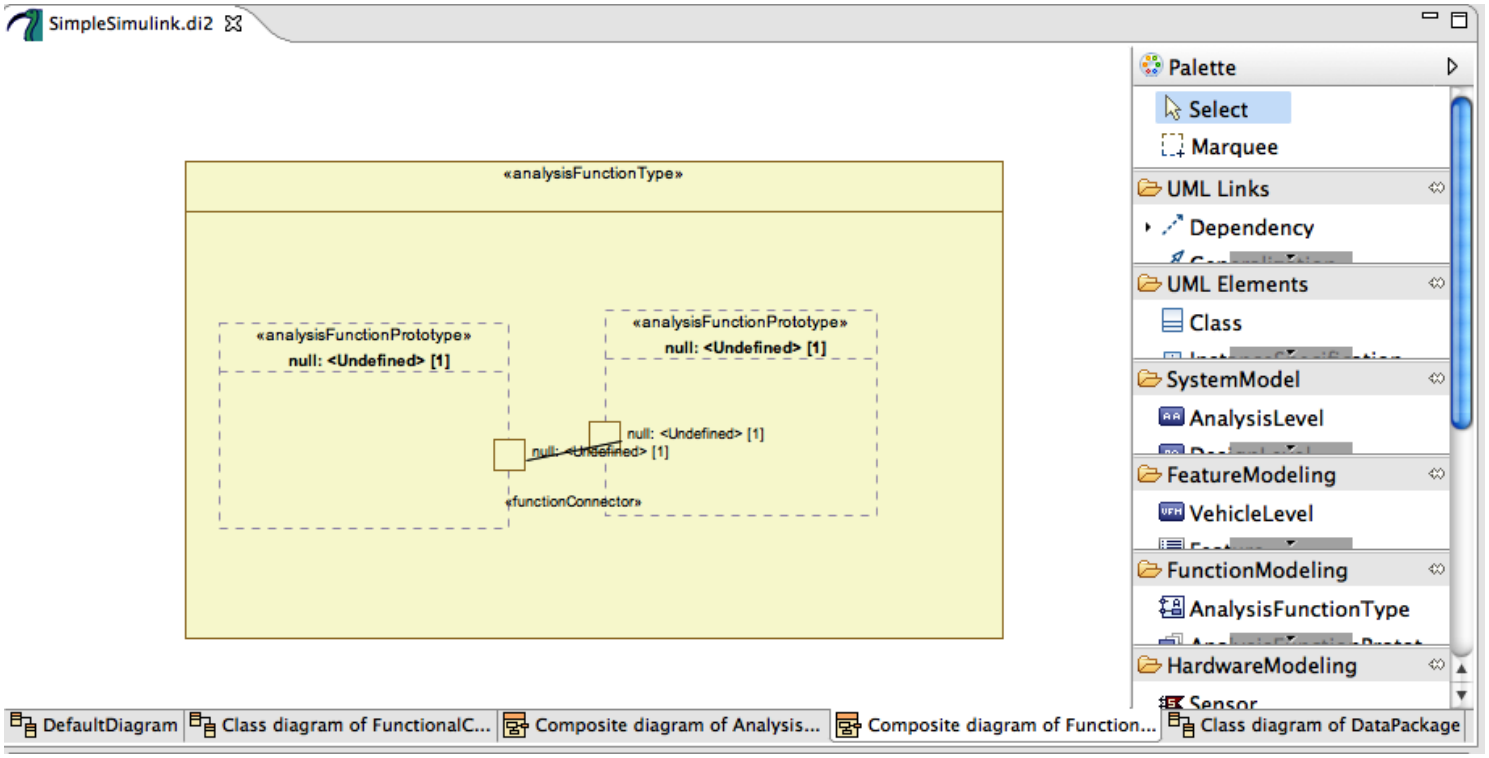
SimulinkExchange

Setup and overview

EAST-ADL environment within Eclipse

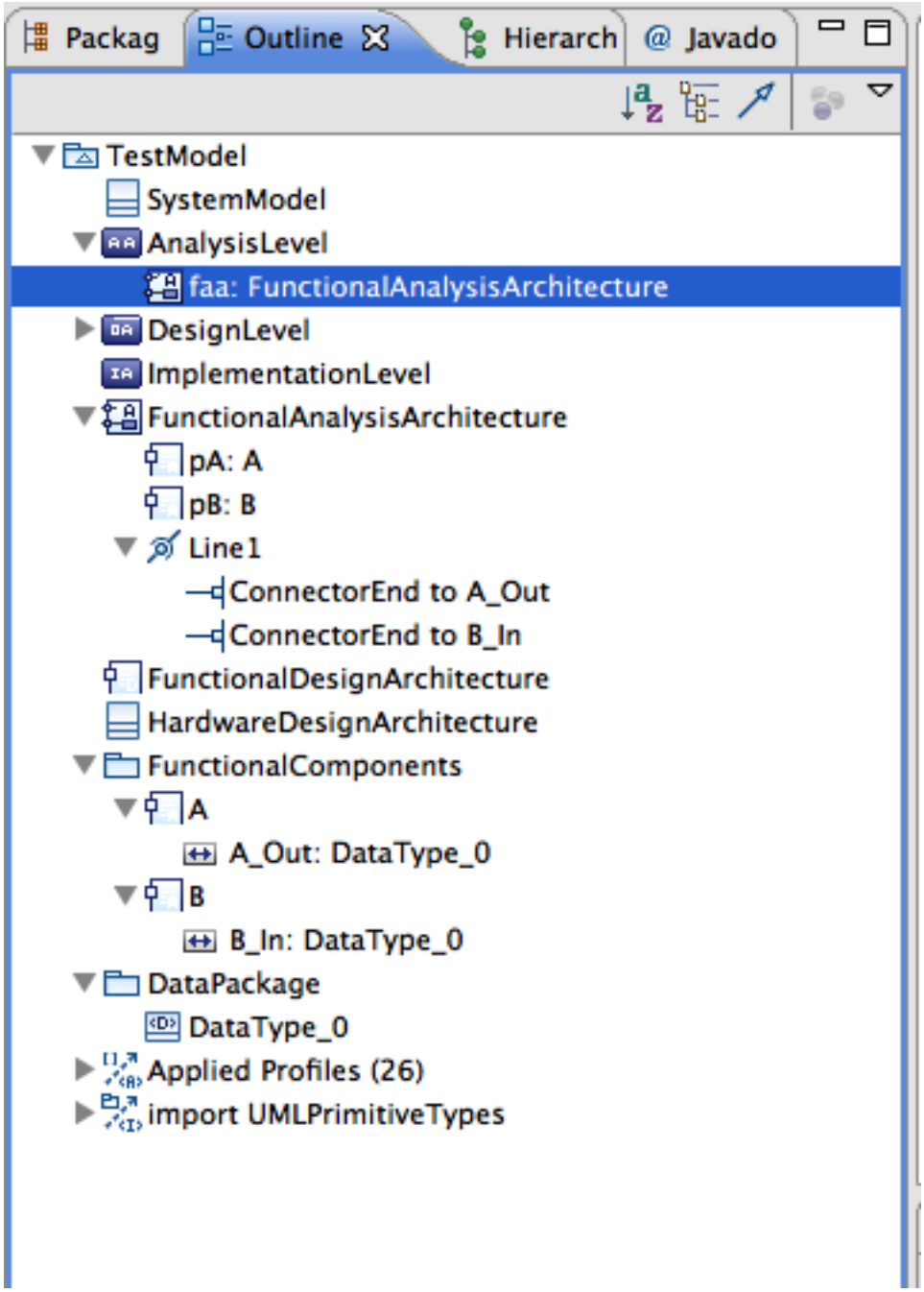
Usgin the Papyrus perspective within Eclipse you can explore and edit EAST-ADL models.

Papyrus perspective



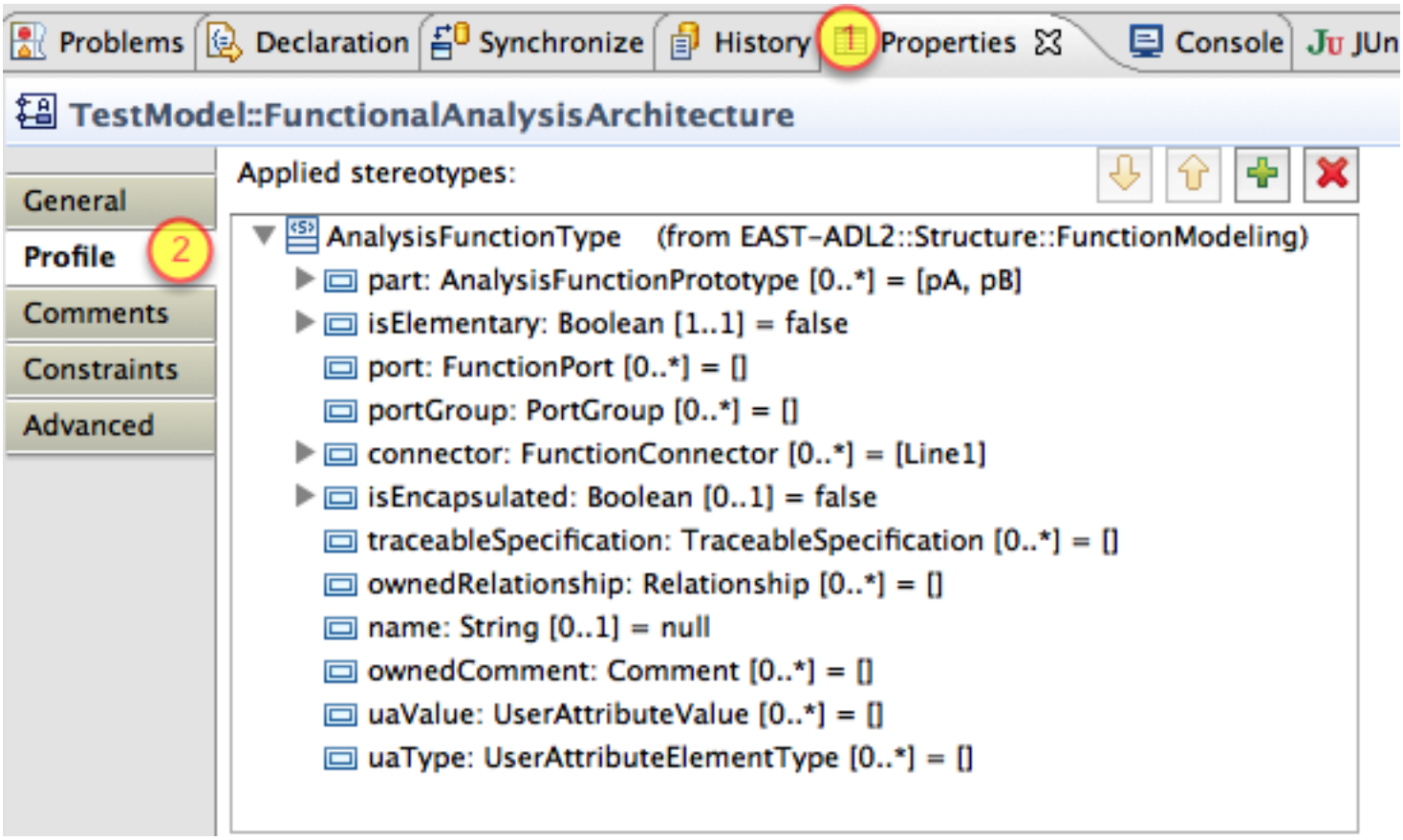
Here, a composite diagram of the Functional Analysis Architecture

Outline view



The outline view is probably the easiest way to navigate models.

UML profiles

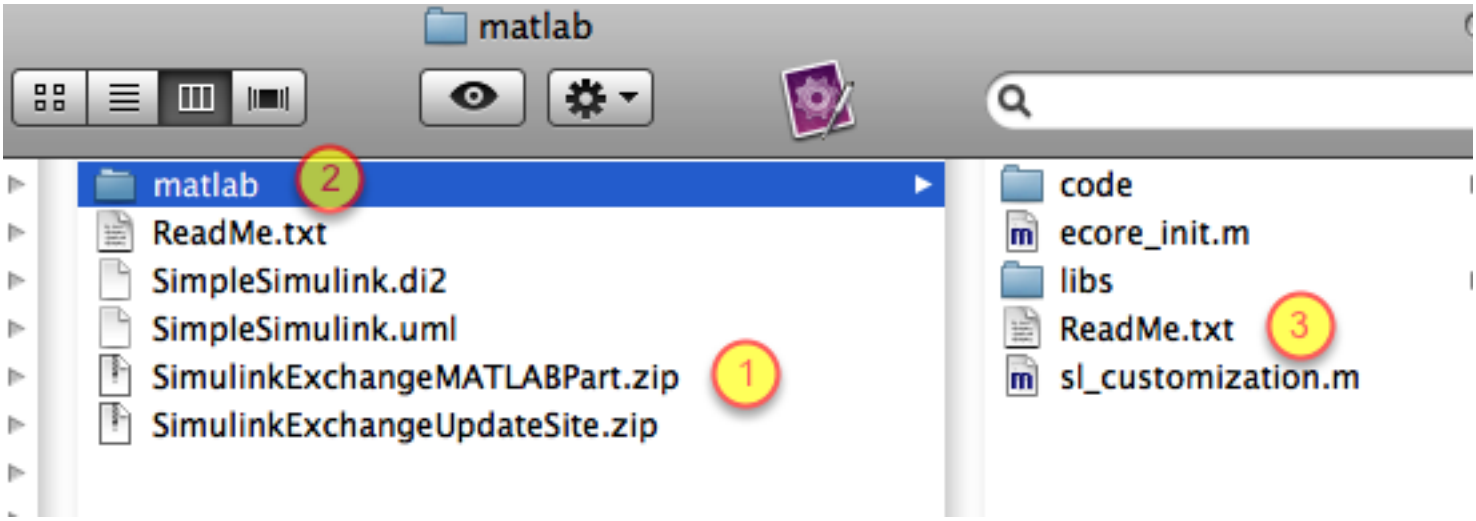


The properties view allows you to see which profiles are applies to elements. Most errors can be found by looking at this view.

MATLAB setup

Setting up the MATLAB environment for the SImulink Exchange plugin

Files needed



1) Unzipping the file SimulinkExchangeMATLABPart.zip will give you a matlab folder.
2) Open MATLAB and change to this newly unzipped directory.
3) Please do read the ReadMe.txt file as it contains workarounds to known problems.

Initiliaze the environment

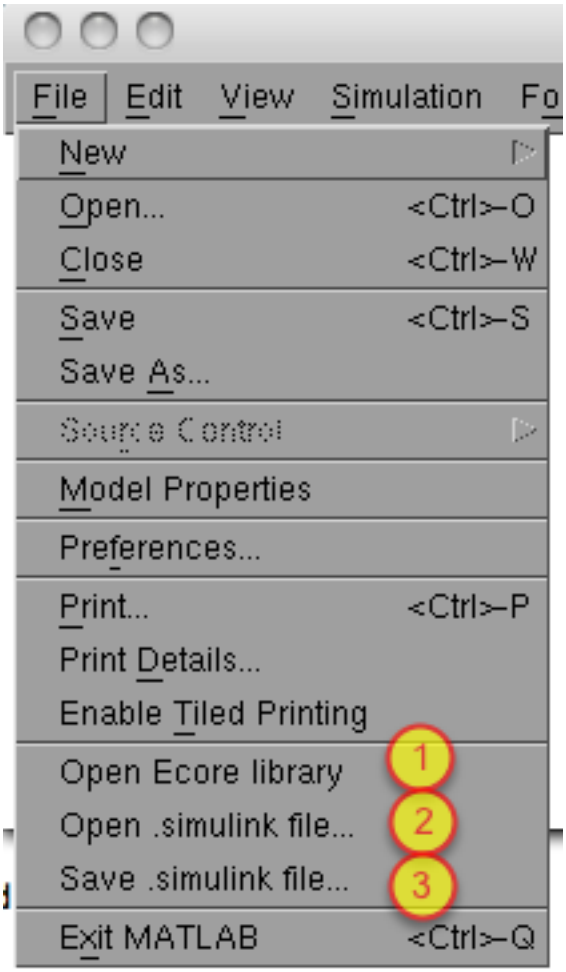
```
>> ecore_init
Welcome to the Simulink to Ecore exchange plugin, developed by KTH in the ATESST2 project.
It is licenced under the Eclipse Public License 1.0.
Please read the documentation!
>>
```

Run the command ecore_init from the console. The result is shown above.

Additions to the Simulink environment in MATLAB

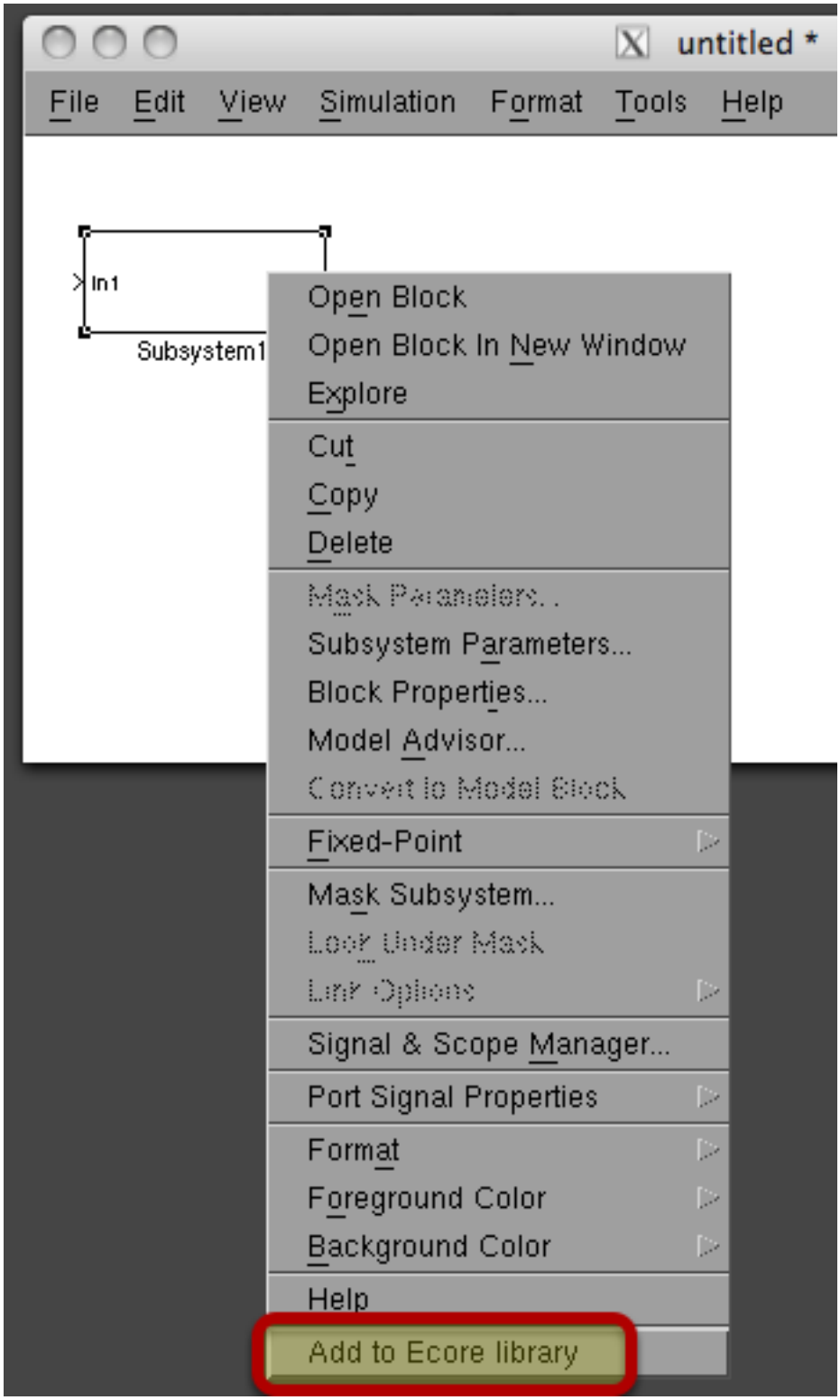
The new items under File menu.

New Menu items



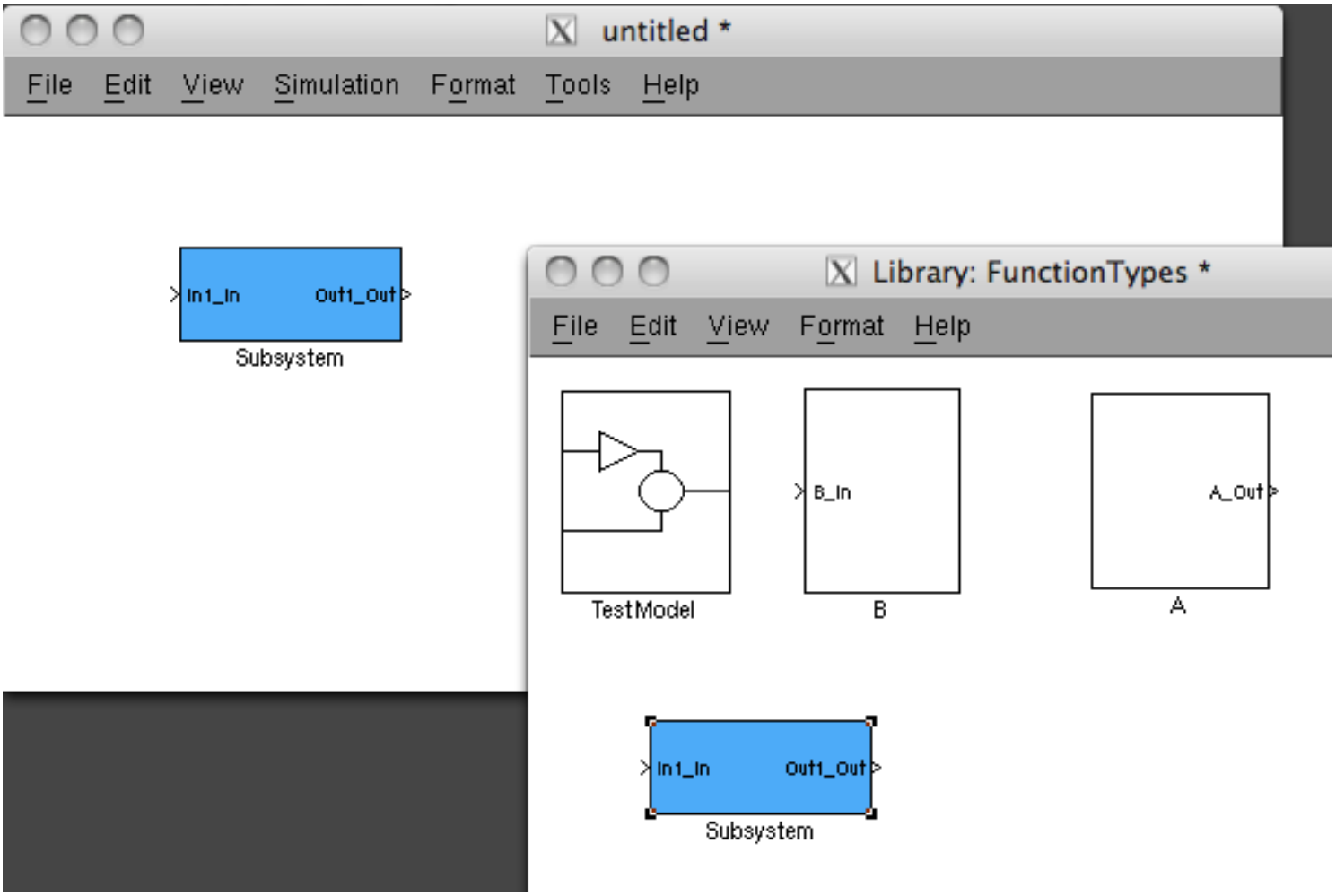
If you open a new simulink model, under the file menu you'll see three new items.
1) Shows a simulink library with the blocks converted so far. Note that EAST-ADL FunctionTypes are stored as library elements. This mdl file is located within the code folder.
2) Opens the intermediate file created by Eclipse, and converts it into simulink elements.
3) Converts a simulink model an the intermediate representation suitable to be used with Ecipse.

Marking blocks for conversion



Only blocks present in the Ecore library (FunctionTypes.mdl) will be converted to EAST-ADL.
To convert a block, right-click on it and select "Add to Ecore library". Its color will change to blue.

A block marked for conversion



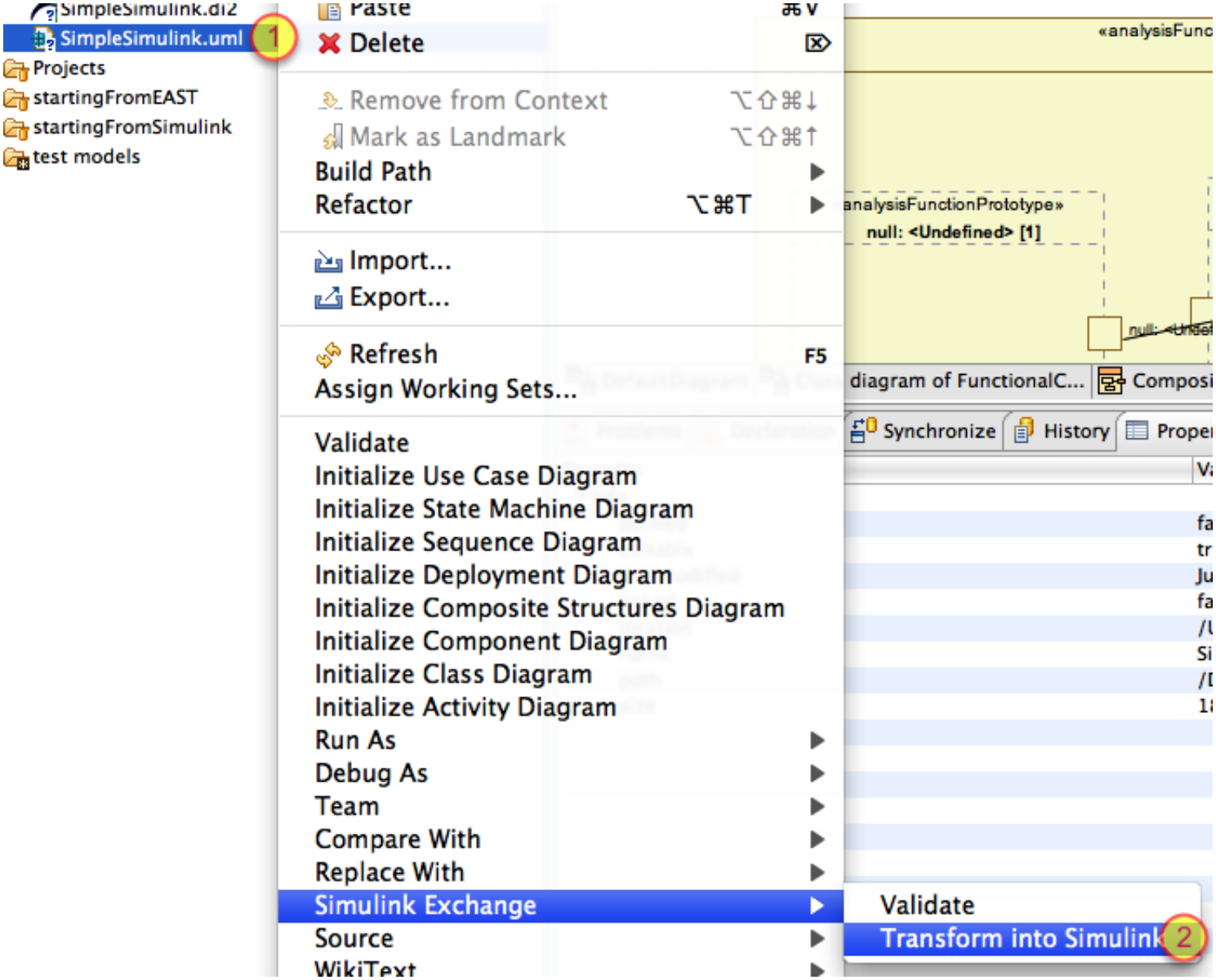
This is the result of the previous operation. The block named "Subsystem" has been added to the library. Its color changes to blue.
The block present in the untitled model is actually an instance of the library element with the same name.

Converting from EAST-ADL to Simulink

Converting from EAST-ADL to intermediate format .simulink

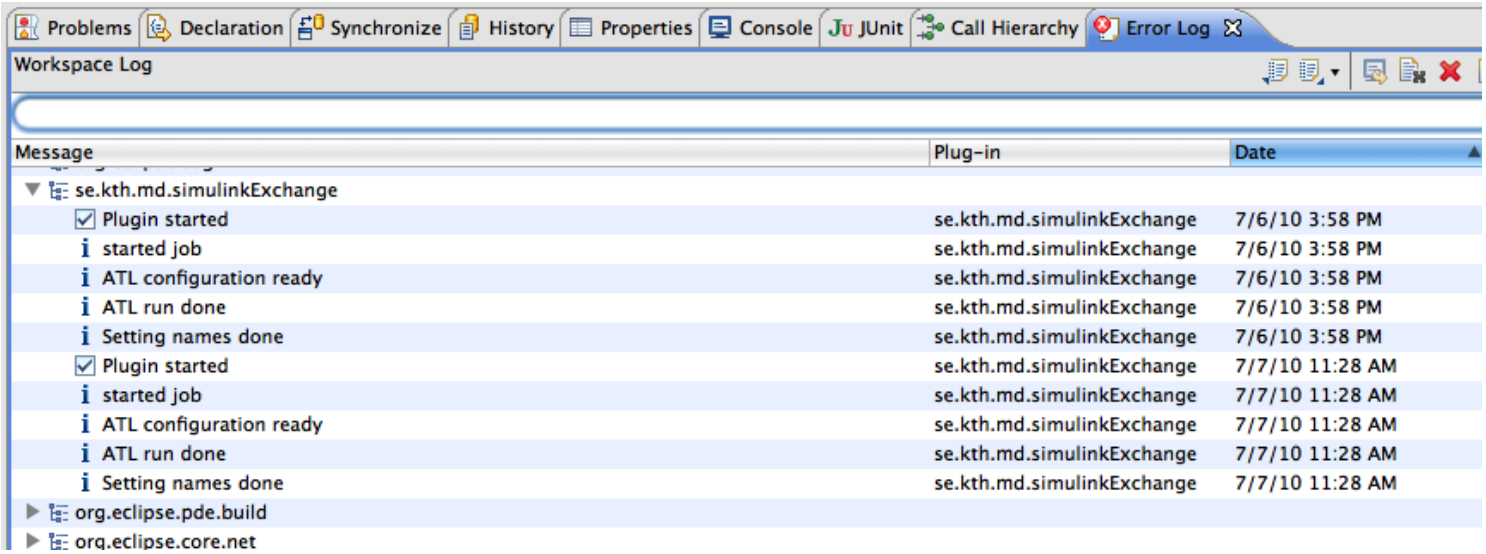
How to convert an EAST-ADL model to the intermediate representation .simulink.

Starting a transformation



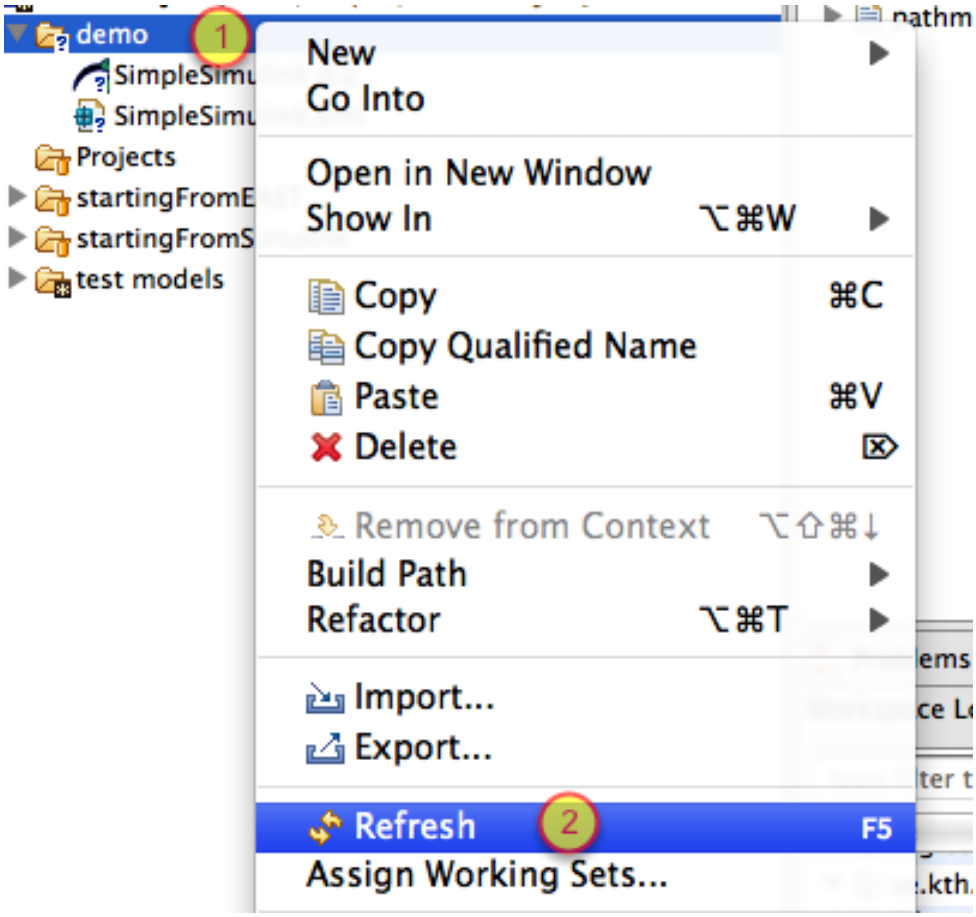
Right-click on a UML file to get a context pop-up menu with the transformation option.

Checking for problems



The Error Log view contains some information on the status of the process.

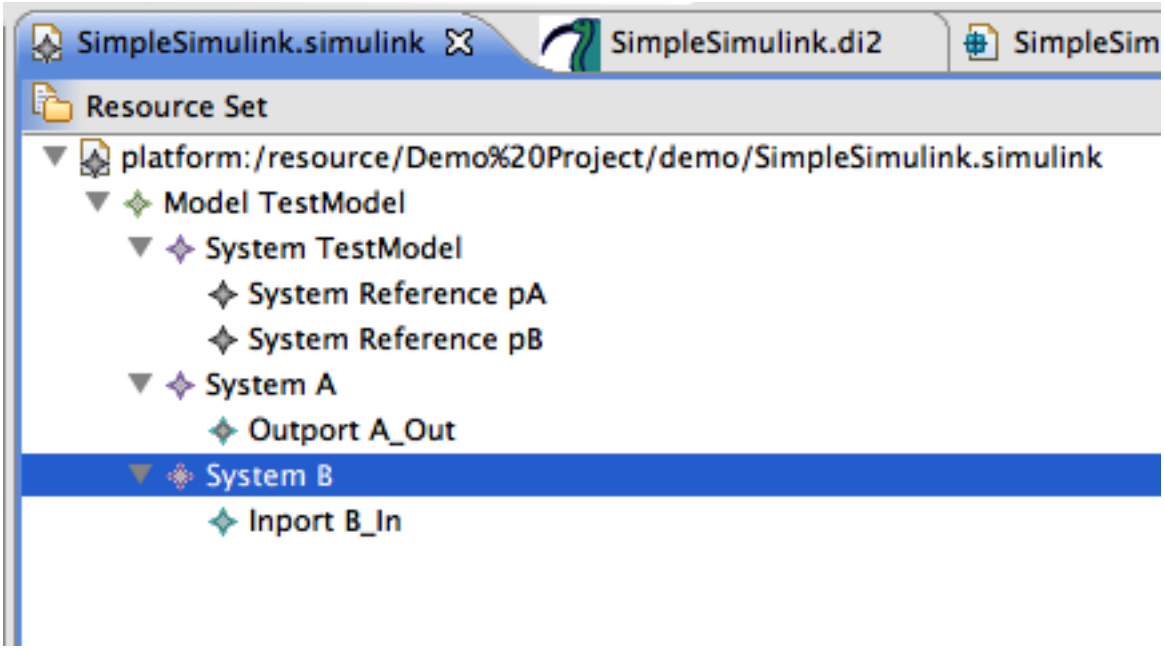
See the results



The Eclipse workspace has to be refreshed before you can see the newly created file with the extension .simulink.

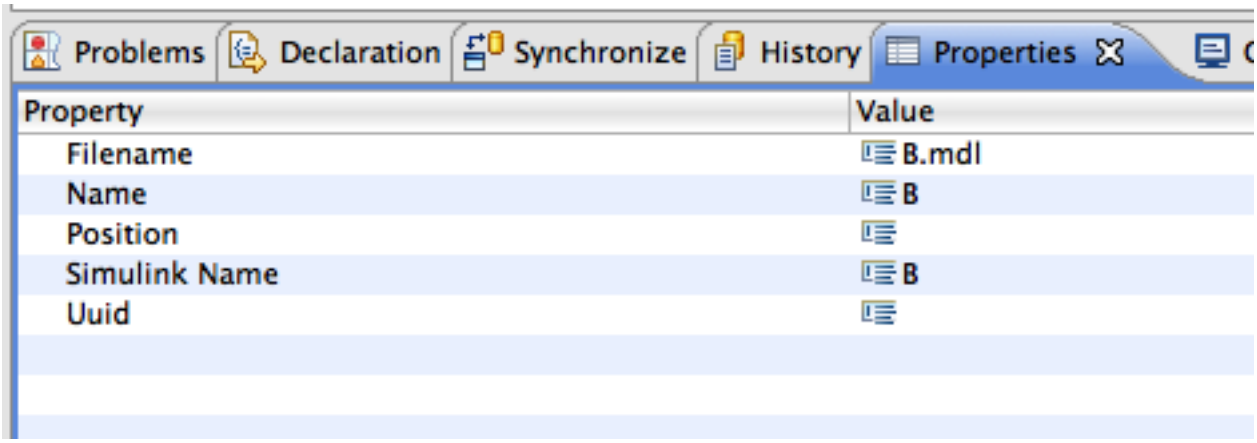
The process continues from within MATLAB.

EMF reflective editor



The editor opens with a double-click on a file with the simulink extension.
The model can be modified within this editor.

Examining elements

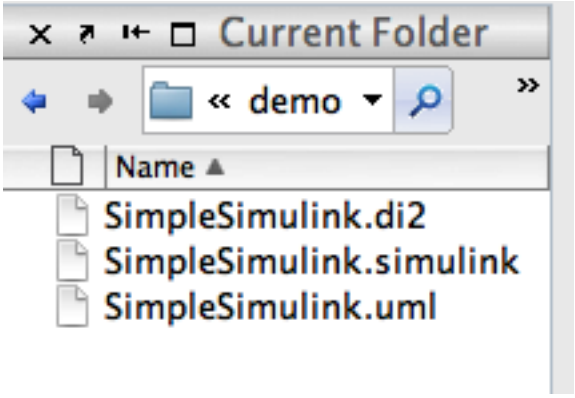


The properties view shows more information on the individual items.

Converting from intermediate format .simulink to MATLAB

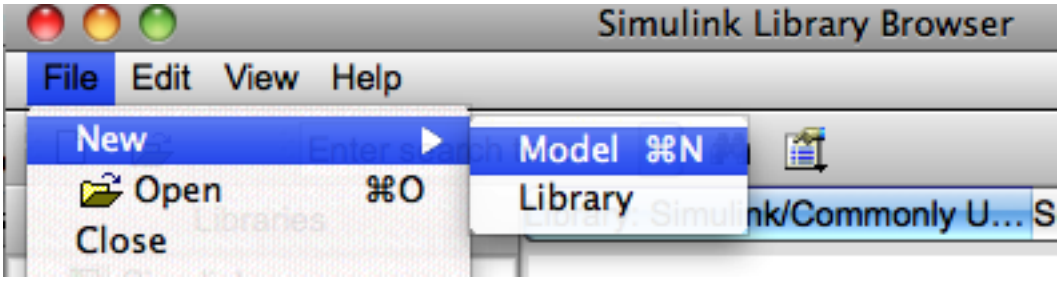
How to convert a model in an intermediate representation to a simulink model.

Change to proper folder

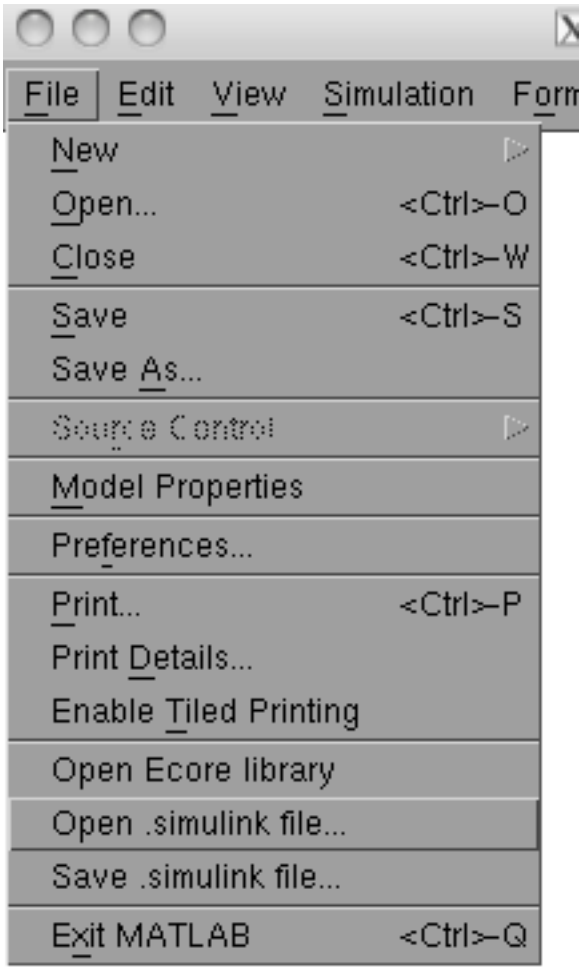


The conversion process creates files, so be sure to start the operation from a writeable folder. In this case, the output folder will be the same folder where the intermediate file is located.
Tip: If you choose the Eclipse project folder, where the UML file is located, you can save an extra step when you are converting back from MATLAB to Eclipse.

Create a new simulink model

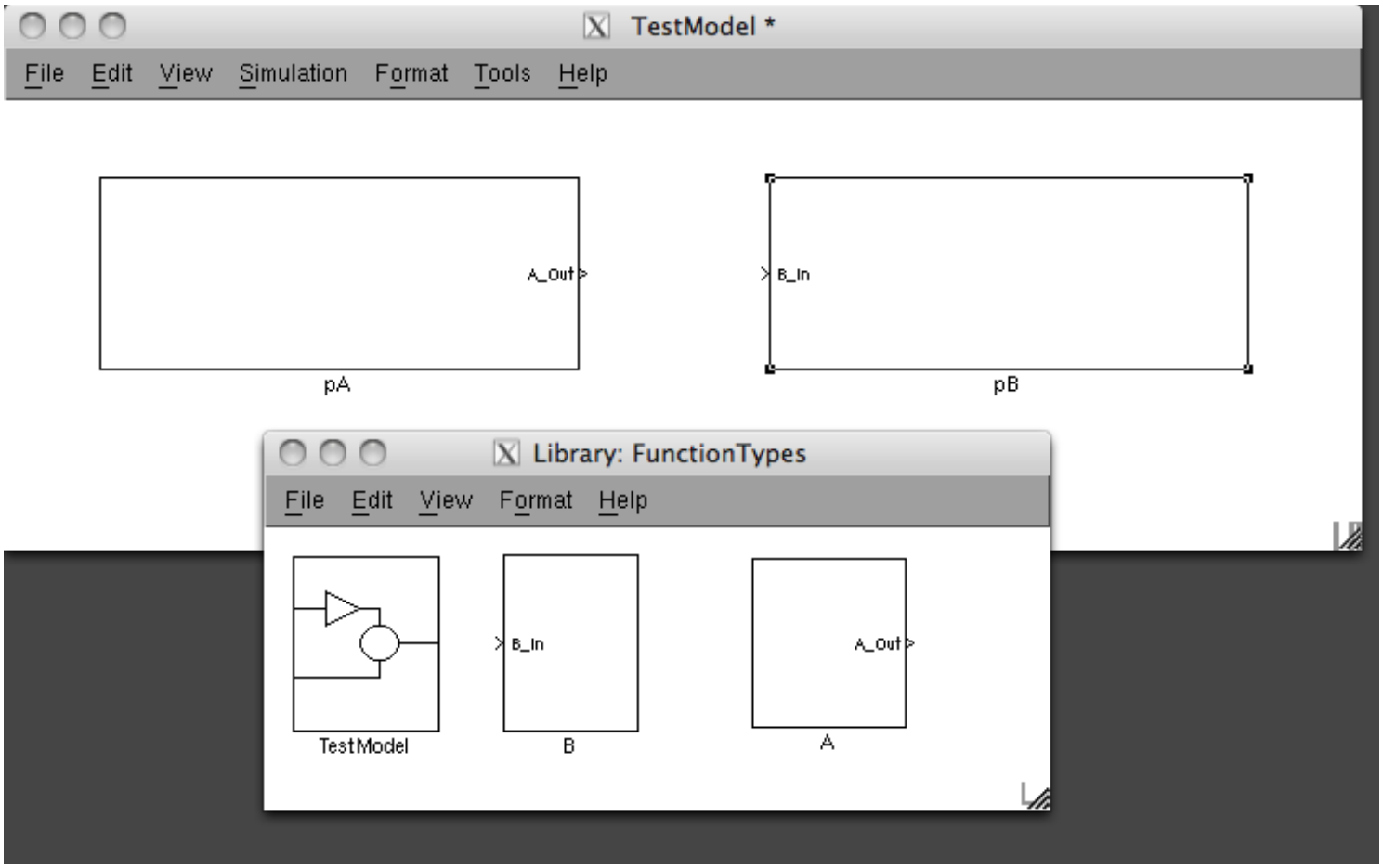


Open a simulink file



From the file menu, open a simulink file.

The result

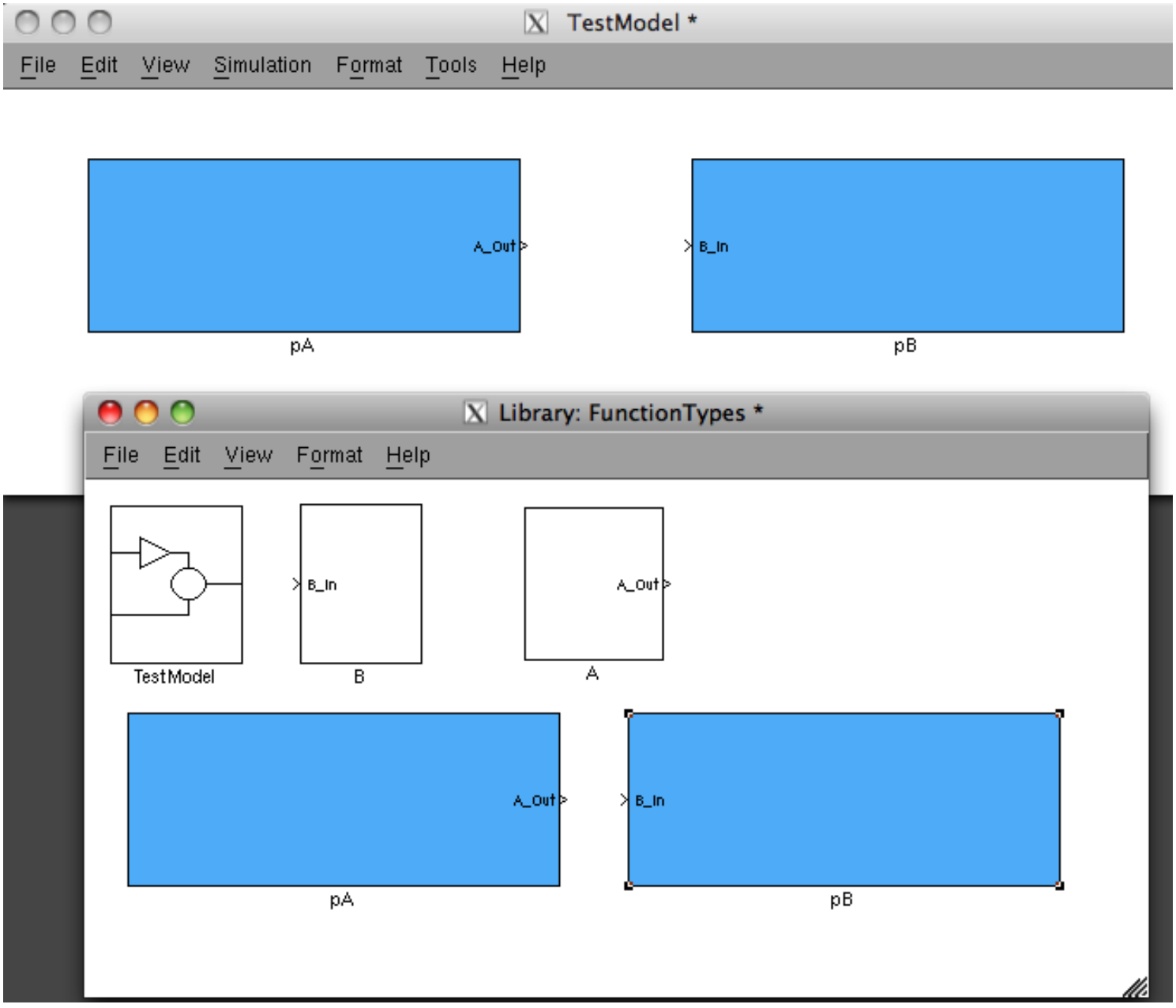

TestModel *
File Edit View Simulation Format Tools Help
A_Out
pA
B_In
pB
Library: FunctionTypes
File Edit View Format Help
TestModel
B_In
B
A_Out
A

Converting from Simulink to EAST-ADL

Converting from MATLAB to intermediate format .simulink

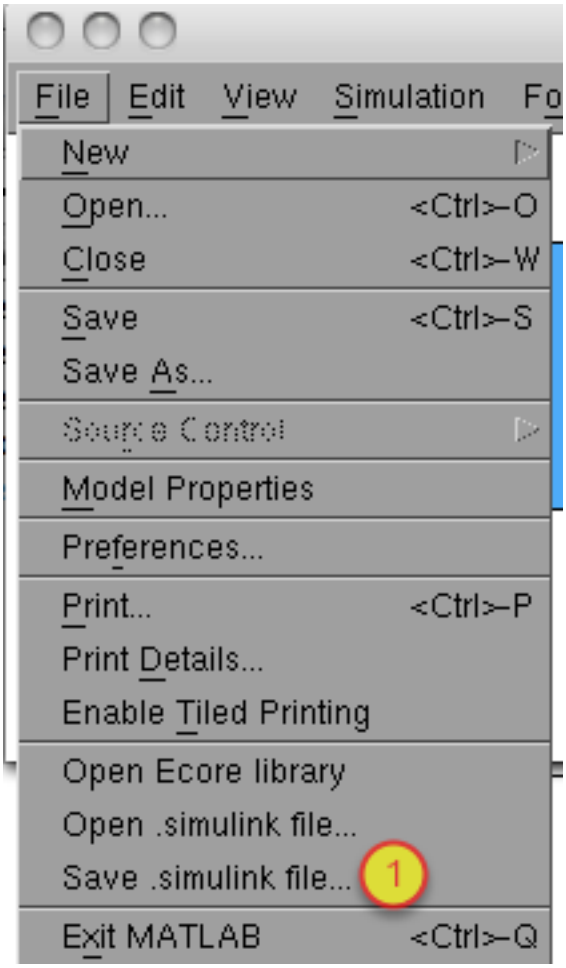
How to convert a simulink model into an intermediate file (with extension .simliunk), suitable for conversion into EAST-ADL.

Mark blocks for conversion



In this example, both blocks are selected for conversion: pA and pB.
For more details on how to mark blocks see: Marking blocks for conversion.
Note: I do not know how to unmark blocks for conversion.

Save in intermediate format



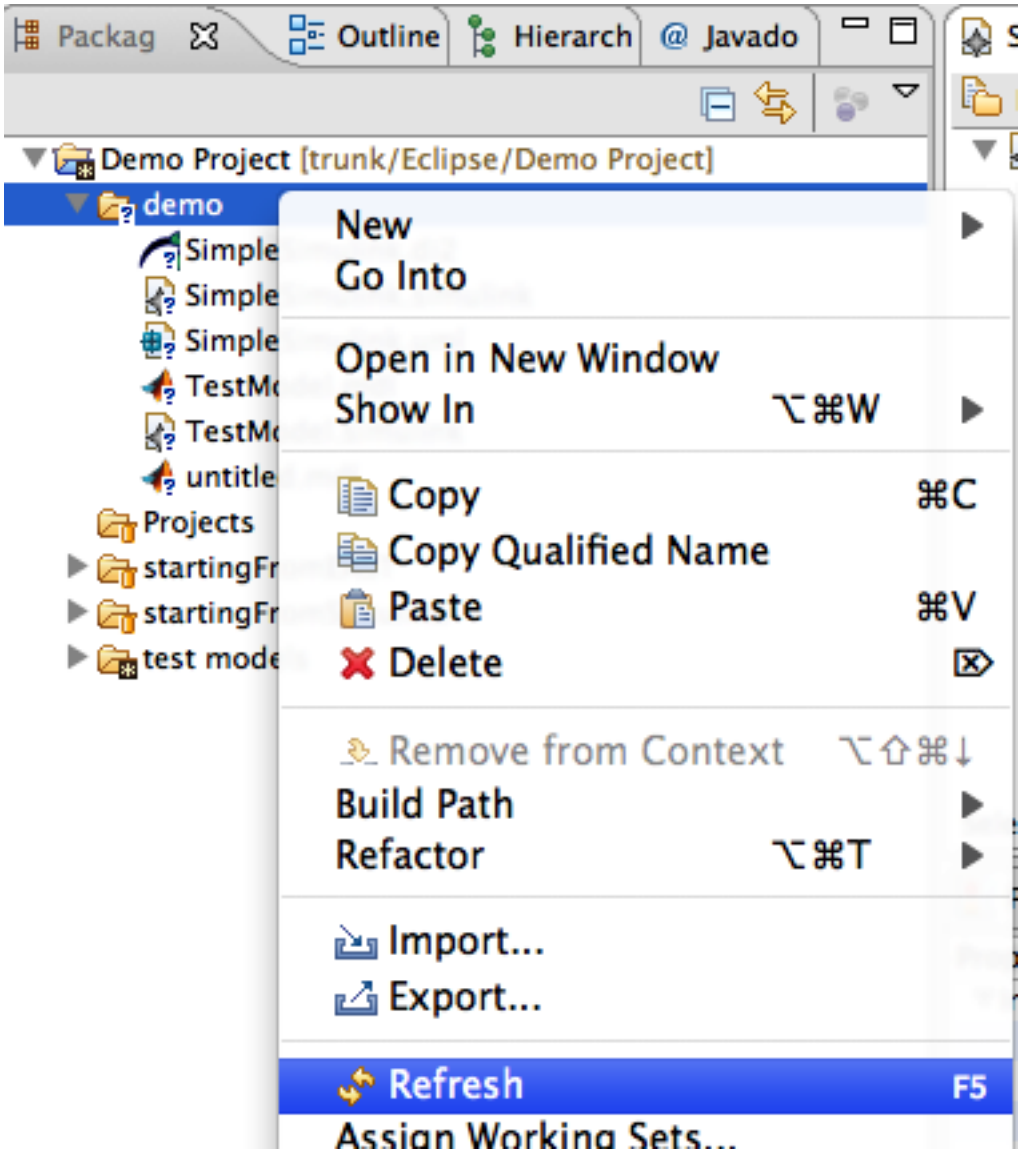
Save the file.

The resulting file should now be opened from Eclipse.

Converting from intermediate format .simulink to EAST-ADL

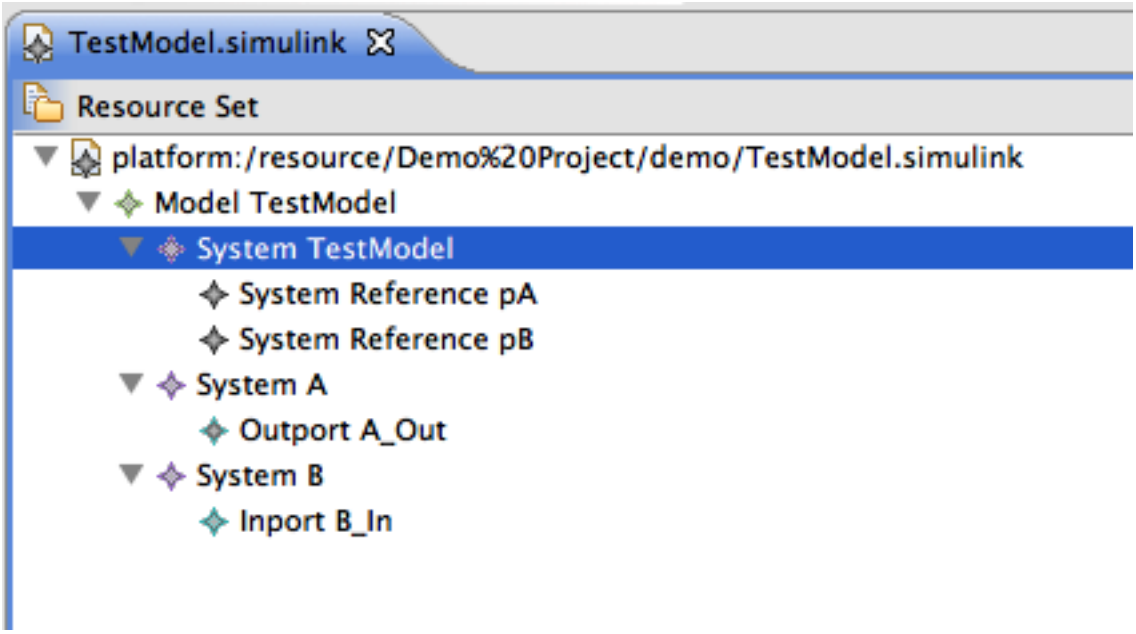
How to convert an intermediate file (with extension .simulink) to a UML file compliant to EAST-ADL.

Refresh the workspace



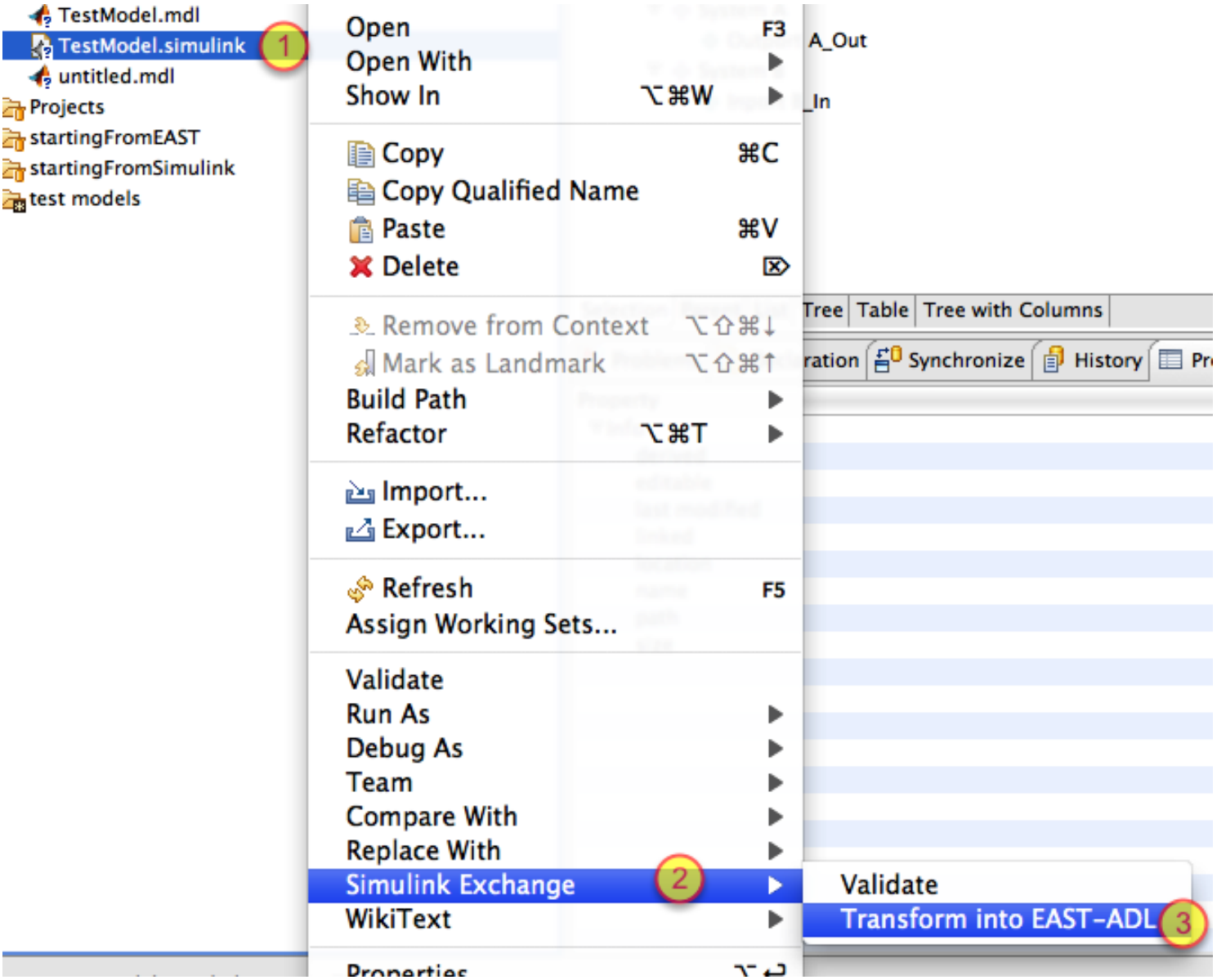
Make sure to refresh the project folder to see the latest version of the intermediate file just created by MATLAB.

Explore the model



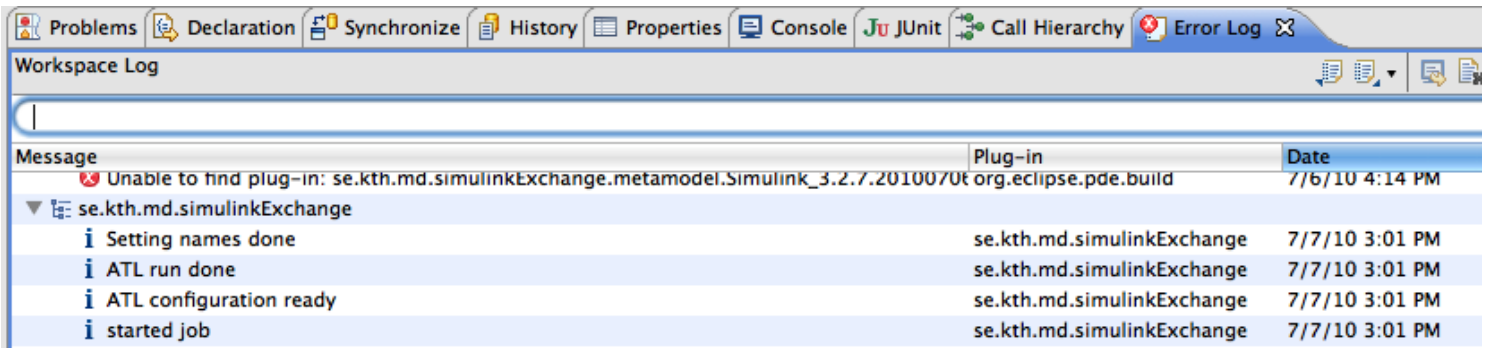
You can explore the model before attempting the conversion to check that it is what you expected.

Attempt a conversion



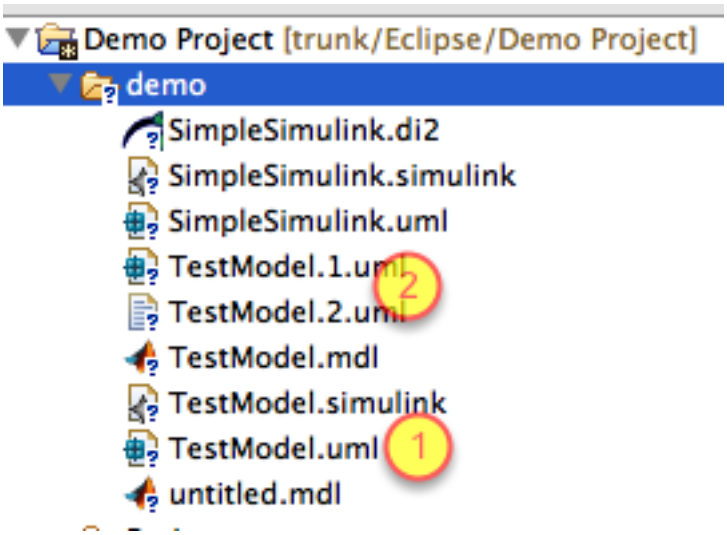
Doing a right-click on the .simulink file will give you a pop-up menu with a conversion item.

Status of the conversion



The Error log for a succesfull conversion will look like the picture above.

Resulting files

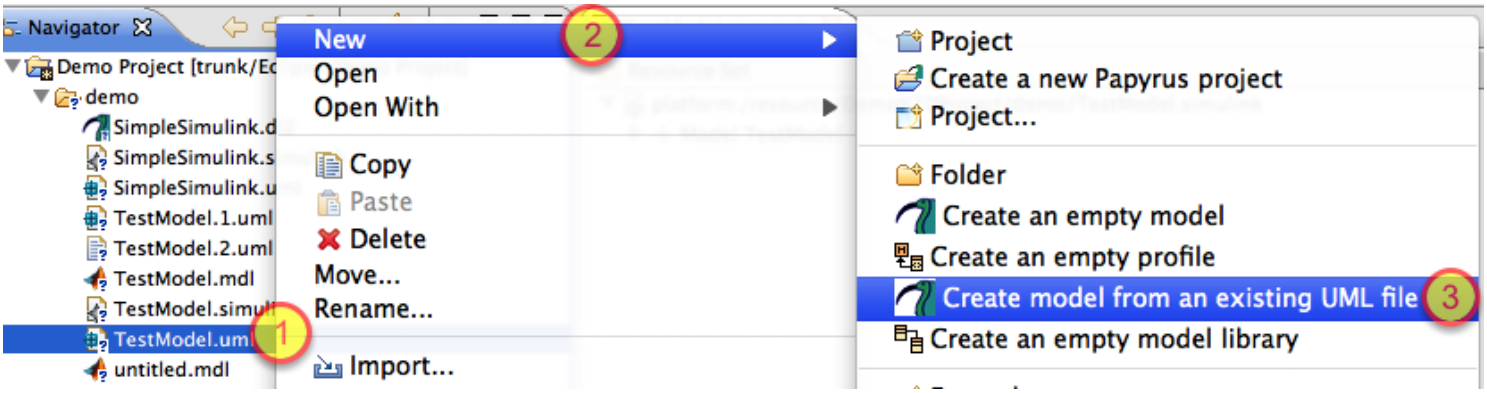


Three files are created, with the same names, but slightly different extensions. Remeber to refresh the project.

1) This is the file you will use.

2) These files can be safely deleted. (File number 1 is a mix of these files.)

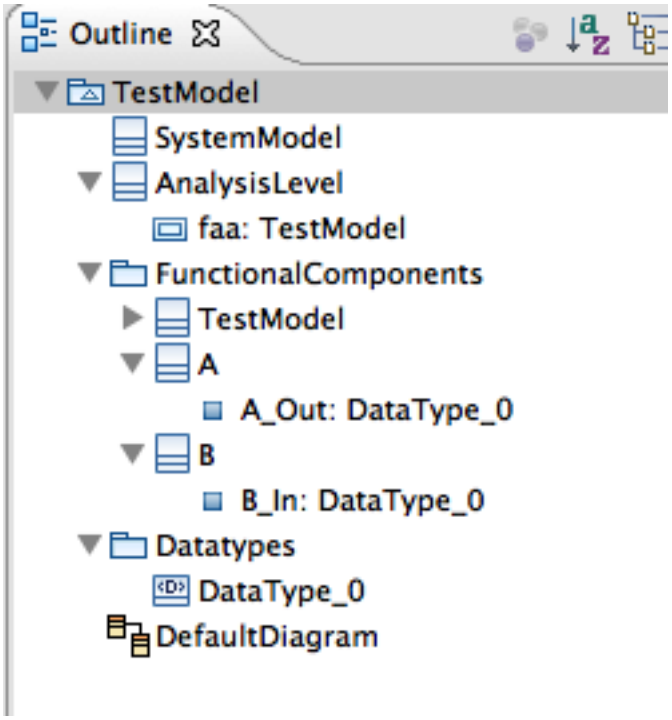
Create and EAST-ADL file from a UML file.



From the Papyrus perspective, right-click on the UML file and create a new model from it.

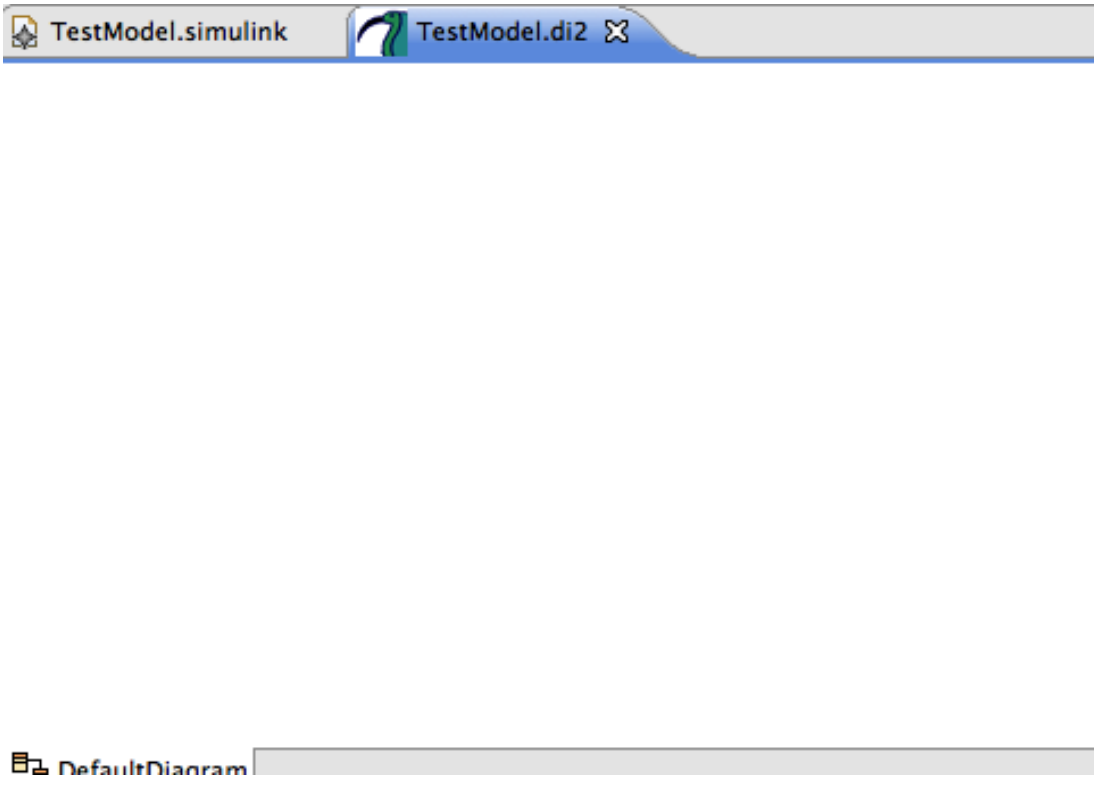
A wizard will appear and you can safely accpet the defaults.

Structure exploration



The ouline view shows the structure of the components within the file.

No graphical information



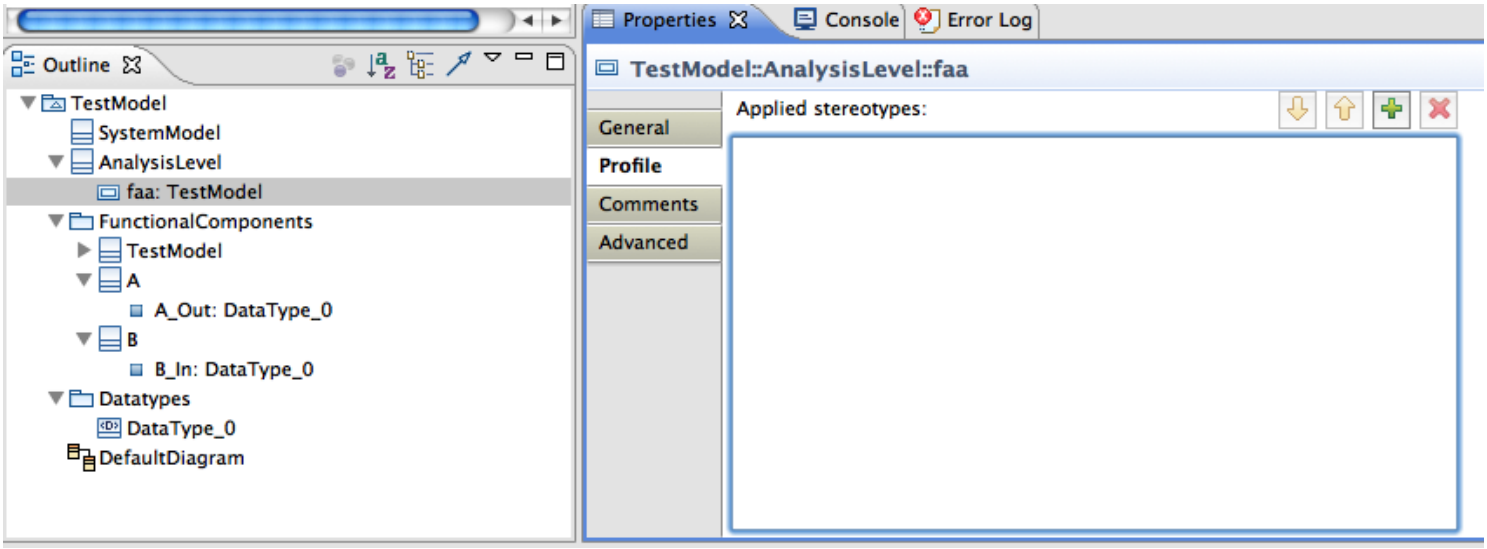
The graphical information is stored in the d12 file, which is ignored by our conversion process. Thus there is no graphical information available.

Known issues

Missing stereotypes

UML stereotypes are not applied to UML classes, thus round trip convertion is not possible yet.

Applied stereotypes



As this picture shows, this class has no stereotypes applied. There should be a FunctionType stereotype applied to it.