OETCS

openETCS

ITEA2

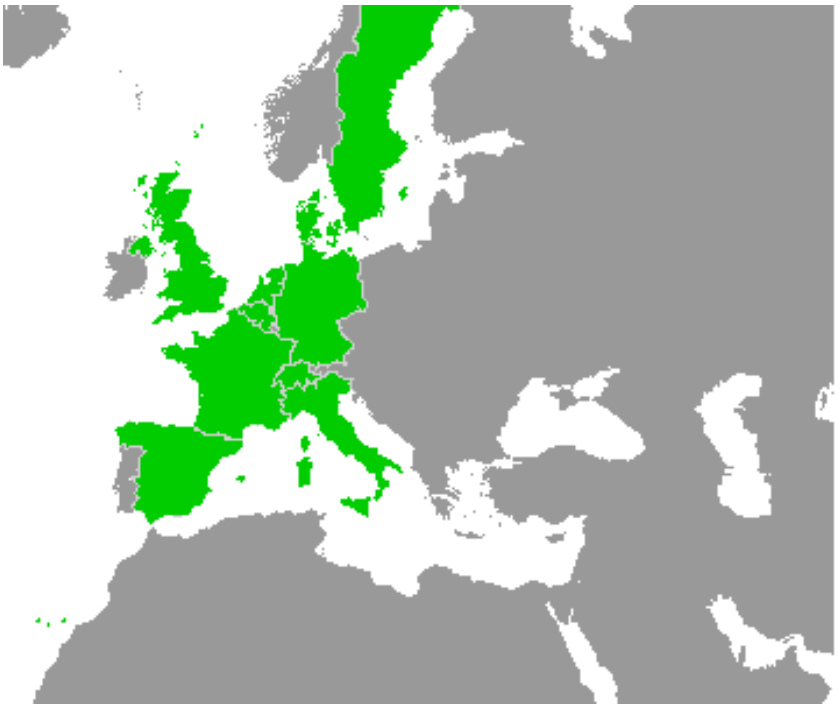
INFORMATION TECHNOLOGY FOR EUROPEAN ADVANCEMENT

ITEA2 Project
Call 6 11025
2012 – 2015

Using Rodin with Projects on github

Matthias Güdemann
Systerel, France

June 2013



This page is intentionally left blank

OETCS
June 2013

Using Rodin with Projects on github

Matthias Güdemann
Systerel, France

Model Description

Prepared for openETCS@ITEA2 Project

Table of Contents

Figures and Tables

Figures

Tables

1 Short Introduction to Event-B

The formal language Event-B is based on a set-theoretic approach. It is a variant of the B language, with a focus on system level modeling **?**. An Event-B model is separated into a static and a dynamic part.

The dynamic part of an Event-B model describes abstract state machines. The state is represented by a set of state variables. A transition from one state to another is represented by parametrized events which assign new values to the state variables. Event-B allows unbounded state spaces. They are constrained by invariants expressed in first order logic with equality which must be fulfilled in any case. The initial state is created by a special initialization event.

The static part of an Event-B model is represented by contexts. These consist of carrier sets, constants and axioms. The type system of a model is described by means of carrier sets and constraints expressed by axioms.

Event-B is not only comprised of descriptions of abstract state machines and contexts, but also includes a modeling approach. This approach consists of iterative refinement of the machines until the desired level of detail is reached. In the Rodin tool, proof obligations are automatically created which ensure correct refinement.

Together with the machine invariants, the proof obligations for the refinement are formally proven, creating proof trees. To accomplish this, there are different options: many proof obligations can be discharged by automated provers (e.g., AtelierB, NewPP, Rodin's SMT-plugin), but as the underlying logic is in general undecidable, it is sometimes necessary to use the interactive proof support of Rodin.

Any external actions, e.g., mode changes by the driver or train level changes are modeled via parametrized events. Only events can modify the variables of a machine. An Event-B model is on the system level, events are assumed to be called from a software system into which the functional model is embedded. The guards of the events assure that any event can only be called when appropriate.

2 Prerequisites

This section shortly describes the basic prerequisites to use Rodin projects on github. First the installation of the Rodin platform itself, then the basic plugins required to use the provided models and finally additional plugins which facilitate usage and extension of the provided models.

2.1 Rodin Platform

This illustration uses the Rodin platform 2.7 [1], for general information on Event-B, Rodin, various plugins etc. see the Event-B Wiki [2]. More details on the installation on Rodin can be found at `http://www.event-b.org/install.html`.

2.2 Basic Plugins

[1] `http://www.event-b.org/`
[2] `http://wiki.event-b.org/index.php/Main_Page`

For plugin installations, it is recommended to tick the “Contact all update sites during install to fin required software” (see Figure **??**). This will install all necessary dependencies for each plugin, in case these are not yet installed.

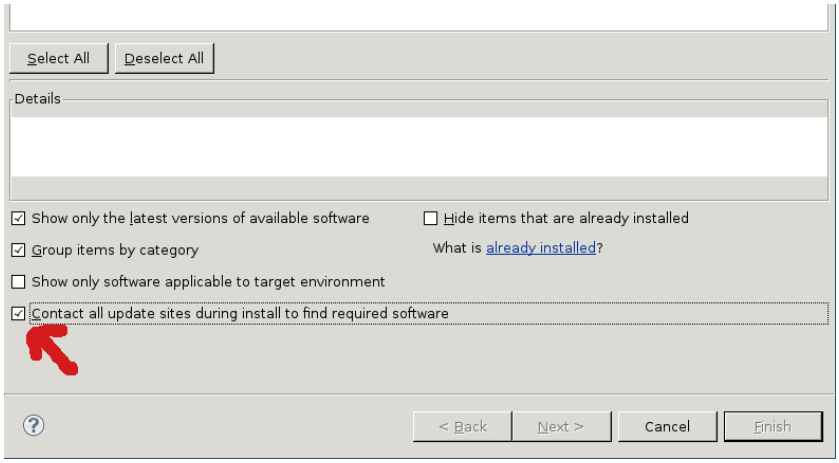


Figure 1. Plugin Installation

Provers

The Atelier-B provers facilitate the construction of the proof trees for Rodin proof obligations, by automatically discharging may proof obligations. Their repository is available under Rodin as “Atelier B Provers”.

Note, that these provers are not open source [3]. Fully free provers are available, i.e., NewPP which is installed by default and the SMT plugin prover (see **??**).

ProR Integration

The integration with ProR allows for the traceability of requirements in the provided ReqIf files. The repository is available under Rodin as “ProR”, where the “ProR Rodin integration feature” must be installed. **NOTE:** For the current ProR version 0.6.1 there are known problems with Java version 6. It is suggested to use Java version 7, if possible the latest JVM from `http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html`.

ProR is available under the EPL license.

EGit

The integration with github is done via the EGit plugin. This plugin allows to collaborate on Rodin models and to push/pull changes to github. The repository is available under the official Eclipse repositories as “Indigo Update Site” where the “Eclipse EGit” must be installed.

EGit is available under the EPL license.

Model Decomposition

The model decomposition plugin allows for structurization of Event-B models. Larger models can be decomposed into smaller models with shared variables or shared events. The decomposition

[3] `http://www.atelierb.eu`

keeps the correct proven refinement relations and allows for independent refinement of the decomposed machines.

The model decomposition plugin is available under the EPL license.

iUML-B State Machines

The "iUML-B State-Machines" plugin allows for graphical modeling of state-machines in an Event-B model. The state machines can be transformed into Event-B code. It has a good integration with the ProB plugin which allows for graphical animation of the machines.

The iUML plugin is available under the EPL license.

2.3 Additional Plugins

The additional plugins are not strictly necessary to analyze and inspect the model. Nevertheless, it is suggested to install them if an extension of the model is intended.

ProB / AnimB

The ProB plugin provides means for model-checking and animation of Event-B models. This requires finite instantiation of carrier sets and selection of an initial state. In this situation, the plugin can verify deadlock freeness and LTL formulas or can animate a system run. It is available from the ProB repository under "ProB for Rodin 2".

AnimB also provides animation capabilities for Event-B machines. It is available from the AnimB repository under "AnimB"

ProB and AnimB are available under the EPL license.

SMT Solver Plugin

The SMT solver plugin will in general lead to a higher degree of automation for the formal proofs. Experiments with industrial cases studies reduced the number of non-automatically discharged proof obligations to one fourth. The plugin comes bundled with two different solvers (CVC3[4] and veriT[5]) and it can be extended with various others, e.g., z3 from Microsoft or MathSAT5 from FBK. It is available from the Rodin repository under "SMT Solvers Integration".

The SMT plugin is available under the EPL license.

Project Diagram

The project diagram plugin allows for visualization of the structure of a model. In particular it visualizes the refinement relations between the machines, the extension relation between the contexts and the sees relation between machines and contexts. It is available from the Rodin repository under "Event-B Project Diagram Plugin".

Team-Based Development

[4] http://www.cs.nyu.edu/acsys/cvc3/
[5] http://www.verit-solver.org/

The team-based development plugin[6] allows an EMF-compare comparison of Event-B models. Each context and each machine is stored in a separate XML file. Although text-based, XML files are not easily readable which makes merging of different versions difficult in a collaborative version control environment. This plugin allows for the presentation of differences between two versions on the model level, which facilitates cooperation of several people on the same model.

3 Importing Rodin Projects from github

After the installation of Rodin and the necessary plugins, projects can be imported from github into the local Rodin workspace. In the following, this is explained using the Eclipse project creation wizard.

The first step is to select "File → Import" from the Eclipse menu and then to select "Git → Projects from Git" as import source (see Figure **??**).

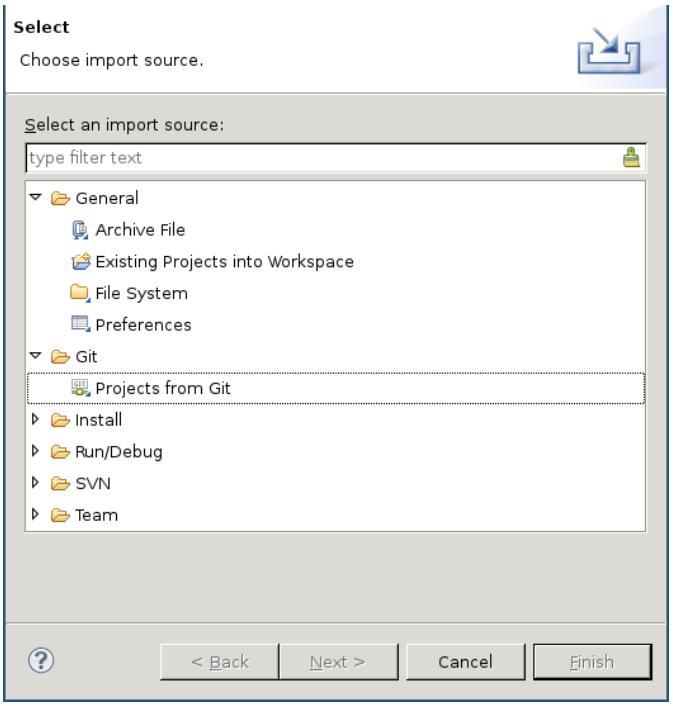


Figure 2. Creating a New Project

The next step is to specify where to find the git repository. For this, one has to select the URI option as shown in Figure **??**.

[6] `http://wiki.event-b.org/index.php/Team-based_development`

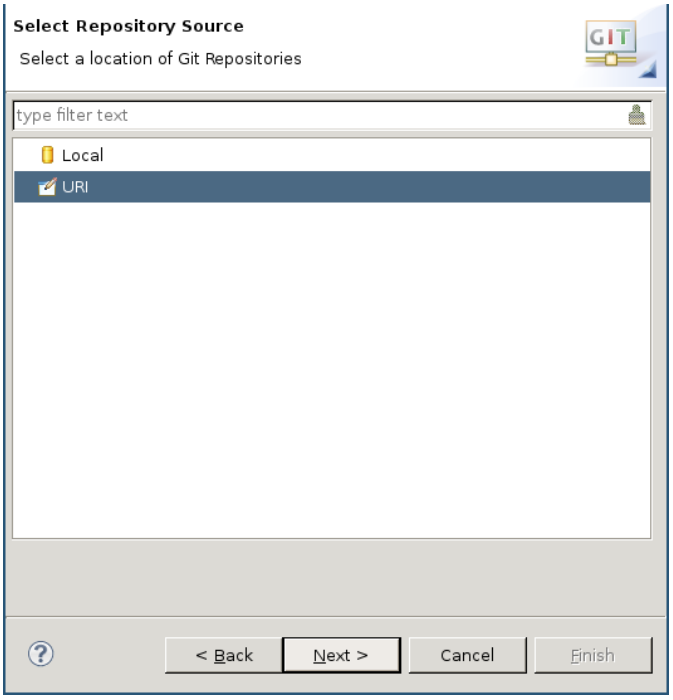


Figure 3. Selection of Repository Source

The next step is to specify the URI of the repository on github. For the model evaluation project, this is `https://github.com/openETCS/model-evaluation.git`. This step also requires the specification of the authentication data, i.e., the username and password for github (see Figure **??**).

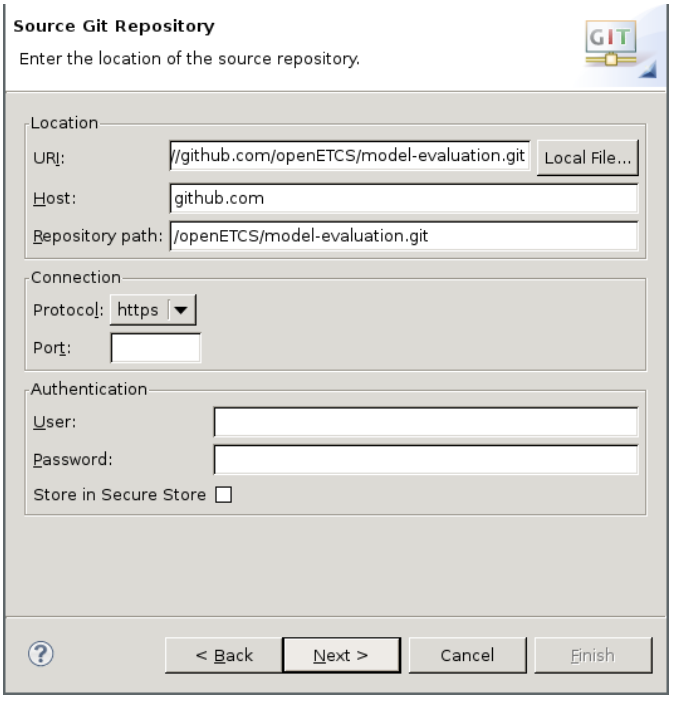


Figure 4. Specify github Repository

The next step is to select the desired branch of the repository. If in doubt and there is more than one branch, selecting the “master” branch, as shown in Figure **??** should be ok.

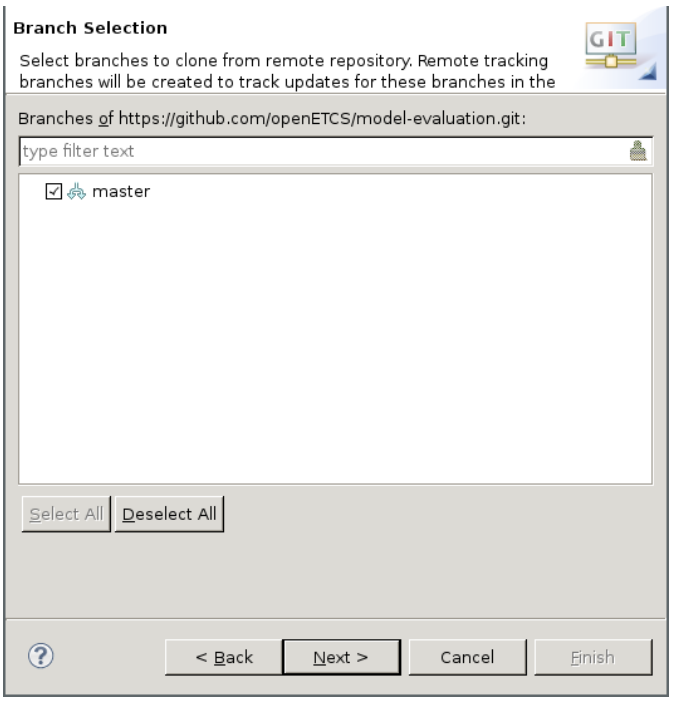


Figure 5. Select Branch of Repository

For a local copy of the repository on github, an empty directory on the local machine must be selected (or created) and a name of the remote repository can be specified, the default is “origin” (see Figure **??**).

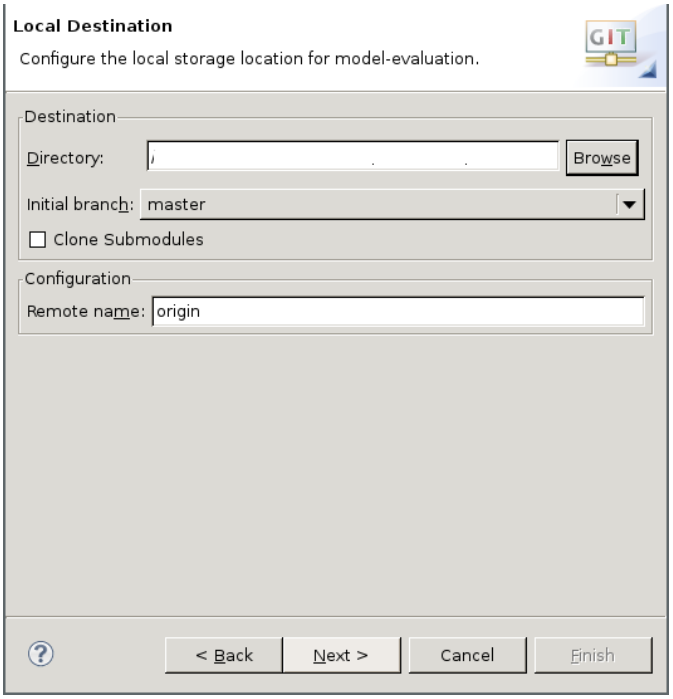


Figure 6. Local Repository Copy

As there can be multiple projects in a repository, e.g., from different tools as for the model-evaluation project, the correct one must be selected. To achieve this, the collapsed directory tree as shown in Figure **??** must be expanded[7].

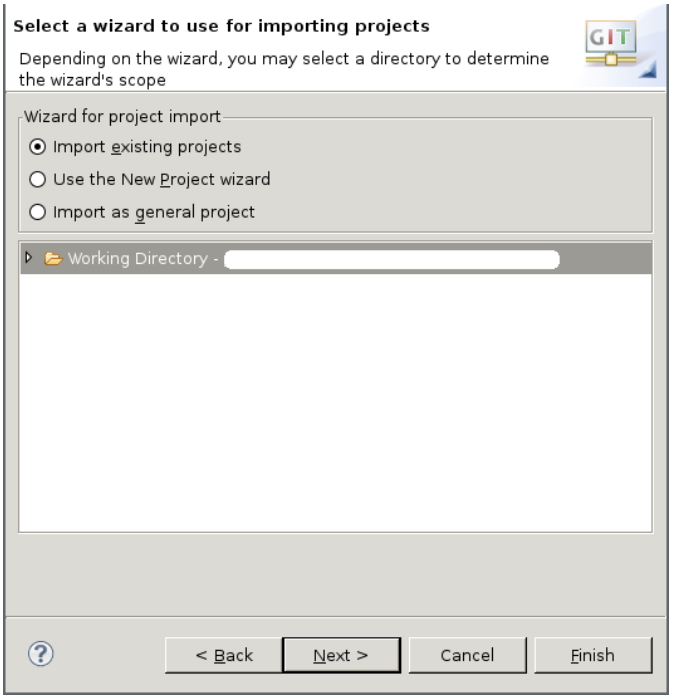


Figure 7. Select Project to Import

In the expanded tree, the project root directory of the project must be selected as shown in Figure **??**.

[7]Sometimes the information which directories are available is not shown directly (the collapse / expand symbol is lacking besides the “working directory”). In this case it helps to push the “back” button to the previous window and return with “next”.

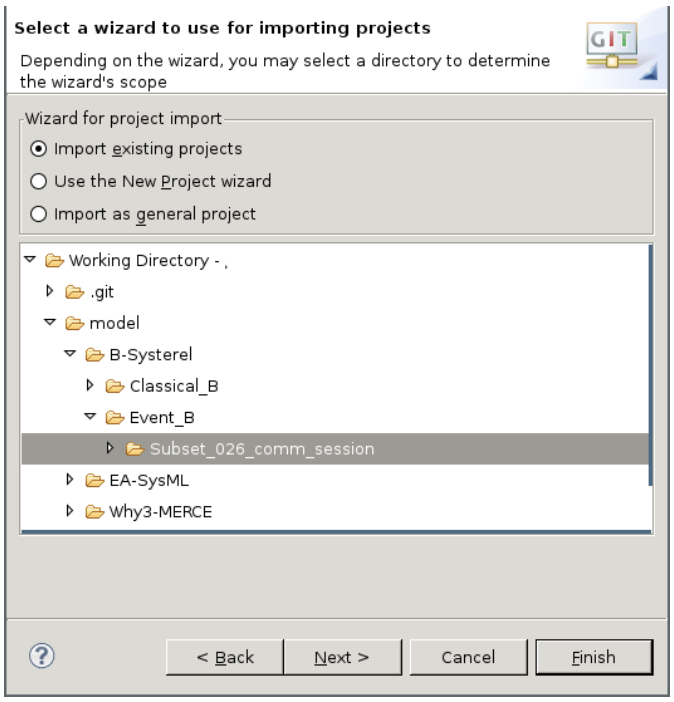


Figure 8. Expanded Directory Tree

Once the correct directory has been selected, the contained projects can be selected as shown in Figure **??** and imported into the local Rodin workspace.

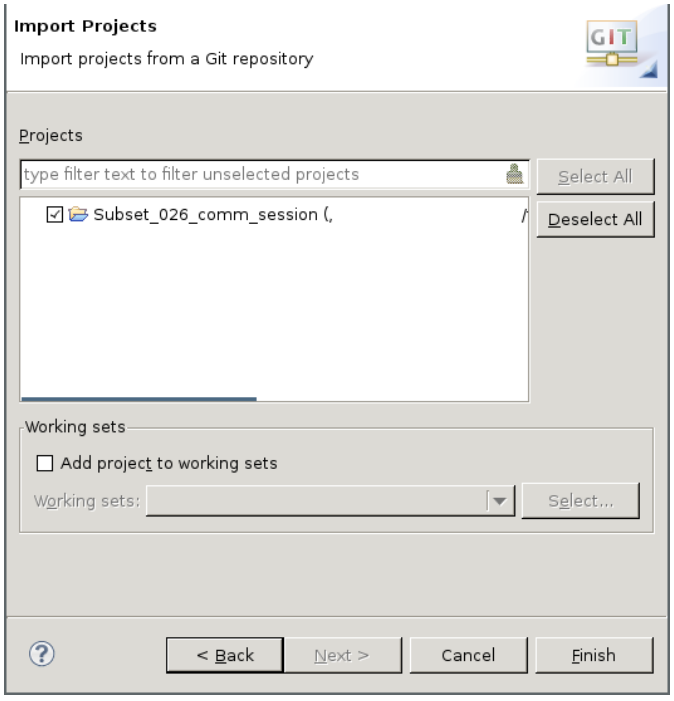


Figure 9. Import Project into Workspace

Once the project is imported into the workspace, it will appear in the Event-B project explorer. There it can be updated to the current revision on the git repository be right-clicking on the project and selecting “Team → Pull”.

4 Frequently Asked Questions - FAQ

4.1 How are requirements traced?

For requirements tracing we use the ProR plugin [8] from formalmind [9]. It is based on the OMG standardized ReqIf format. It allows for tracing of Event-B modeling artifacts in ReqIf files. Changes in either the model or the ReqIf are marked automatically, the ProR view of Rodin is shown in Figure **??**.

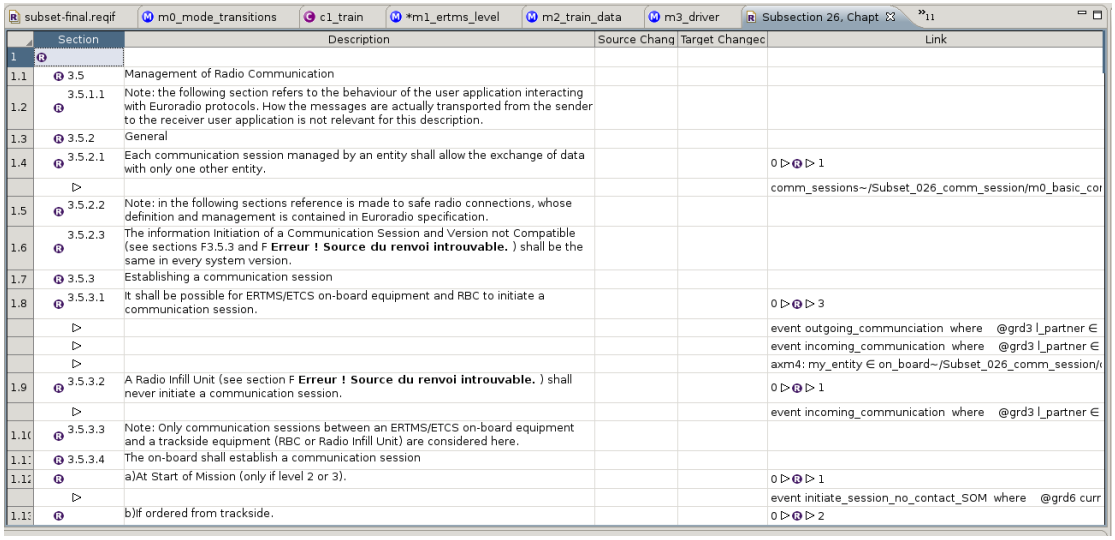


Figure 10. Requirements Tracing with ProR

To open a ReqIf file, the “Resources” view of Rodin must be selected which will show all files in the project in the Project-Explorer.

4.2 Proof obligations are not shown in the Event-b explorer. How to create them?

This can have different causes. Trivial proof obligations are not shown, this concerns in particular proofs for well-definedness [10]. It is also possible to hide proven proof obligations by toggling the green button on top of the Event-B Explorer shown in Figure **??** (button is inactive in this example).

[8] http://wiki.event-b.org/index.php/ProR
[9] http://www.formalmind.com
[10] see `http://handbook.event-b.org/current/html/well_definedness_proof_obligations.html`

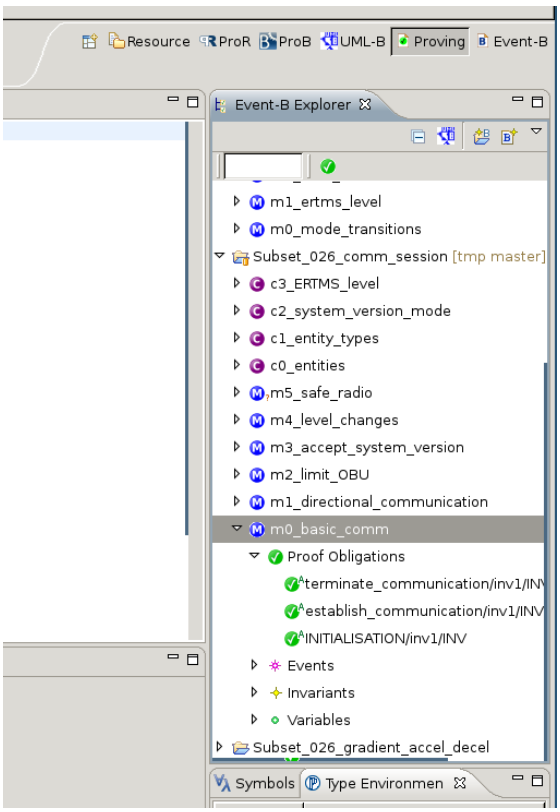


Figure 11. Event-B Project Tree View with Proof Obligations

Another possibility is a problem with some installed plugins, in this case the project can be cleaned by selecting “Project->Clean” from the main Rodin menu. If “Project->Build Automatically” is selected the project will be automatically refreshed, this is also possible manually by selecting the project, hitting F5 or selecting “Refresh” from the right-click menu.

4.3 If I try to open a ReqIf file, Rodin nothing happens and Rodin freezes. What happened?

There is a problem in the ProR plugin which can cause this behavior (e.g., see here `https://bugs.eclipse.org/bugs/show_bug.cgi?id=397672`). Currently the best solution is to install the most recent JDK from `http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html`.

4.4 How can I see the definition of an Event-B model artifact which is linked in a ReqIf file?

When opening a ReqIf file, the standard view only displays whether a requirement is linked or not. To see all linked model elements, the “SpecRelations (Links)” button must be toggled (see Figure **??**).

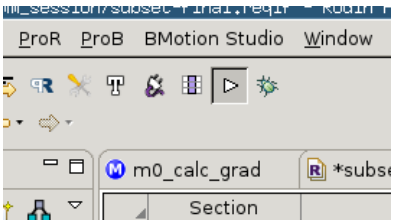


Figure 12. Active SpecRelations Button

When active, each linked requirements shows a list of linked model elements. When selecting one of these elements, its textual Event-B definition is shown in the “Properties” window (see Figure **??**).

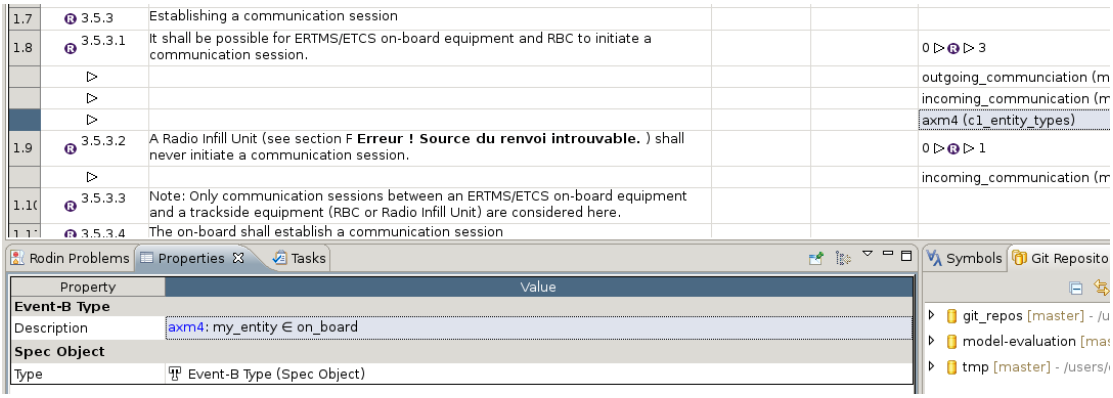


Figure 13. Properties Window with Element Definition

4.5 How can I animate iUML state machines?

When a state machine is opened and selected, the buttons to control the animation and generation of the state machine are activated as shown in Fig. **??**.

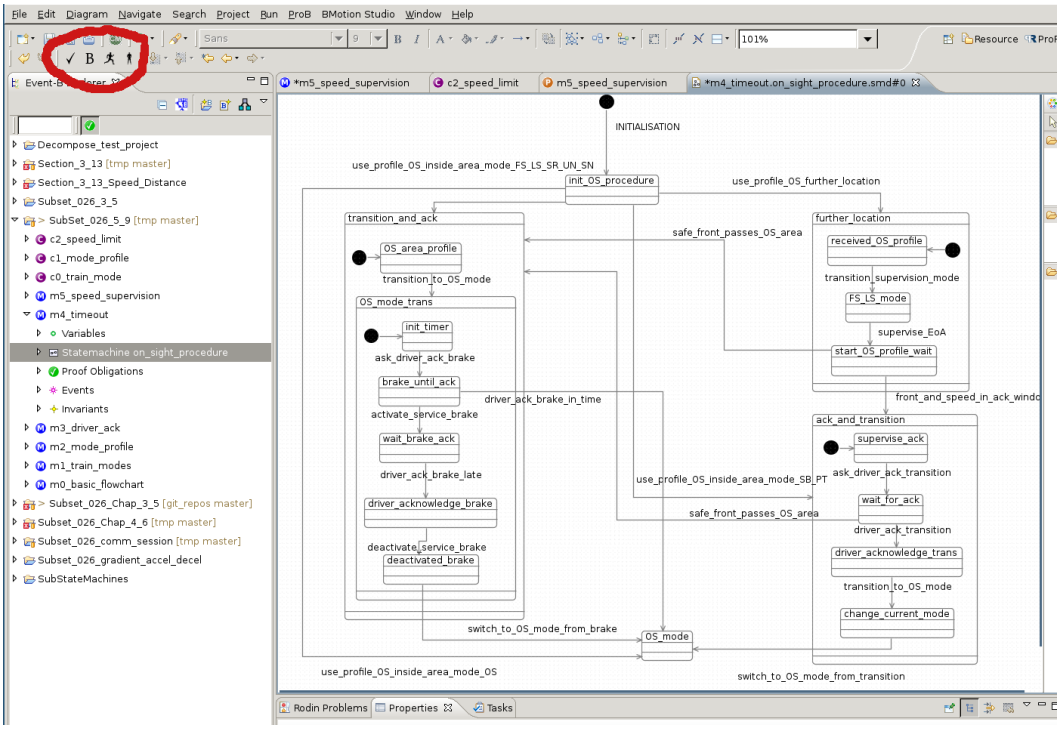


Figure 14. Opening State Machine for Animation

After selecting the “Animate With ProB” button, the state machine is opened in the animation view as shown in Fig. **??**. The active transitions are highlighted (middle), the activated events are listed and can be executed by clicking on them (upper left) and the current state of the variables of the Event-B machines is shown (bottom).

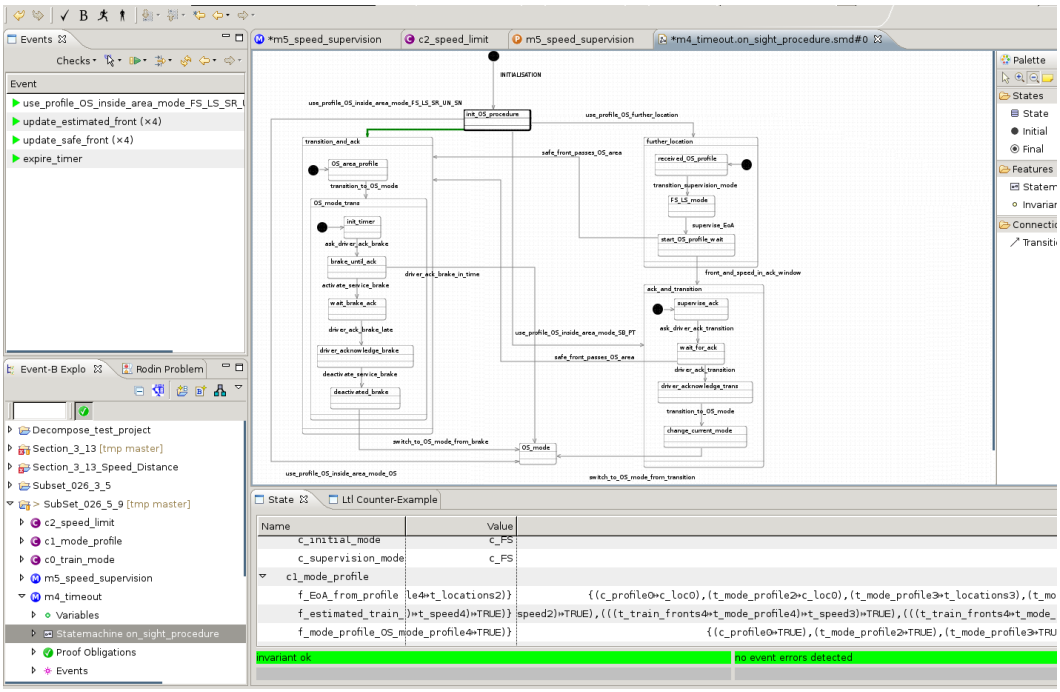


Figure 15. Animation of State Machine