



Grant ANR-12-INSE-0011

---

# **ANR INS GEMOC**

## **D1.2.1 - DSML behavioral semantics definition tools**

### **Task 1.2.1**

**Version 0.2**

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

## DOCUMENT CONTROL

	–: 2013/11/22	A: 2013/11/28	B:	C:	D:
Written by	Didier Vojtisek	Didier Vojtisek			
Signature					
Approved by					
Signature					

Revision index	Modifications
–	version 0.1
A	version 0.2 - addition of workflow chapter
B	
C	
D	

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

## Authors

Author	Partner	Role
Didier Vojtisek	INRIA	Lead author
Benoit Combemale	INRIA	Contributor
Mélanie Bats	OBEO	Contributor
Florent Latombe	INPT	Contributor

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

## Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Purpose . . . . .	5
1.2	Perimeter . . . . .	5
1.3	Definitions, Acronyms and Abbreviations . . . . .	5
1.4	Summary . . . . .	6
<b>2</b>	<b>Architecture of the DSML behavioral semantics definition tools</b>	<b>8</b>
2.1	Implementation of an Executable DSML (xDSML) . . . . .	8
2.2	xDSML development method . . . . .	8
2.3	xDSML development tool . . . . .	9
2.3.1	xDSML Project . . . . .	9
2.3.2	Internal structure of the GEMOC Language Workbench . . . . .	10
2.3.3	xDSML Metamodel . . . . .	10
<b>3</b>	<b>Gemoc Language Workbench: xDSML development tool workflow</b>	<b>11</b>
3.1	xDSML definition . . . . .	13
3.1.1	New Gemoc Language project Command . . . . .	13
3.2	Domain Model definition (AS) . . . . .	13
3.2.1	New EMF Project Command . . . . .	13
3.2.2	Select existing EMF Project Command . . . . .	15
3.3	Modelers definition (CS) . . . . .	15
3.3.1	New Tree editor Command . . . . .	15
3.3.2	Select existing Tree editor Command . . . . .	15
3.3.3	New Xtext editor Command . . . . .	17
3.3.4	Select existing Xtext editor Command . . . . .	17
3.3.5	New Sirius editor Command . . . . .	17
3.3.6	Select existing Sirius editor Command . . . . .	18
3.4	RunTime Data and Domain-Specific Actions . . . . .	18
3.4.1	New Kermeta 2 project Command . . . . .	18
3.4.2	Select existing Kermeta 2 project Command . . . . .	18
3.4.3	New Kermeta 3 project Command . . . . .	20
3.4.4	Select existing Kermeta 3 project Command . . . . .	20
3.5	Model of Computation (MoC) definition . . . . .	20
3.5.1	New CCSL Moc project Command . . . . .	20
3.6	Domain-Specific Events specification file definition . . . . .	21
3.6.1	New ECL file Command . . . . .	21
3.6.2	New Modelh'x project Command . . . . .	22
3.6.3	Select existing ECL file Command . . . . .	22
3.7	Animators definition . . . . .	23
<b>4</b>	<b>Current implementation</b>	<b>24</b>
<b>5</b>	<b>Conclusion</b>	<b>25</b>
<b>A</b>	<b>Getting started with the Language Workbench</b>	<b>26</b>

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

# 1. Introduction

## 1.1 Purpose

This document presents the tools supporting the methodology for defining an Executable Domain-Specific Modeling Language (xDSML). The developed tools orchestrate a set of features proposed by the GEMOC language workbench to help the language designer in the various activities leading to the definition of an xDSML.

We describe the various steps of the process and the tools that support it.

The tools and the wizards supporting the process are integrated and deployed into the GEMOC Studio.

## 1.2 Perimeter

This document describes the version v0 of the software deliverable D1.2.1 (*DSML behavioral semantics definition tools*). It presents the current state of the tools provided in the Language Workbench of the GEMOC studio. These tools ease the application of the methodology defined in Task 1.1.

## 1.3 Definitions, Acronyms and Abbreviations

- **Model:** model which contributes to the content of a View
- **Language Workbench:** a language workbench offers the facilities for designing and implementing modeling languages.
- **Language Designer:** a language designer is the user of the language workbench.
- **Modeling Workbench:** a modeling workbench offers all the required facilities for editing and animating domain specific models according to a given modeling language.
- **Domain Engineer:** user of the modeling workbench.
- **GEMOC Studio:** Eclipse-based studio integrating both a language workbench and the corresponding modeling workbenches.
- **DSML:** Domain Specific Modeling Language.
- **xDSML:** Executable Domain Specific Modeling Language.
- **AS:** Abstract Syntax.
- **MOC:** Model of Computation.
- **RTD:** RunTime Data.
- **DSA:** Domain-Specific Action.
- **MSA:** Model-Specific Action, an instance of a Domain-Specific Action. While a DSA is specific to a language (to its Abstract Syntax), an MSA is specific to a model conforming to a language.
- **DSE:** Domain-Specific Event.
- **MSE:** Model-Specific Event, an instance of a Domain-Specific Event. While a DSE is specific to a language, an MSE is specific to a model conforming to a language.

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

- **GUI:** Graphical User Interface.
- **Eclipse Plugin:** an eclipse plugin is a Java project with associated metadata that can be bundled and deployed as a contribution to an Eclipse-based IDE.
- **API:** Application Programming Interface
- **CCSL:** Clock-Constraint Specification Language.
- **TESL:** Tagged Events Specification Language.
- **Static semantics:** Constraints on a model that cannot be expressed in the metamodel. For example, static semantics can be expressed as OCL invariants.
- **Execution semantics:** Define when and how elements of a language will produce a model behavior.
- **Dynamic semantics:** see *Execution semantics*.
- **Behavioral semantics:** see *Execution semantics*.

## 1.4 Summary

This document describes the current state of the tools provided in the Language Workbench of the GEMOC studio as part of the D1.2.1 deliverable in its version v0.

The GEMOC language workbench is intended to domain experts who wish to design a new Executable Domain-Specific Modeling Language (xDSML).

Building an xDSML consists in defining the various parts of a language (abstract syntax, concrete syntaxes and behavioral semantics), aggregate together thanks to a model of their aggregation (xDSML model) as part of a dedicated project (xDSML project). The combination of these parts provides a concrete tooling supporting the execution of the conforming models. It will also provide a convenient way to deploy the resulting language via a modeling workbench. In this case, the domain designers will be able to create and execute models conforming to this language.

The parts required to build an xDSML are: a domain model (Abstract Syntax or AS), the Domain-Specific Actions (DSAs) including both the execution function and data, a Model of Computation (MoC), the Domain-Specific Events (DSEs), one or several editors (modelers) and optionally one or several graphical animators.

The GEMOC studio offers a workflow that is supported by a dashboard and a suite of wizards for helping the domain experts while designing an xDSML.

The workflow explicitly supports various technologies for implementing the required parts of an xDSML. The concrete technologies explicitly supported by the studio for building the xDSML parts are:

- Domain Model:
  - Ecore + EMF genmodel
- DSAs:
  - Kermeta 2
  - Kermeta 3
  - direct modification of ecore and generated code
- DSEs:
  - ECL
- MoC:
  - CCSL with the Timesquare CCSL Solver.

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

- Editors:
  - EMF tree editor
  - Xtext text editor
  - Obeo designer viewpoint
- Animators (Backends):
  - Obeo Designer for Simulation

Other technologies can be used by creating them separately and including them manually. We plan to replace ECL with a new language more tailored to our needs, and we plan to support MoCs specified in TESL, a formalism used by ModHel'X.

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

## 2. Architecture of the DSML behavioral semantics definition tools

### 2.1 Implementation of an Executable DSML (xDSML)

The implementation of an xDSML consists in implementing the required components of the language so that it can support the execution of the conforming models. Figure 2.1 illustrates the overall process to define the different components, namely the abstract syntax, the concrete syntax (both for edition and animation), and the behavioral semantics (incl., the DSAs, the MoC and the DSEs).

Each of these components can be implemented using different tools if they are compatible with the Eclipse platform and EMF. For example, the editors can be tree editors (from EMF), Sirius viewpoints or Xtext editors.

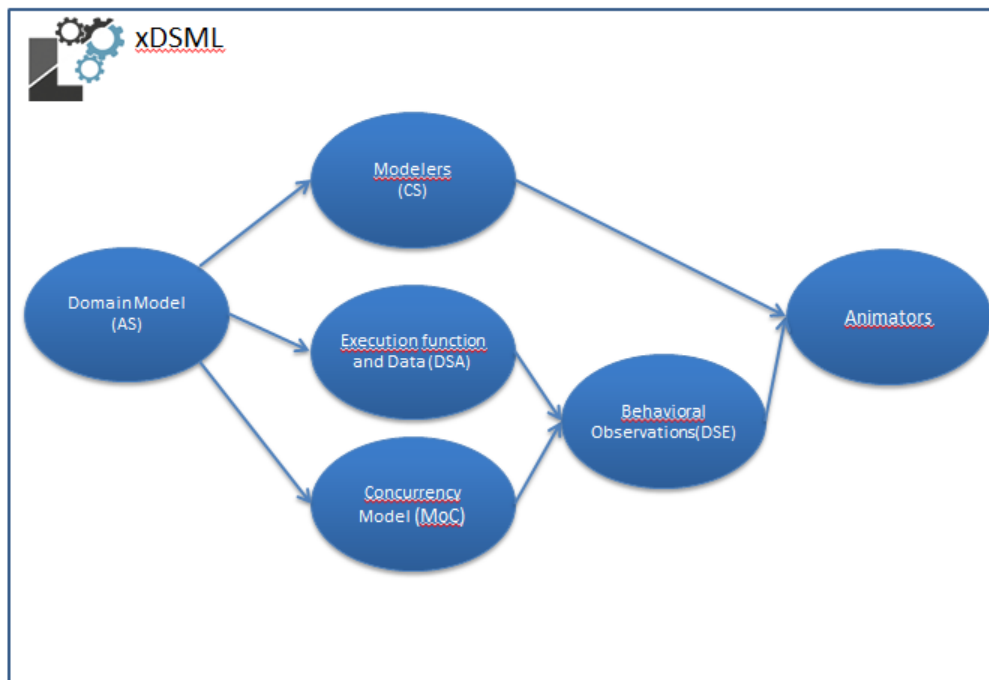


Figure 2.1: xDSML components

### 2.2 xDSML development method

It is possible to develop an xDSML by directly linking all the components together. However, in order to ease the integration and to tackle the possible variability of the technology used for implementing the different components, the GEMOC Studio offers a development method that provides guidances and drives the language designer.

The method implements a workflow corresponding to the guidelines specified in D1.1.1.



ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

## 2.3 xDSML development tool

In the GEMOC Studio, the workflow is equipped with a tooling support that offers GUI interactions for launching the concrete wizards. This tooling strongly relies on a special eclipse project: the xDSML project, that includes in a dedicated model (an xDSML model) an explicit description of the aggregation of the various xDSML components. the xDSML model contains the aggregation definition that will be used by the various tools to update the xDSML project accordingly.

### 2.3.1 xDSML Project

The role of the xDSML project is twofold:

- to support the interactions with the user while building an xDSML,
- to offer the technical services used by the Modeling Workbench tooling.

#### 2.3.1.1 Typical structure

An xDSML project contains the following elements:

**project.xdsml** This is the xDSML model containing the xDSML definition of the current language. This is actually a model conforming to the metamodel xdsml.ecore (cf. Section 2.3.3)

**plugin.xml** This file declares the available services which can be used by other components (*e.g.*, components of the GEMOC Modeling Workbench).

**xdsml-java-gen** This folder is automatically generated by the language workbench from the xDSML model. It implements the technical services.

**manifest.mf** The dependencies to the other language components are automatically added to the manifest.mf.

**Other technology-dependent artefacts** Depending on the tool used for a given xDSML component, some additional artifact might be created and included into the project (*e.g.*, DSEs written in ECL will produce a QVTo file that will be used by the solver).

#### 2.3.1.2 User interactions

The xDSML project is the support for various user interactions.

- visualization of the current implementation of the xDSML. A view allows to see what are the language components aggregated into the xDSML.
- create a new component. It allows to launch wizards able to build the new components.
- set a new component from an existing one. It allows to launch wizards able to select an existing compatible component and associate it to the xDSML implementation.
- open the language components. It allows to open the project or the file for a language component already aggregated into the xDSML.

These interactions are driven by the process described in chapter 3.

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

### 2.3.1.3 Technical services

The xDSML project offers the following technical services for the Modeling Workbench tooling (e.g., to be used by the Execution Engine).

- A *Model Loading service* which is able to load a model described in (one of) the available syntax(es) (i.e., the one implemented for the xDSML and specified in the xDSML model).
- A *DSA execution service* which is responsible for locating the methods and objects referenced by instances of DSEs in the DSAs (i.e., bytecode).
- A *Solver service* which is responsible for interpreting the solver input (resulting from the compilation of the Domain-Specific Events for a given model) and for returning a scheduling trace that can be interpreted by the Execution Engine.
- A *Feedback Policy service* which specifies how the returning values of DSAs should impact the MoC. For example, the evaluation of the condition of a decision node will impact which branches of the decision node can happen in the future of the execution.

Depending on the tool used for a given language component, some additional services might be associated to the project.

### 2.3.2 Internal structure of the GEMOC Language Workbench

All the tools required for developing a new xDSML are gathered and deployed in the GEMOC Language Workbench.

It also supports the definition of the xDSML project, as well as the interactions and automated tasks relying on the xDSML project. This workbench is based on the following main internal components:

**xDSML metamodel (xdsml.ecore)** This is the metamodel for defining xDSML models. This is an Ecore model specifying the information about a given language. This metamodel is supported by the tool ensuring the aggregation of the various language components.

**xDSML editor** This is an editor implementing the view on a given xDSML.

**Actions supporting the xDSML design process** Depending on the state of the xDSML model (i.e., language description), the GUI proposes context-dependent actions to support the user in the language definition. It launches Eclipse commands and Eclipse wizards.

**xDSML builder** This Eclipse service tracks the changes on the xDSML model. Whenever a change occurs, it automatically updates or generates the relevant artefacts in the xDSML project.

### 2.3.3 xDSML Metamodel

An xDSML model conforms to the xDSML metamodel. The metamodel is presented in Figure 2.2 and Figure 2.3. It allows to represent resources such as files or eclipse projects in the language workbench in order to aggregate them as an xDSML.

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

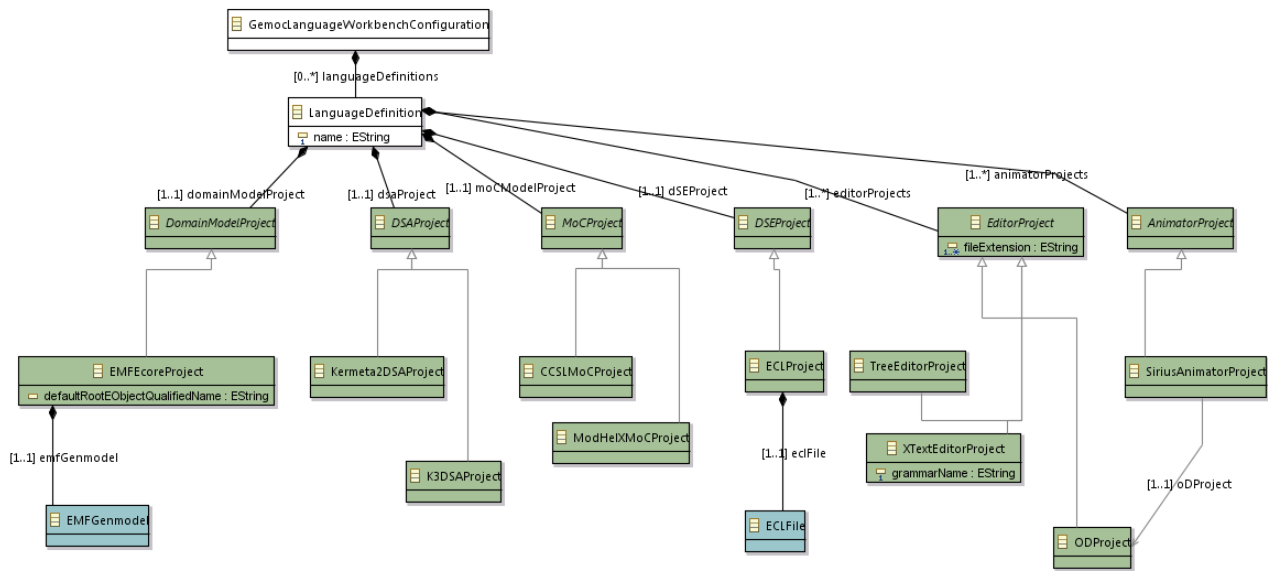


Figure 2.2: xDSML metamodel - containment view

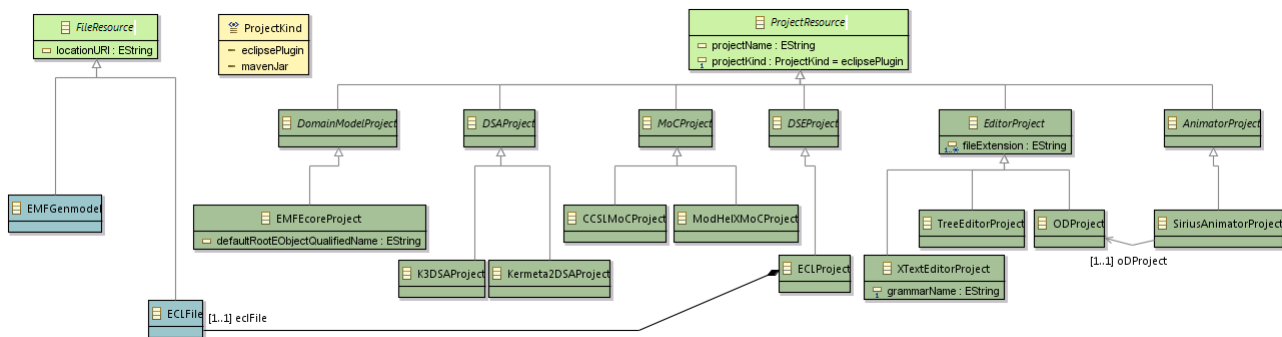


Figure 2.3: xDSML metamodel - Local resource view

### 3. Gemoc Language Workbench: xDSML development tool workflow

The GEMOC Studio offers several services for defining a xDSML. The underlying tools can be called in a predetermined sequence in order to achieve the various activities that are required to build an xDSML. A specific workflow tool has been implemented to help the user in performing the activities (or steps) of the GEMOC xDSML development method. Figure 3.1 presents the global view of the tools that implements the different activities of the Language Workbench workflow.

In the following sections, each activity will be fully detailed, including the major concrete artifacts resulting from the commands.

#### Caption:

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

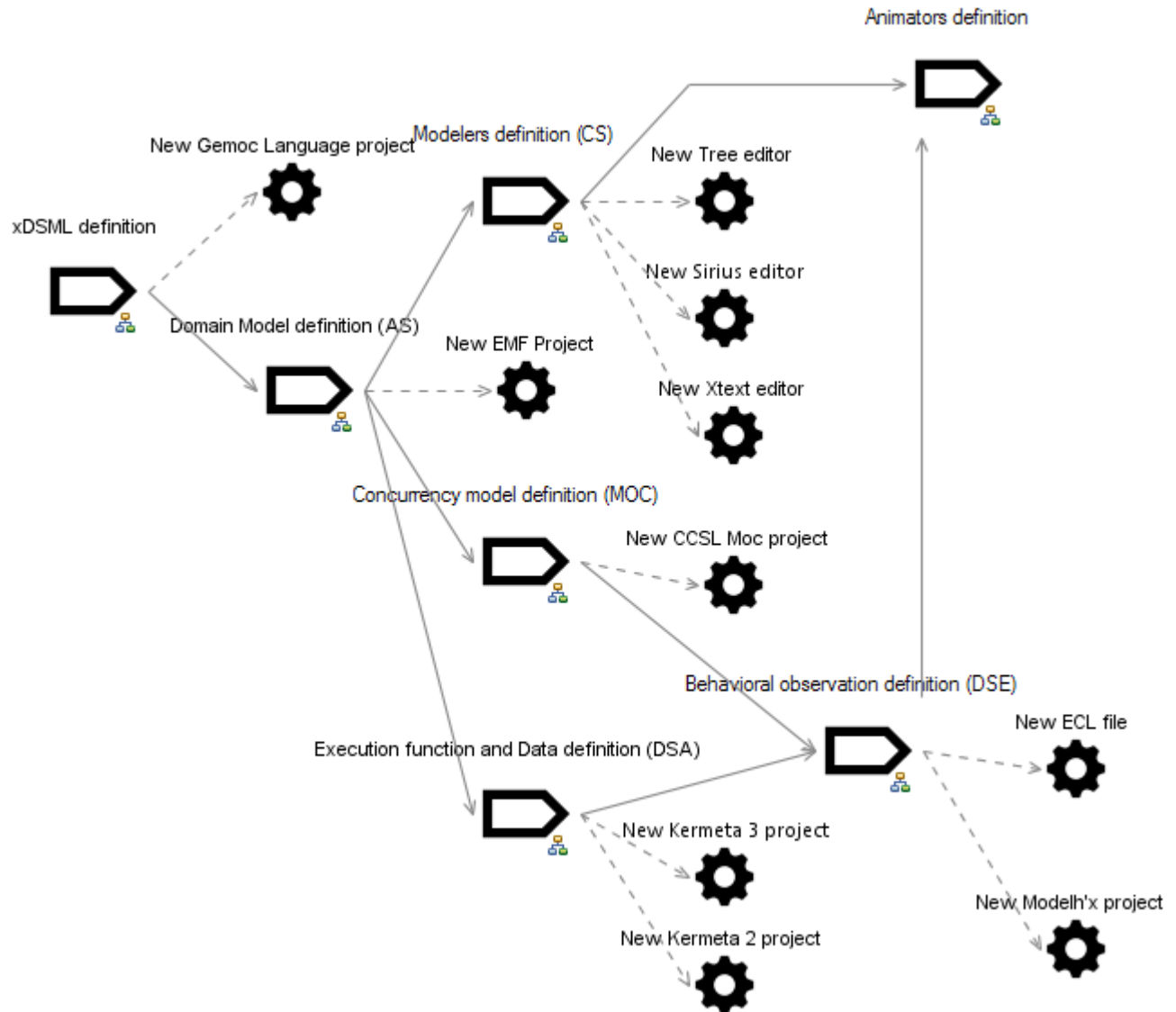






Figure 3.1: Workflow proposed in the GEMOC Language Workbench, and supported tools

-  Activity: an activity is a step of the workflow. Each activity is supported by at least one concrete Command
-  Command: a command is a concrete action of the studio, usually implemented as a wizard.
-  Artifact creation: Artifact created as the result of a command.
-  Artifact update: Artifact updated as the result of a command.

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

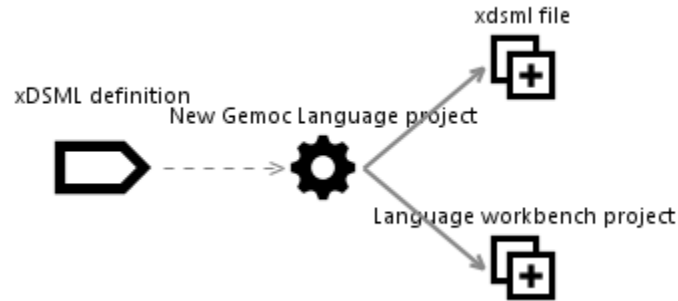


Figure 3.2: xDSML definition activity

### 3.1 xDSML definition

The *Create language workbench definition* Activity is the global activity of assembling the various components of a xDSML tool suite.

The figure 3.2 presents the activity and its supporting Commands.

#### 3.1.1 New Gemoc Language project Command

The *New Gemoc Language project* Command is a wizard that creates the eclipse project hosting the xdsml definition and the glue code connecting the components for this definition.

##### 3.1.1.1 Created artefacts

Artifacts created by the New Gemoc Language project Command:

**Gemoc Language project** This is the eclipse project hosting the xdsml definition and the glue code connecting the components for this definition.

**xdsml file** This is the definition of the language assembly. It acts as a dashboard presenting the implementation choices and provides a link to the concrete implementations components.

##### 3.1.1.2 Updated artefacts

Artifacts updated by the New Gemoc Language project Command:

None

### 3.2 Domain Model definition (AS)

The *Domain Model definition* Activity is the activity of creating the domain model and the components that implement this model.

The figure 3.3 presents the activity and its supporting Commands.

#### 3.2.1 New EMF Project Command

The *New EMF Project* Command is a wizard that creates a new eclipse project hosting the ecore file of the domain definition and its java implementation in an eclipse plugin.

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

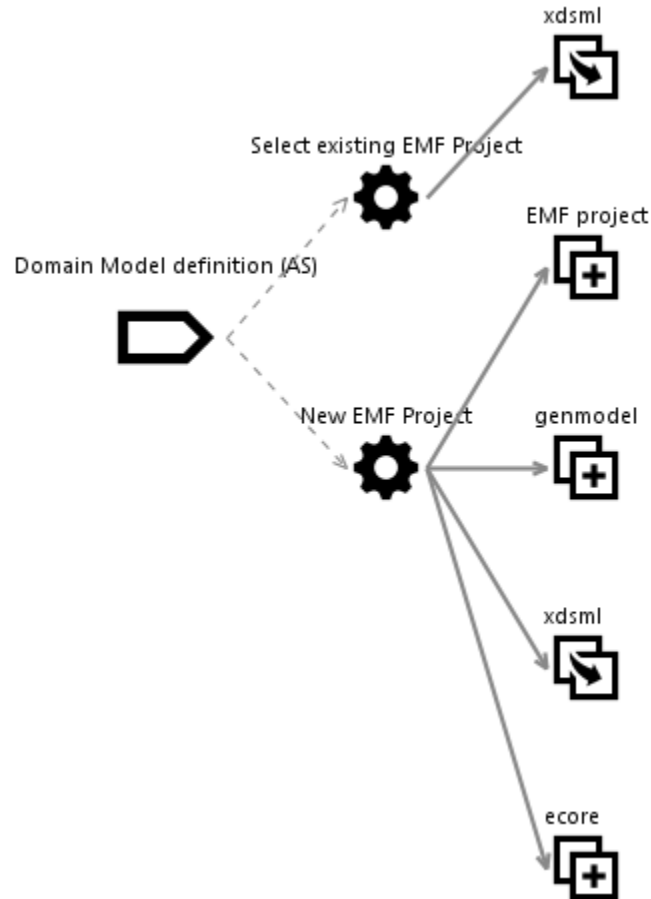


Figure 3.3: Domain Model definition (AS) activity

#### 3.2.1.1 Created artefacts

Artifacts created by the New EMF Project Command:

**EMF project** This artifact is an eclipse plugin project that hosts an ecore file and its java implementation.

**ecore** This artifact is an ecore model representing the domain of the xDSML.

**genmodel** This artifact is a genmodel file that is used internally by the eclipse plugin to build the java implementation.

#### 3.2.1.2 Updated artefacts

Artifacts updated by the New EMF Project Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

### 3.2.2 *Select existing EMF Project Command*

The *New EMF Project* Command is a wizard that selects an eclipse plugin project hosting the ecore file of the domain definition and its java implementation.

#### 3.2.2.1 Created artefacts

Artifacts created by the Select existing EMF Project Command:  
None

#### 3.2.2.2 Updated artefacts

Artifacts updated by the Select existing EMF Project Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

## 3.3 Modelers definition (CS)

This activity will create or associated implementations of concrete syntaxes for the domain Model.

The figure 3.4 presents the activity and its supporting Commands.

### 3.3.1 *New Tree editor Command*

The *New tree editor* Command is in charge of creating the plugins implementing a tree editor using EMF based on the Domain Model.

#### 3.3.1.1 Created artefacts

Artifacts created by the New Tree editor Command:

**Edit project** The *Edit project* is an eclipse project that is part of a tree editor.

**Editor project** The *Editor project* is an eclipse project that is part of a tree editor.

#### 3.3.1.2 Updated artefacts

Artifacts updated by the New Tree editor Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

### 3.3.2 *Select existing Tree editor Command*

This command is a wizard that selects existing eclipse plugin projects hosting a tree editor implementation.

#### 3.3.2.1 Created artefacts

Artifacts created by the Select existing Tree editor Command:  
None

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

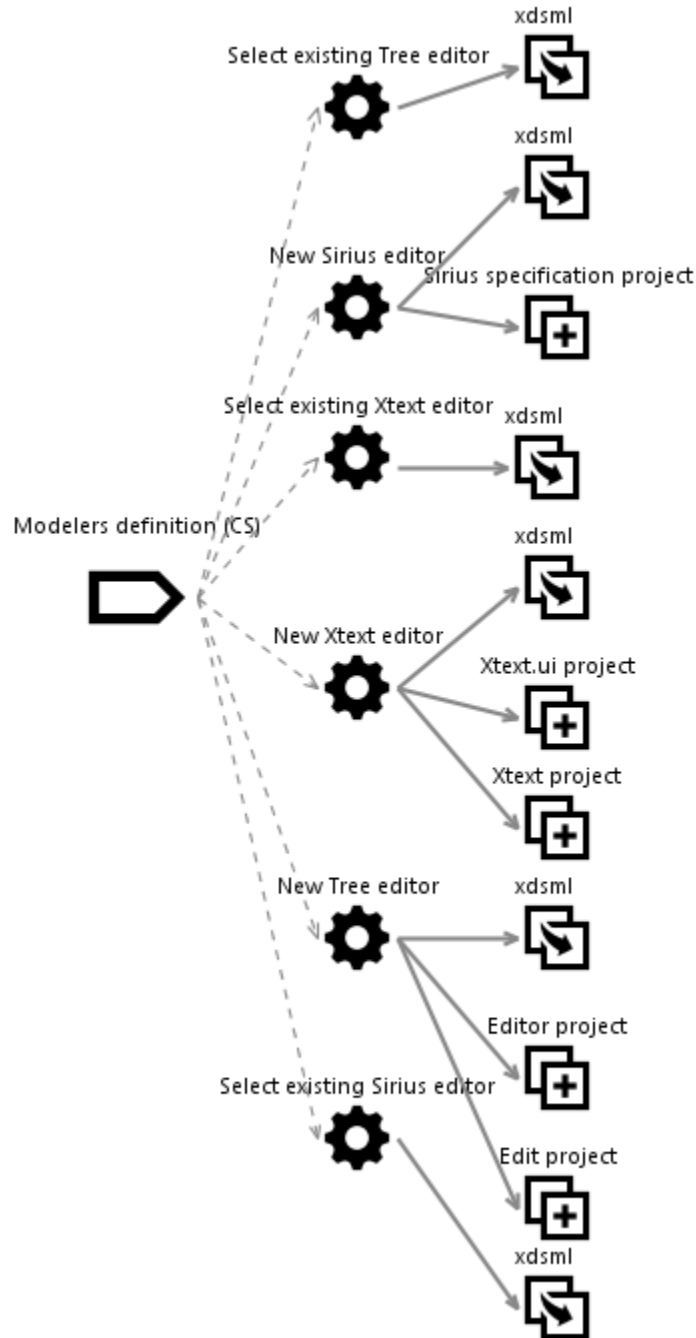


Figure 3.4: Modelers definition (CS) activity

### 3.3.2.2 Updated artefacts

Artifacts updated by the Select existing Tree editor Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.



ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

### 3.3.3 New Xtext editor Command

The *Create Xtext editor* Command is in charge of creating the plugins implementing a textual editor using Xtext based on the Domain Model.

#### 3.3.3.1 Created artefacts

Artifacts created by the New Xtext editor Command:

**Xtext project** The *Xtext project* is an eclipse project that is part of a textual editor editor.

**Xtext.ui project** The *Xtext.ui project* is an eclipse project that is part of a textual editor editor.

#### 3.3.3.2 Updated artefacts

Artifacts updated by the New Xtext editor Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

### 3.3.4 Select existing Xtext editor Command

This Command is a wizard that selects existing eclipse plugin projects hosting a Xtext editor implementation.

#### 3.3.4.1 Created artefacts

Artifacts created by the Select existing Xtext editor Command:  
None

#### 3.3.4.2 Updated artefacts

Artifacts updated by the Select existing Xtext editor Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

### 3.3.5 New Sirius editor Command

The *New Sirius editor* Command is in charge of creating the plugins implementing a textual editor using Sirius framework based on the Domain Model.

#### 3.3.5.1 Created artefacts

Artifacts created by the New Sirius editor Command:

**Sirius specification project** The *Sirius specification project* is an eclipse project that implements a Sirius editor.

#### 3.3.5.2 Updated artefacts

Artifacts updated by the New Sirius editor Command:

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

### 3.3.6 *Select existing Sirius editor Command*

This Command is a wizard that selects existing eclipse plugin projects hosting a Sirius editor implementation.

#### 3.3.6.1 Created artefacts

Artifacts created by the Select existing Sirius editor Command:  
None

#### 3.3.6.2 Updated artefacts

Artifacts updated by the Select existing Sirius editor Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

## 3.4 RunTime Data and Domain-Specific Actions

This is the activity of creating the components that implement the RunTime Data and the Domain-Specific Actions of the xDSML. These components are based on the Abstract Syntax, as explained in Deliverable 1.1.1.

The figure 3.5 presents the activity and its supporting Commands.

### 3.4.1 *New Kermeta 2 project Command*

This command is a wizard that is in charge of creating the plugin implementing the DSAs using Kermeta 2.

#### 3.4.1.1 Created artefacts

Artifacts created by the New Kermeta 2 project Command:

**K2 project** This is an eclipse project written in Kermeta 2 that implements the DSAs of the xDSML.

#### 3.4.1.2 Updated artefacts

Artifacts updated by the New Kermeta 2 project Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

### 3.4.2 *Select existing Kermeta 2 project Command*

This command is a wizard that is in charge of selecting an existing plugin implementing the DSAs using Kermeta 2.

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

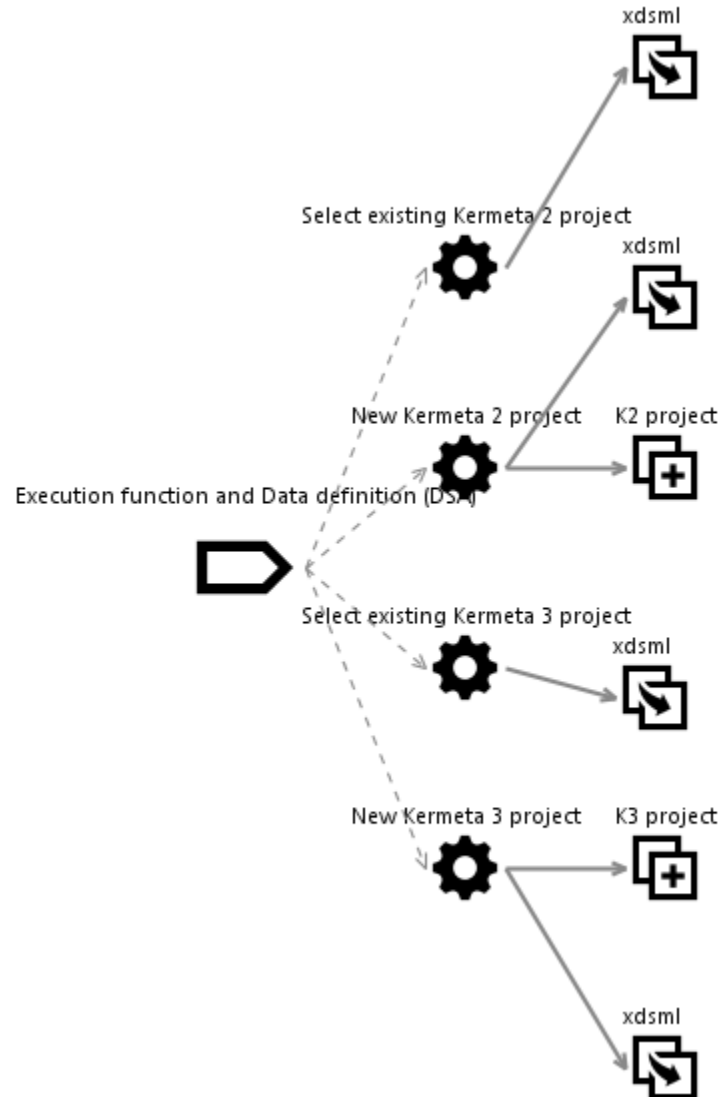


Figure 3.5: RTD and DSAs definition activity

#### 3.4.2.1 Created artefacts

Artifacts created by the Select existing Kermeta 2 project Command:  
None

#### 3.4.2.2 Updated artefacts

Artifacts updated by the Select existing Kermeta 2 project Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

### 3.4.3 *New Kermeta 3 project Command*

This command is a wizard that is in charge of selecting an existing plugin implementing the DSA using Kermeta 3.

#### 3.4.3.1 Created artefacts

Artifacts created by the New Kermeta 3 project Command:

**K3 project** This is an eclipse project written in Kermeta 3 that implements the DSA for the xDSML.

#### 3.4.3.2 Updated artefacts

Artifacts updated by the New Kermeta 3 project Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

### 3.4.4 *Select existing Kermeta 3 project Command*

This command is a wizard that is in charge of selecting an existing plugin implementing the DSA using Kermeta 3.

#### 3.4.4.1 Created artefacts

Artifacts created by the Select existing Kermeta 3 project Command:  
None

#### 3.4.4.2 Updated artefacts

Artifacts updated by the Select existing Kermeta 3 project Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the selected artifacts so the glue can be automatically adapted to use them.

## 3.5 **Model of Computation (MoC) definition**

This activity is in charge of creating the components that implement the model of computation of the Domain Model for the xDSML.

The figure 3.6 presents the activity and its supporting Commands.

### 3.5.1 *New CCSL Moc project Command*

The current implementation supposes that the DSE written in ECL have an import to the MoC definition. This MoC definition is supposed to be a ccsLib file that is bundled in an eclipse plugin.

#### 3.5.1.1 Created artefacts

Artifacts created by the New CCSL Moc project Command:

**eclipse plugin project** This is an eclipse plugin project that hosts and deploys the ccsLib file.

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

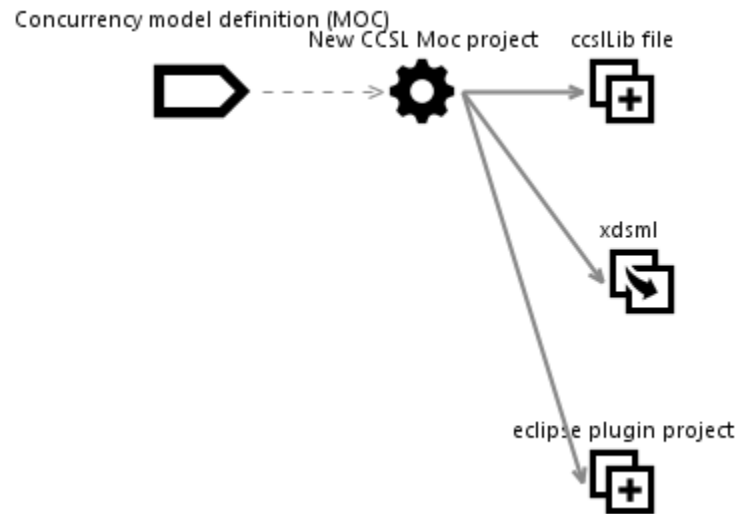


Figure 3.6: Model of Computation definition activity

**ccsLib file** This is the ccsLib file implementing the MOC in CCSL.

#### 3.5.1.2 Updated artefacts

Artifacts updated by the New CCSL Moc project Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

### 3.6 Domain-Specific Events specification file definition

This is the activity of creating the components that implement the Domain Specific Event for the xDSML.

The figure 3.7 presents the activity and its supporting Commands.

#### 3.6.1 New ECL file Command

The command is in charge of creating the plugin hosting the DSEs. This plugin is written using ECL (Event-Constraint Language).

##### 3.6.1.1 Created artefacts

Artifacts created by the New ECL file Command:

**ecl file** The *ecl file* is the implementation of the DSEs written in ECL.

##### 3.6.1.2 Updated artefacts

Artifacts updated by the New ECL file Command:

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

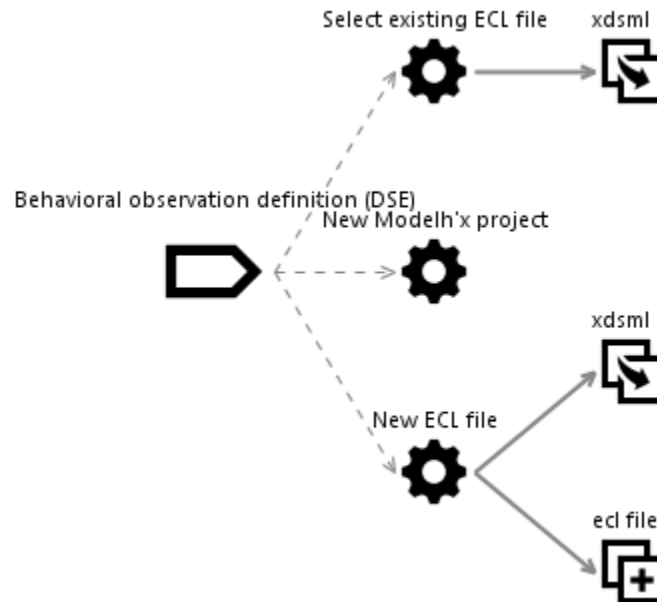


Figure 3.7: Behavioral observation definition (DSE) activity

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the created artifacts so the glue can be automatically adapted to use them.

### 3.6.2 New Modelh'x project Command

Not implemented in this version of the Studio.

#### 3.6.2.1 Created artefacts

Artifacts created by the New Modelh'x project Command:  
None

#### 3.6.2.2 Updated artefacts

Artifacts updated by the New Modelh'x project Command:  
None

### 3.6.3 Select existing ECL file Command

This command is a wizard that is in charge of selecting an existing plugin implementing the DSEs using ECL.

#### 3.6.3.1 Created artefacts

Artifacts created by the Select existing ECL file Command:  
None

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

### 3.6.3.2 Updated artefacts

Artifacts updated by the Select existing ECL file Command:

**xdsml** The command also updates the xdsml artifact in order to indicate the various locations of the ECL file so the glue can automatically create a qvto transformation from it and deploy it.

## 3.7 Animators definition

Animators definition



*Figure 3.8: Animators definition activity*

Not implemented in this version of the Studio.

The figure 3.8 presents the activity and its supporting Commands.

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

## 4. Current implementation

The GEMOC Language Workbench is currently composed of the following eclipse plugins and development projects:

**org.gemoc.gemoc\_language\_workbench.api** Eclipse plugin exposing the interfaces that a xDSML project must offer.

**org.gemoc.gemoc\_language\_workbench.conf.model** EMF Eclipse plugin defining the xdsml.ecore and its implementation classes.

**org.gemoc.gemoc\_language\_workbench.conf.model.edit** Simple tree editor support for xdsml models.

**org.gemoc.gemoc\_language\_workbench.conf.model.editor** simple tree editor for xdsml models.

**org.gemoc.gemoc\_language\_workbench.documentation** Eclipse help associated to the Gemoc Language workbench.

**org.gemoc.gemoc\_language\_workbench.extensions.k3** Eclipse plugin offering an implementation of the execution service for DSA built with Kermeta 3.

**org.gemoc.gemoc\_language\_workbench.feature** Eclipse feature used to deploy the language workbench plugins.

**org.gemoc.gemoc\_language\_workbench.p2updatesite** Description of the update site deploying all these features and plugins. (used by continuous integration).

**org.gemoc.gemoc\_language\_workbench.root** maven root description used by continuous integration to build all the other parts.

**org.gemoc.gemoc\_language\_workbench.sample.deployer** Eclipse plugin deploying the examples illustrating the Language workbench. It currently deploys the tfsm example.

**org.gemoc.gemoc\_language\_workbench.ui** Main eclipse plugin for the language workbench. It implements the workflow and its commands. It also implements the builder that updates the xDSML project according to the definition.

**org.gemoc.gemoc\_language\_workbench.utils** Eclipse plugin containing various reusable code.



ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

## 5. Conclusion

The software deliverable D1.2.1 is a set of tools supporting the process to define an xDSML.

The various steps of the process for developing an xDSML are summarized in a dashboard (currently presented as a tree view). The dashboard uses several wizards to drive the end user.

All the tools and the wizards supporting the process are integrated and deployed into the GEMOC studio<sup>1</sup>. The source code is available on the git server<sup>2</sup>

---

<sup>1</sup>Cf. [http://ci.inria.fr/gemoc/job/org.gemoc.gemoc\\_studio.root/](http://ci.inria.fr/gemoc/job/org.gemoc.gemoc_studio.root/)

<sup>2</sup>Cf. <gitclonegit+ssh://developer-name@scm.gforge.inria.fr//gitroot/gemoc-dev/gemoc-dev.git>

ANR INS GEMOC / Task 1.2.1	Version:	0.2
DSML behavioral semantics definition tools	Date:	January 6, 2014
D1.2.1		

## A. Getting started with the Language Workbench

This annex presents some of the major steps to get started with the current GUI of the workflow tooling. It will use one the example that is bundled in the GEMOC Studio (TFSM). It shows how to install the example. It also shows some of the basic GUI commands used to create or navigate an xDSML by aggregating components with a xDSML project and model.

Understanding the TFSM example itself and how its aggregated components are designed internally is not in the scope of this document. To understand the design of the xDSML TFSM and play with it, we refer the reader to <http://gemoc.org/sle13/> (including videos).

- Download the GEMOC Studio from [http://ci.inria.fr/gemoc/job/org.gemoc.gemoc\\_studio.root/](http://ci.inria.fr/gemoc/job/org.gemoc.gemoc_studio.root/).
- Install the TFSM example definition. This is done by navigating in eclipse menu: File > new > Example... > GEMOC TFSM Language example (see figureA.1).
- In this example you can open the project.xdsml file in order to have a dashboard that synthetizes the current components used by the xDSML definition. As shown on figure A.2, in the current version, the dashboard is presented as a tree view.
- The workflow actions are accessible via a right click on the xdsml file or on the project hosting the file. These actions are in the menu labeled *Gemoc Language*, (see figure A.2.)
- Using the wizard via the actions will automatically update the project.tfsml accordingly. The project.xdsml file will in turn be used in the background in order to update the org.gemoc.sample.tfsml project so it can act as the glue between the elements that are part of the xDSML. (update of the dependencies, creation of factories, ...)

ANR INS GEMOC / Task 1.2.1	Version: 0.2
DSML behavioral semantics definition tools	Date: January 6, 2014
D1.2.1	

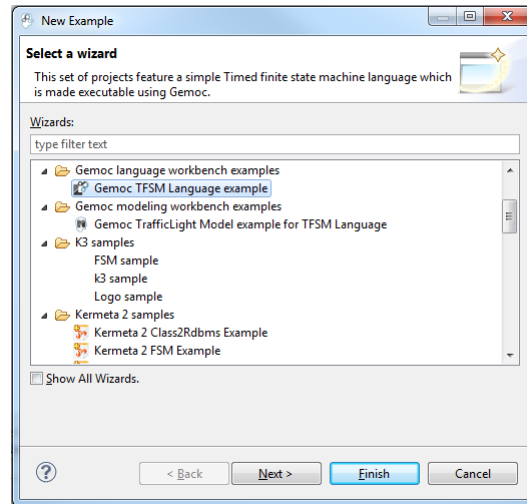


Figure A.1: install TFSM language example screenshot

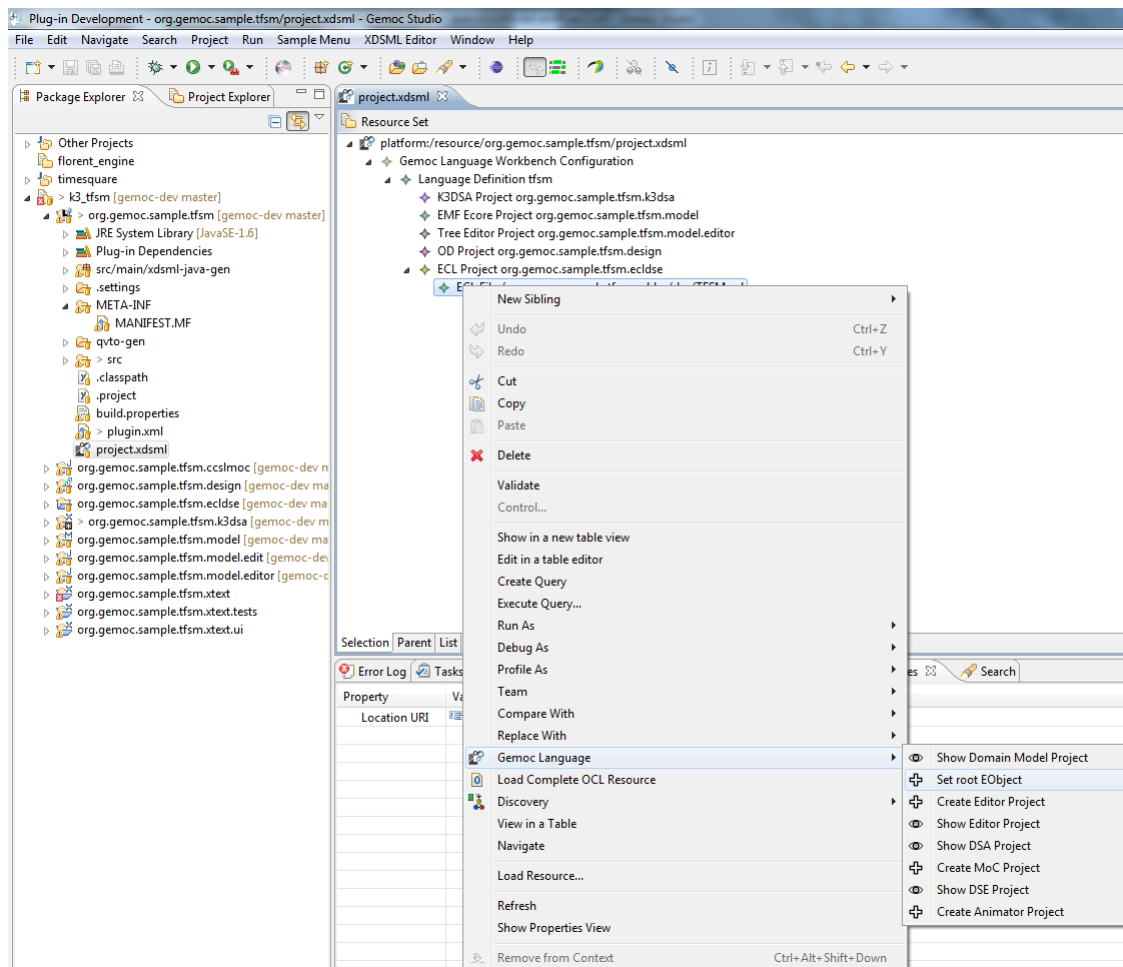


Figure A.2: Dashboard and activity menu screenshot