Artifacts for Paper #35 of the Software Language Engineering Conference 2017

A Requirements Engineering Approach for Usability-Driven DSL Development

Authors: Same as paper

Abstract: This document presents the artifacts used to evaluate the requirements engineering approach described in paper #35 of the SLE 20017 conference. The artifacts mainly consist of requirements, use case and architecture models of the presented Gyro robot DSL example and the metamodel of the introduced DSSL language. The models can be edited via the Eclipse-based RDALTE tool. The document first explains how to install the tool. Then for every metamodel and model presented in the paper, the document describe how the elements can be viewed and edited using the tool.

1 Overview of Archive

The archive contains this document, the submitted paper and the RDAL language specification document. In order to simplify the evaluation of the artifacts, we have prepared Eclipse packages for the Windows and Linux platforms into which all the required tools have already been installed. Example projects for the models of the paper are provided and can easily be installed in the tool as described in the installation instructions section below.

2 Installation

First install Eclipse with the proper tools and then install the example projects following the instructions below.

2.1 Installing Eclipse

2.1.1 Using the Prepared Packages

Pre-installed Eclipse packages for the Windows and Linux platforms can be downloaded at the following links:

Windows: https://partage.mines-telecom.fr/index.php/s/UTr4TPqkNjv3Wsj/download

Linux: https://partage.mines-telecom.fr/index.php/s/yjiZZZC1nF6Zttw/download

Mac: https://partage.mines-telecom.fr/index.php/s/sC0w2jTGb5BCIyY/download

Unzip the package and run the Eclipse executable.

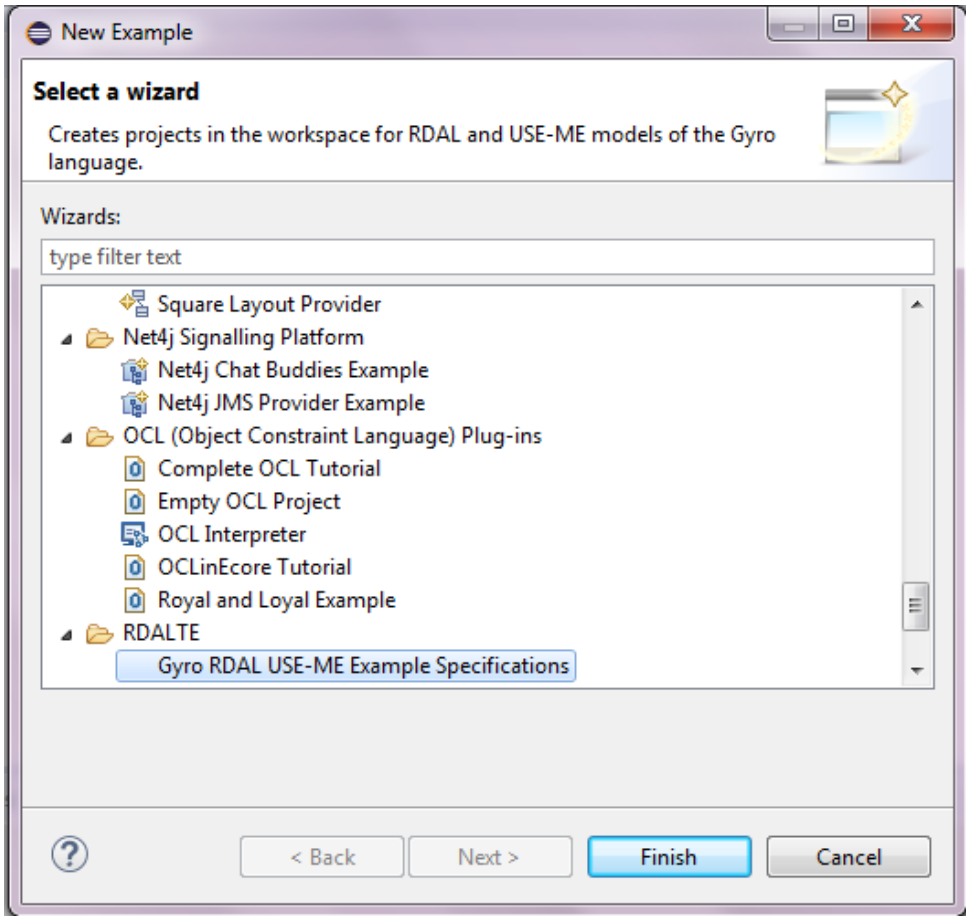
2.1.2 From the Eclipse Modeling Tools Package

Follow the instructions at https://mem4csd.telecom-paristech.fr/blog/index.php/rdal/ in order to install the tools from a fresh Eclipse.

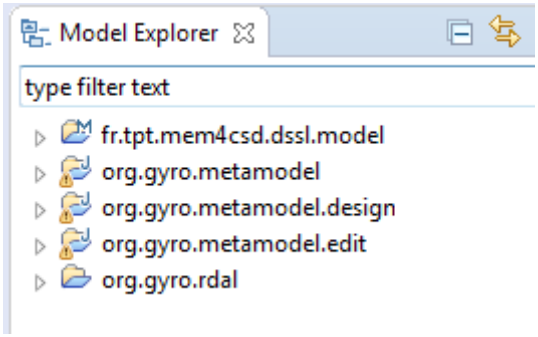
2.2 Installing the Example Projects

Switch to the *Modeling* perspective by opening menu *Window>>Perspective>>Open Perspective>>Other*. Select the *Modeling* perspective from the displayed dialog box.

Next, install the example projects by opening menu *File>>New>>Example*. Select the *Gyro RDAL USE-ME Example Specifications* as shown in the figure below and click *Finish*.



This will install in your workspace the following 5 projects:



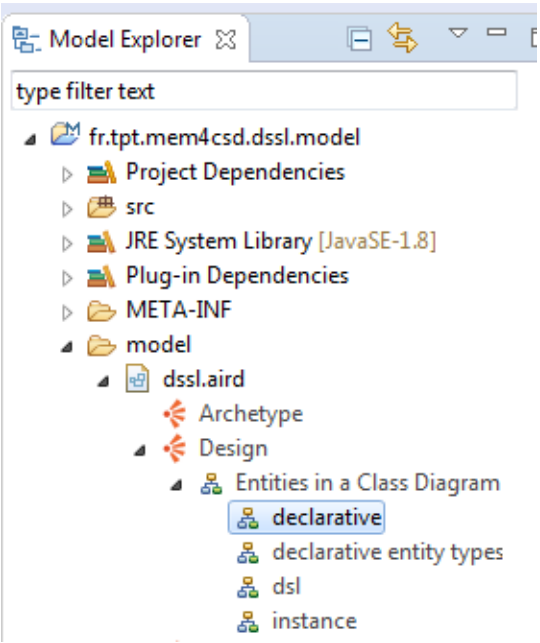
3 Editing and Viewing the Models

The following describes how to view the models presented in the paper. The section or figure number of the paper corresponding to the presented artifacts is written in parenthesis. Editing the models is performed with the provided EMF tree editors for the languages. Double-click the file in the model explorer view to open the file. Select the element of interest in the editor and view its properties via the standard Eclipse *Properties* view. If the properties view is not visible, right click any element in the editor and click *Show Properties View* in the displayed contextual menu to open the view.

Note that the presented tool is an initial prototype that is still being developed. For instance, the concrete syntax for the DSSL and RDAL languages presented in the paper are not yet implemented in the tool. Therefore, the models can only be edited via the EMF tree editors. In addition, the presented models are not complete and mostly contain the elements presented in the paper to illustrate the approach. For example the Ecore metamodel for the Gyro language and its Sirius concrete syntax model are issued from an initial implementation of the language and therefore differ from the real Gyro language.

3.1 DSSL Language (Section 4)

The class diagrams of the DSSL language presented in section 4 of the paper can be viewed by unfolding project *fr.tpt.mem4csd.dssl.model* as shown in the figure below:



Each part of the metamodel presented in the figures of section 4 can be opened by double-clicking the corresponding diagram under the *dssl.aird* file as shown in the figure above for the declarative part.

3.2 Models for the Gyro Language (Section 5.1)

The following presents the models corresponding to the figures of section 5.1 of the paper.

3.2.1 System Overview Diagram (Figure 6)

The DSSL entity types of the system overview diagram can be seen in the *gyro.dssl* file of the *org.gyro.rdal* project, under the root *Specification* element. The RDAL system overview element located under the root RDAL *specification* element can be viewed from the properties view by selecting the element from the *gyro.rdal* file. Note the reference to the DSSL Gyro entity type element from the *System To Be* property.

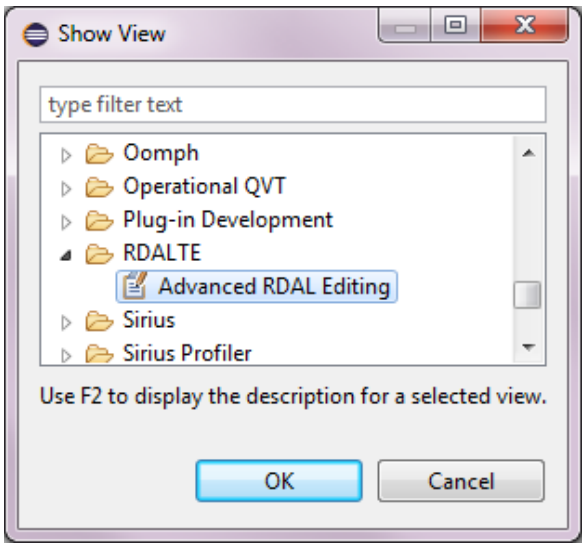
3.2.2 Normal Use Context Diagram (Figure 7)

The DSSL *Normal Use* context specification can be seen in the *gyro.dssl* model. Unfold the element to view the contained entity instances of the types of the system overview and their connections. On

the RDAL model, see the *Normal Use* context element contained in the system overview element and traced to the Gyro DSL entity instance of the DSSL context via the *System To Be* property.

3.2.3 System Goals (Figure 8)

The goals can be seen under the *Gyro System Goals* goals package of the *gyro.rdal* model. RDALTE provides an additional view to edit some elements of a RDAL model. In order to show this view, open menu *Window>>Show View>>Other* and select the *Advanced RDAL Editing* view as shown in the figure below.



The view provides a first tab allowing selecting the element of the DSSL model to which the goal selected in the RDAL file is allocated. The second tab allows selecting the language to be used for expressing the goal (including natural language for high level goals) and to edit and evaluate the defined expression when a formal language is used. Unfold the goal elements in the RDAL model to see their rationales and stakeholders. Also see property *Use Cases* of goal elements for the trace to use cases that achieve the goal (e.g. *G1*).

3.2.4 Use Cases (Figure 9 and 10)

Open the *gyro.jucm* file under the *org.gyro.rdal* project to view the use cases for the Gyro DSL. Double-click the diamonds in the first diagram to navigate to the called sub use cases (e.g. *Create Root*). Note that not all sub use cases have been modeled in this preliminary version. Also displaying a use case diagram automatically switches the perspective to that of the jUCMNav tool so you need to switch back to the *Modeling* perspective after.

3.2.5 Environmental Assumptions (Figure 11)

The environmental assumptions are located under the *Gyro Functional Requirements* requirements package of file *gyro.rdal*. Use the RDAL editing view to see the allocation of assumption to the DSSL entity types and the OCL expressions of the assumptions. Press the *OCL* button to evaluate the OCL constraints. Note the rationale contained by the assumptions and stakeholder association.

3.2.6 Functional Requirements for the Gyro Abstract Syntax (Figure 10)

The *Gyro* DSL entity type is located in the DSSL file and contains an *Ecore abstract syntax* element that itself refers to the actual Ecore package of the Gyro metamodel. This metamodel is located

under the *model* folder of project *org.gyro.metamodel*. On the RDAL model, requirement *R1* is assigned to the *Gyro* DSL entity type and expressed in natural language. *R1* is refined into *R1.1* and *R1.2* through the *requirement refinement* element *R1*. *R1.2* is further decomposed into leaf requirements *R1.2.1* and *R1.2.2*, which are both assigned to the gyro Ecore package and expressed in OCL. Such requirement can be automatically verified by clicking the OCL button of the constraint view. The OCL expressions make use of libraries that can be seen on the *Constraints Libraries* tab. Select a row from the table to open the file. Also note the link from these leaf requirements to steps of the UCM use case diagram through property *Function Used In* in the Eclipse properties view.

3.2.7 Functional Requirements for the Gyro Graphical Concrete Syntax (Figure 12)

The *Gyro* DSL entity type represented in the DSSL model contains a *Sirius Concrete Syntax* element that itself refers to a *Sirius Specification* defining the Gyro graphical syntax. The Sirius model is located under the *description* folder of project *org.gyro.metamodel.design*. Note that the Sirius diagram model is not complete and only the elements required for the provided example requirements are defined.

On the RDAL model, requirement *R1.1* is decomposed into the leaf requirements *R1.1.1.1* and *R1.1.1.2*, which are both assigned to the Sirius *Default* diagram layer providing the visual elements for concepts of the Gyro metamodel. The requirements expressed in OCL can be automatically verified by evaluating the constraints against the assigned design element.

3.3 Models for the USE-ME Language (Section 5.2)

The following presents the models corresponding to the figures of section 5.2 of the paper for the USE-ME language.

3.3.1 Integration between USEME and RDAL-DSSL (Figure 13)

The RDAL and DSSL parts of the figure can be viewed by opening their model files presented above. The USE-ME model can be viewed by opening the *gyro.useme* file located under project *org.gyro.rdal*.

3.3.2 USE-ME Models (Figure 14)

The Gyro goal diagram of the USE-ME language of figure 14 as well as all other USE-ME diagrams can be viewed by opening the diagram in file *representations.aird* of project *org.gyro.rdal*.

Goal modeling diagram is located under the *goalModelling* folder. Detailed explanations on implementation of the Gyro usability model can be found in:

Ankica Barišić, Vasco Amaral, Miguel Goulão, ***Usability Driven DSL development with USE-ME****, Computer Languages, Systems & Structures, 2017, ISSN 1477-8424, http://dx.doi.org/10.1016/j.cl.2017.06.005.*

4 Further Information

4.1 RDAL

See https://mem4csd.telecom-paristech.fr/blog/index.php/rdal/

4.2 USE-ME

See https://github.com/akki55/useme/wiki

Video tutorial https://youtu.be/RjIGFex-zQM