Oblig 2
Creating a Domain Specific Language using the Eclipse Modelling Framework

Tobias Birmili, Florian Hagenauer, Kacper Surdy

07.05.2012

Contents

1 Introduction

This article is part of the second assignment for the course "Modelbased System Development". It discusses how the three parts of the assignment have been solved. The basic task was to build a model for customer journeys, make it possible to extract statistics out of customer journeys and finally the option to choose between three additional tasks. For the third part it was possible to choose between a user-friendly editor, a graphical exporter or adding support for multiple journeys. For this article the graphical export option was chosen.

Section 2 describes how the model was built and discusses the different components. The second section 3 depicts which statistics were generated. Section 4 compares the initial graph idea to the final graphs produced by the exporter. Finally, section 5 shows some details about the implementation and how we used the EMF tools.

2 Creating a DSL metamodel with EMF

The first part of the work was to create a metamodel using the Eclipse Modelling Framework (EMF) for a customer journey. A customer journey describes a collection of touchpoints a customer has with a product. Such a journey usually consists of a reference which is the ideal way. After finalizing this journey some customer are asked to take the journey and rate the different touchpoints and mybe add new ones. Every touchpoint should be rated and, if possible, commented. For this metamodel, we had some example journeys consisting of a reference journey and four different customer journeys which differ in some ways from the reference.

2.1 Domain Specific Language Metamodel

The model itself consists of three main components: The first one is the `JourneySet`. This component includes the reference journey and a set of examples. A reference journey has to be set to allow comparison between the example journeys and the reference. This component also has a method which allows the comparison of the journeys to the reference journey.

The second component is the `Journey` which depicts the Customer journey itself. This journey consists of an ID, a more descriptive name, a date, the status and an optional comment. If the ID of a journey is `reference` then the journey is considered to be the reference journey. The model also contains some methods which allow to compare the journeys and generate statistics for a special journey or generate the graphical representation. A more detailed description of this can be found in the Sections 3 and 4.

The last main component is called `Touchpoint`. It describes the users interaction and experience during a journey. Beside an ID, a name, a date and a comment it also has a type which describes if it is a generic or and ad-hoc touchpoint, and evaluation where the customer votes how good the experience at the touchpoint was, a initiator for the touchpoint and finally a channel through which this touchpoint was experienced. The variables channel, component, type, initiator and evaluation were done as enums. In that way they can be extended quite easily by adding a new type to the enum and generate the new code.

Another part, which is important for 3, is the component `JourneyDiff` which stores the differences between the reference journey and the different example journeys.

3 Making a transformation to extract statistics

The second task was to extend the model to generate statistics and print them in a convenient form. To represent this generation in the model some methods were added to the `JourneySet` and to the `Journey` components. The `JourneySet` has the operation `getComparedToExpected` to compare the example journeys from the set to the reference

journey. The single journeys got methods which allow to extract single statistics for the journey. These statistics are:

Rating Statistic These statistics scan the different rating types and allow to see how often each rating was given and the percentage of the total ratings.

Channel Statistics These statistics cover the number of channels and percentage.

Initiator Statistics How often an initiator invoked touchpoints can be found in this statistic.

Compare To Reference This statistic shows how much the example journey differ from the given reference journey.

One part of the assignment was to print the statistics in a convenient and useful way. We choose the generate a HTML report which contains all data in a standalone file for easy sharing. The report contains all the journeys with their touchpoints, the generated graphs and statistics. The list of touchpoints is linked with the graphical representation by clientside scripting.

A reasonably modern browser with SVG support is necessary to display the HTML report.

4 Graphical Export

The last part of the assignment was to choose between different extensions for the metamodel. In this article the graphical export feature was chosen. The task was to generate a graphical representation of the customer journeys to make them more easy to understand. The choice in this article fell to Graphviz [1] a visualization tool that allows to print graphs in a convenient way. The metamodel was extended with the operation `getGraphviz` inside of `JourneySet` and `Journey`. The basic idea can be seen in Figure 1.

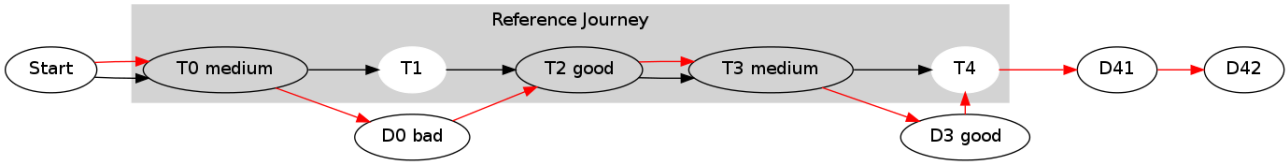

Figure 1: The basic idea for the visualization

The basic idea is to have the reference journey, here depicted in the grey box and the black arrows, and the example journey, which has red arrows in the example. It should be easy to spot the difference between the reference and the example journey. Also the ratings are directly included into the graph.

[1] `http://graphviz.org/`

Figure 2 shows the same customer journey as Figure 1 but this time the generated version of the program. In the final version instead of written ratings colors are used (blue for medium, red for bad and green for good). This makes it easier to see the general satisfaction. One rectangle always shows the reference journey and another the new (mostly ad-hoc) touchpoints added by the example. Also, the orange arrows make it easy to follow the example journey.

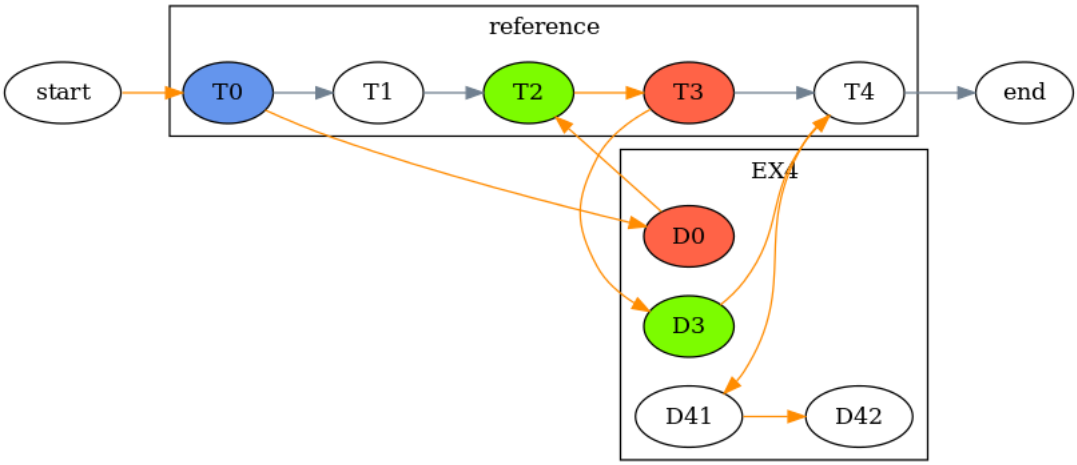

Figure 2: The realization generated by the Visualizer

5 Implementation Details

In this section some details about the implementation and the general project can be found. Figure 3 shows the different projects for the assignment.

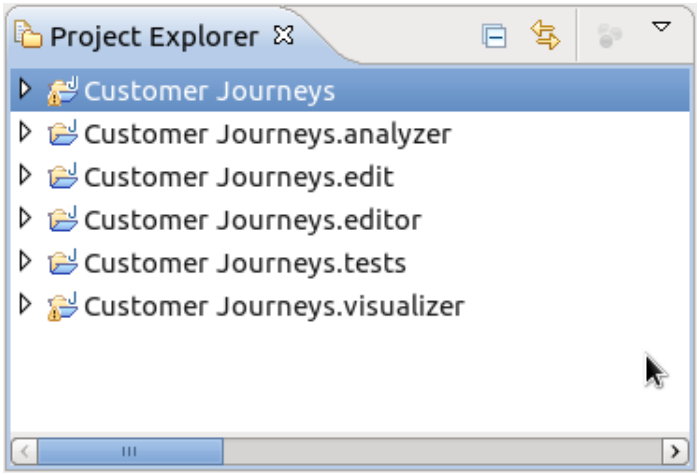

Figure 3: The different Eclipse projects of the assignment

The project Customer Journey is the main project which contains the model and the generated code. The three projects .edit .editor and .tests are generated by the model. The .editor was used to define the example and the reference journeys. The project .analyzer contains the source code to load a customer journey set and return the analysis. Finally, the .visualizer project is responsible for visualizing the customer journeys as described in Section 4.

For version control a github repository was used[2].

[2] `https://github.com/floxyz/INF5120`