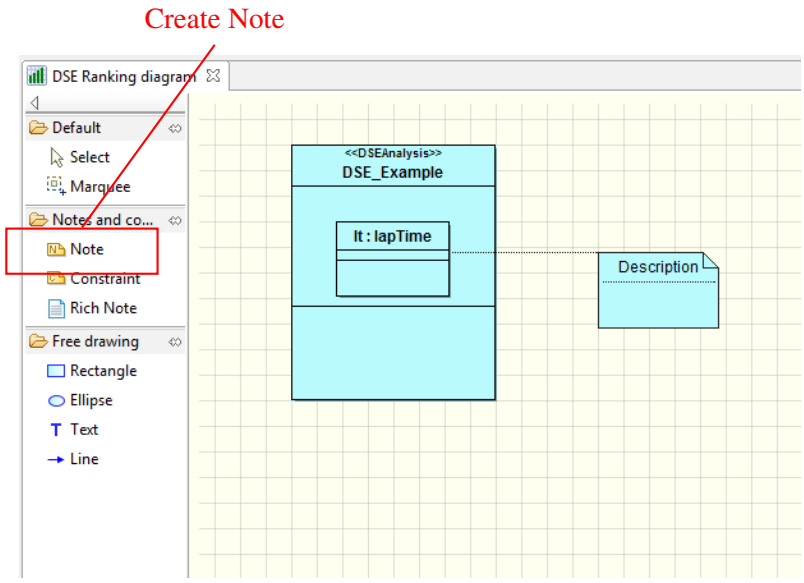
Step 40. Double click on the '$lapTime$' objective in the model outline. Click 'ExternalScript' nested under 'INTO-CPS'. Assign the value '$-$' to {}weight and close the editor window.



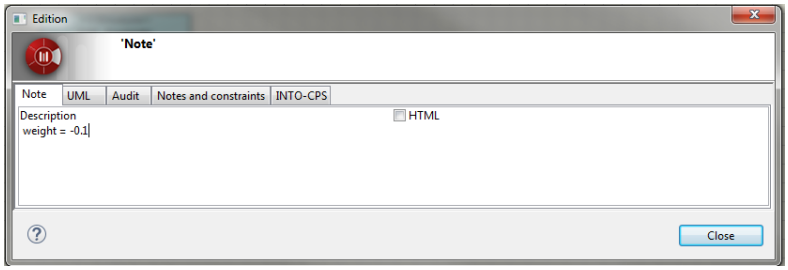
Step 41. Expand the '$DSE\_Example$' block in the model outline and drag the nested instance 'lt' onto the DSE example block now visible on the diagram.



Step 42. Select the Note tool from the toolbox and click on the '*lt*' block just created. Then click on a clear space in the diagram to create the description block.

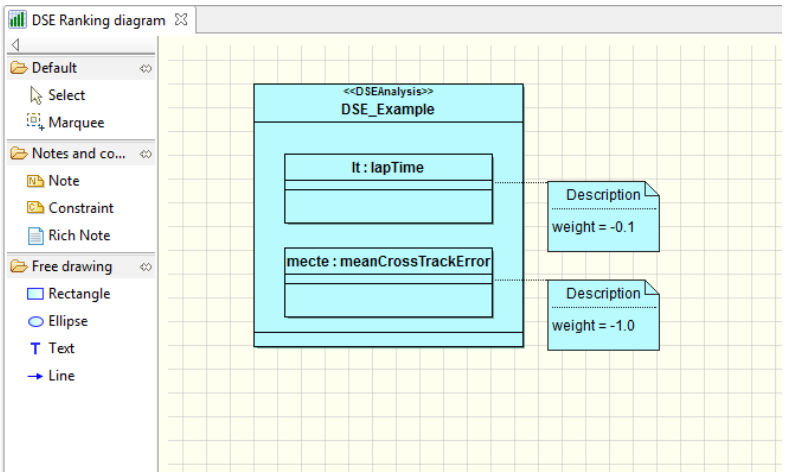


Step 43. Double click the description block to edit the note text. Uncheck the 'HTML' option and paste the text '$weight = -0.1$'. Close the description window to update the diagram.



Step 44. Repeat Step 40. – Step 43. to set the {}weight property of '$meanCrossTrackError$' and add the '$mecte$' objective instance with the description '$weight = -1.0$'.

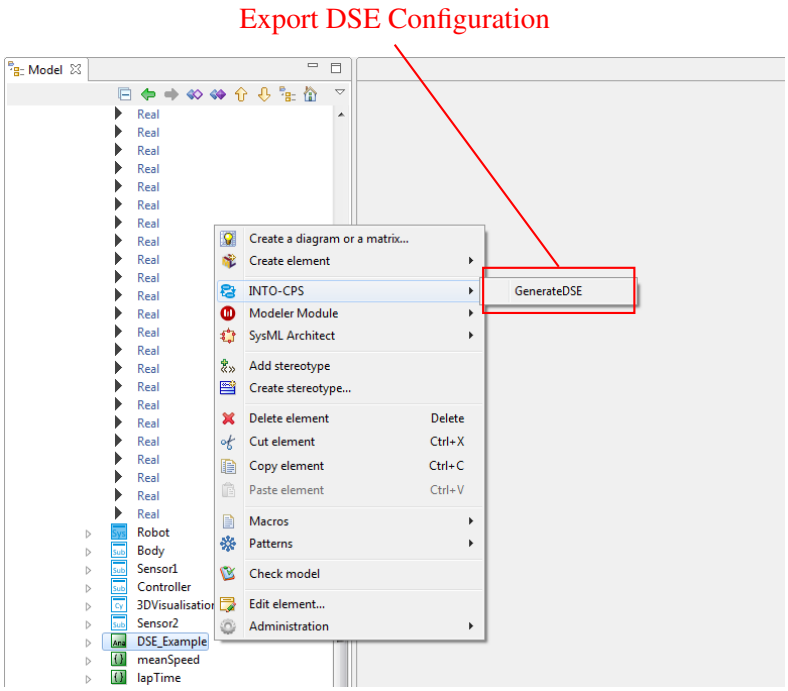
The model should now look like this:



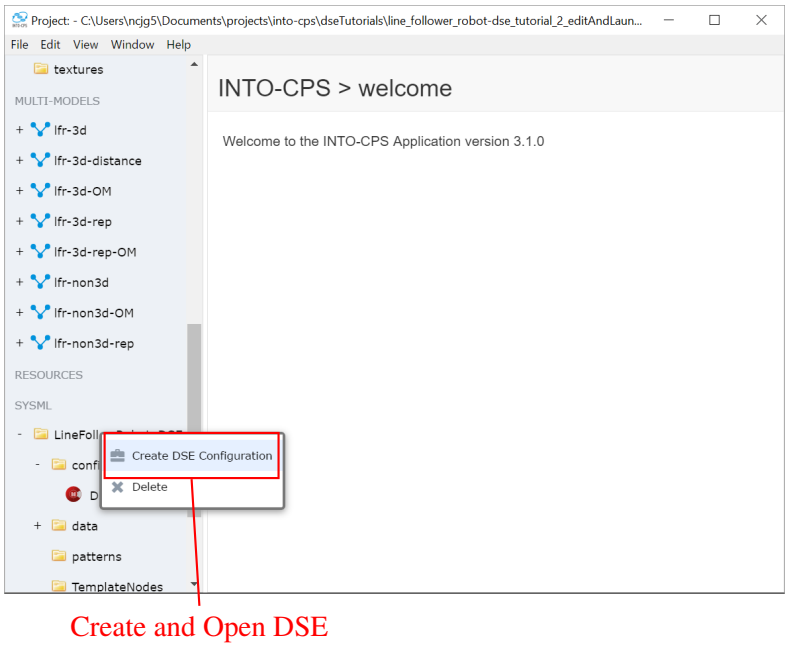
7 Exporting DSE Configuraton

This section will outline the steps necessary to generate a DSE configuration in Modelio and include the file to the INTO-CPS app.

Step 45. Find the '$DSE\_Example$' block in the model outline and right click. Select 'INTO-CPS > GenerateDSE'. Click *Export* then *OK*.



Step 46. In the INTO-CPS app, open the project in Tutorial 6. Then the configuration file generated in the previous step can be found in 'SysML > LineFollowerRobot > config'. Right click the configuration file and select 'Create DSE Configuration'.



Create and Open DSE

The configuration will be moved to the DSE directory and opened. The DSE can now be viewed, edited, or started. This will be covered in Tutorial 7.