DAT240 / DIT596
Assignment 4
Model Transformation

Model-Driven Engineering

Group 1

Supriya Supriya (supe@student.chalmers.se)
Himanshu Chuphal (guschuhi@student.gu.se)
Kristiyan Dimitrov (gusdimkr@student.gu.se)

List of Figures

1. Introduction

Model-to-Text transformation transforms any model conforming to the meta-model (or the meta-model from assignment 1) into a DOT representation. DOT is a plain text language for describing graphs. These descriptions are used to generate graphs using tools like Graphviz.

Model-to-Model transformation, written in QVT-O, transforms a hierarchical model, which corresponds to our Manufacturing System metamodel, into a flattened Manufacturing System model. The result is a model corresponding to the same metamodel as the input model.

A. Model-To-Text Transformation

- ### Code Generation With XTend

  For the Model to Text transformation we used a form of xtext/xtend templating. Similar to String Templates everything inside ‘’’ ‘’’ will be treated as a string. The only exceptions are simple operations like FOR loops and IF statements. Additionally, everything inside << >> similar to String templates will be replaced with the according method output that’s called or operation performed. In general we used FOR loops to iterate through the ManufacturingSystem and ManufacturingSystemElement. Once looping to grab the ManufacturingSystemElements, we loop within the first for loop to grab the transition of the element and build it in the following format:
  <<*element name*>> -> <<*name of the containing transition element>>
  The same structure was used to link the responsible persons, workpieces, etc. Since everything is within the the file generator of xtend and the Template option that we used writes every iteration on a new line within the file we end up with nodes and edges which correspond to the DOT format, representing the model.

  MyDslGenerator.xtend, *(* editing the file in the xtext project->src->generator->yMyDslGenerator.xtend)

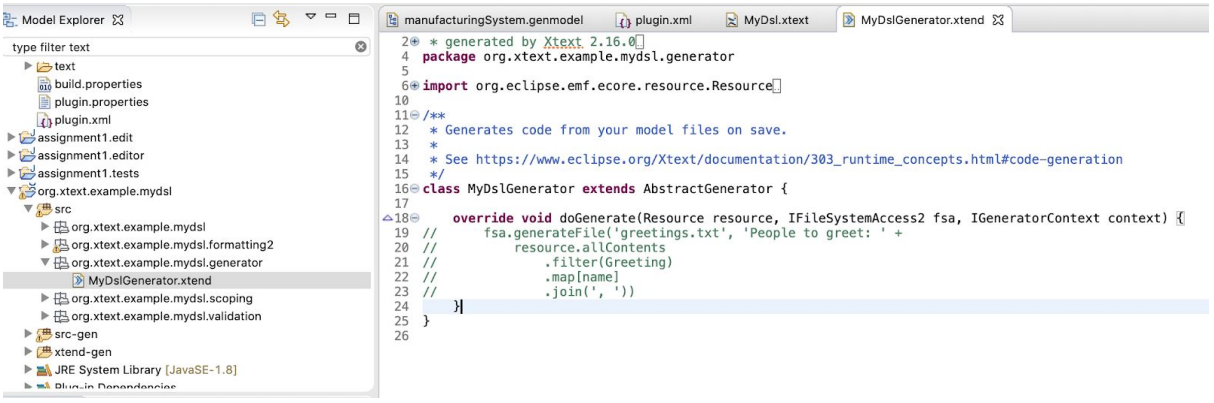


Image 1: XTend Generator file

- Generated code (ManuSysStg.gv)

  This is taken as an input for graphical representations from the generated files to online tool Webgraphviz (http://www.webgraphviz.com/)

Eclipse Running Application

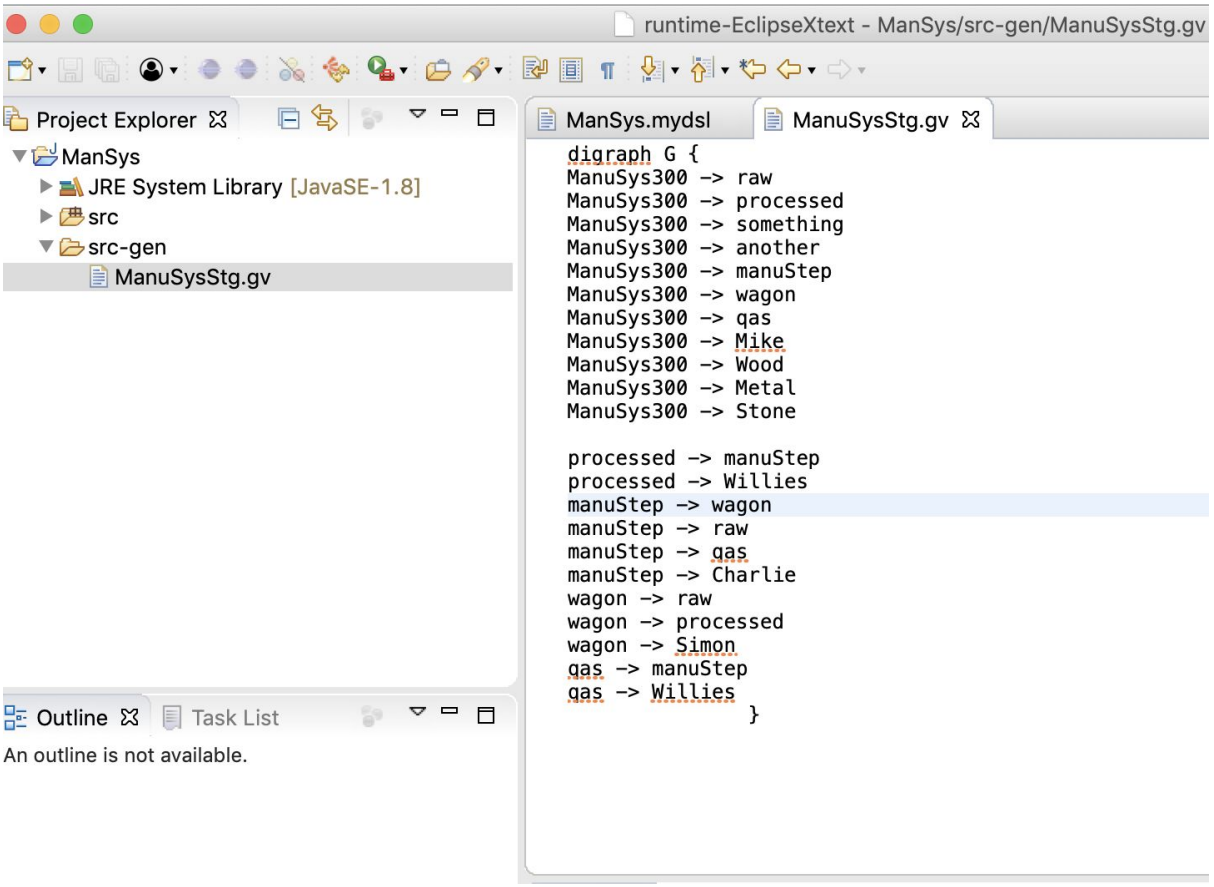
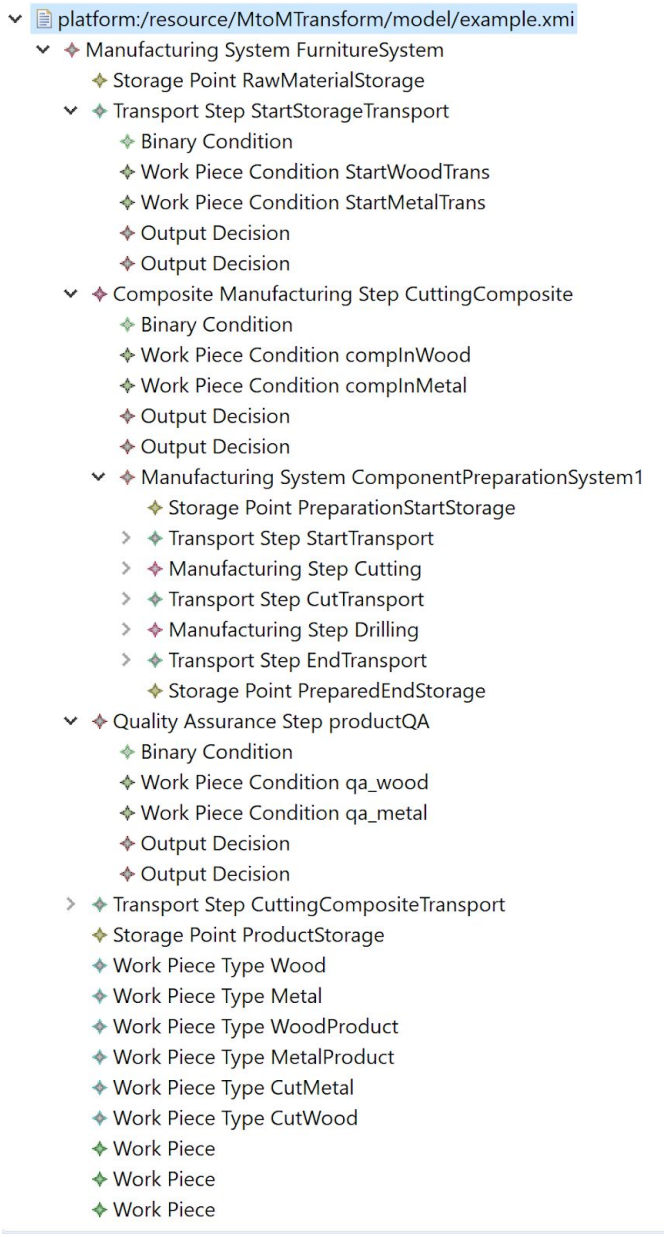


Image 2: Running Application

B. Flattening hierarchical models

● Model Transformation using QVTo

Here in Model to Model Transformation, we use QVT operational. Since the model in this context is transformed into a model of the same type, we follow an endogenous method of transformation, mapping the classes to the same name as in the original model. The metamodel from assignment 1 supports hierarchy. This means that ManufacturingSystems can contain other ManufacturingSystems via CompositeManufacturingSteps. Removing CompositeManufacturingSteps from the ***example.xmi*** (from Assignment 1), the result is a model corresponding to the same metamodel as the input model but with no ambiguous hierarchy by containing manufacturing system.



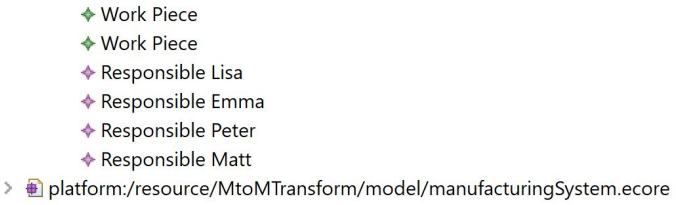


Image 3: *example.xmi file*

The image3 shows the model before transformation.

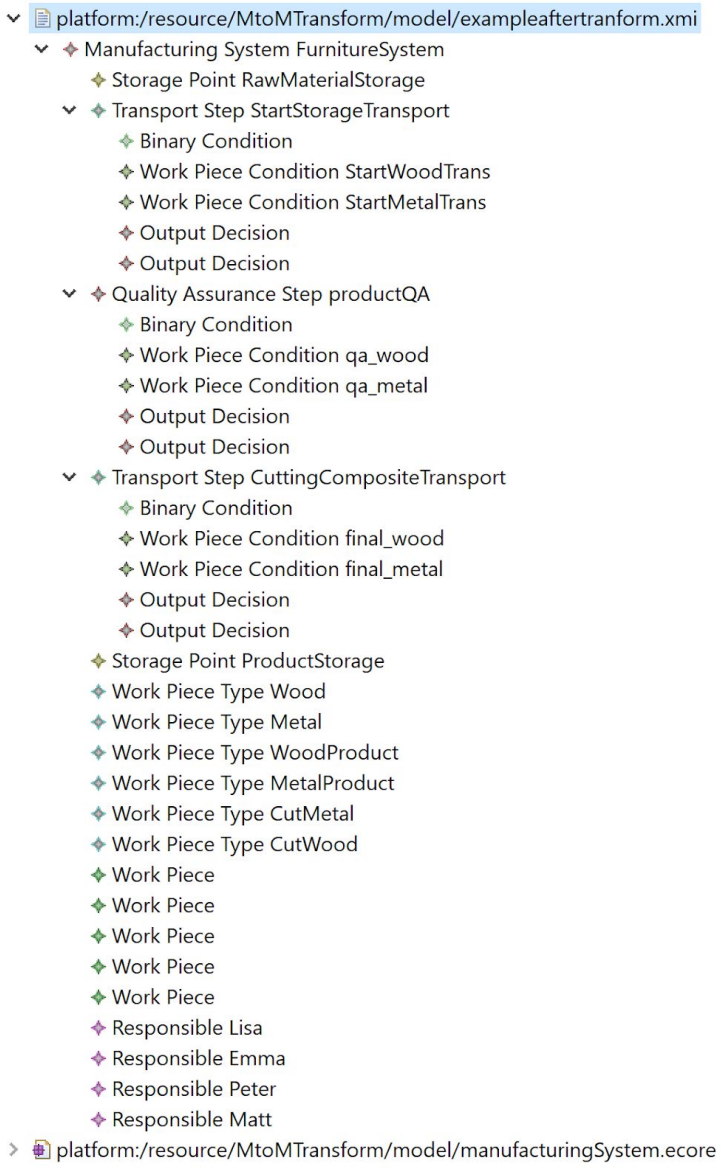


Image 4: exampleaftertransform.xmi file

Image4 shows the structure of the flattened transformed model. The ManufacturingSystem class is made to include all the steps excluding the CompositeManufacturing Step.

References

1. http://en.wikipedia.org/wiki/DOT_graph_description_language (Links to an external site.)Links to an external site.
2. http://www.graphviz.org/
3. http://www.webgraphviz.com/
4. https://bitbucket.org/itu-square/mdsebook/wiki/Home
5. Eclipse XTend https://www.eclipse.org/xtend/documentation/203_xtend_expressions.html#lambdas