CSC 4504 : Langages formels et applications

(Event-B and Rodin)

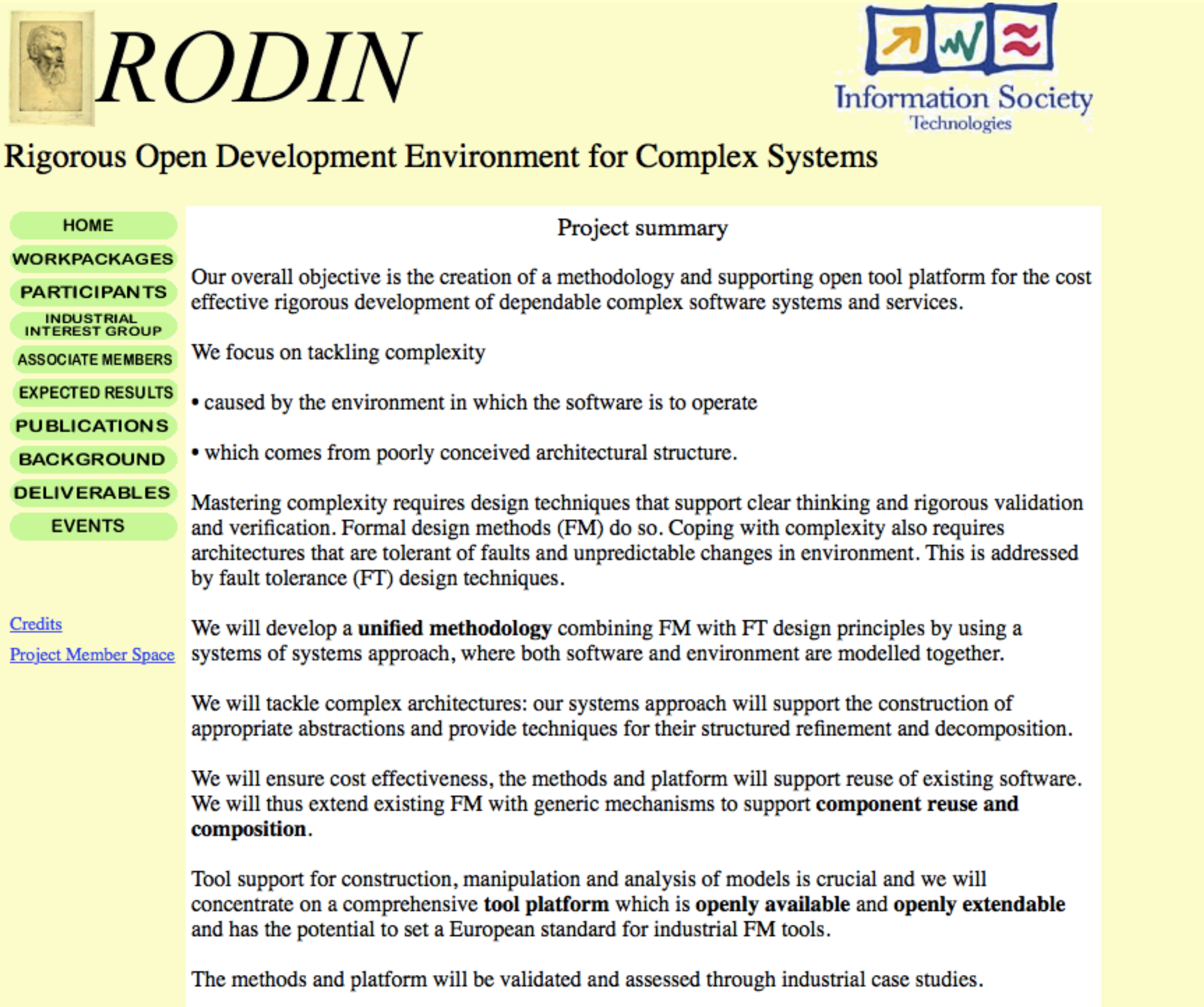
J Paul Gibson, D311

paul.gibson@it-sudparis.eu

http://www-public.it-sudparis.eu/~gibson/Teaching/CSC4504/

Introduction to automated proof - Odd and Even Properties

http://rodin.cs.ncl.ac.uk/ The EU project home page



Rigorous Open Development Environment for Complex Systems

Project summary

Our overall objective is the creation of a methodology and supporting open tool platform for the cost effective rigorous development of dependable complex software systems and services.

We focus on tackling complexity

- caused by the environment in which the software is to operate

- which comes from poorly conceived architectural structure.

Mastering complexity requires design techniques that support clear thinking and rigorous validation and verification. Formal design methods (FM) do so. Coping with complexity also requires architectures that are tolerant of faults and unpredictable changes in environment. This is addressed by fault tolerance (FT) design techniques.

We will develop a **unified methodology** combining FM with FT design principles by using a systems of systems approach, where both software and environment are modelled together.

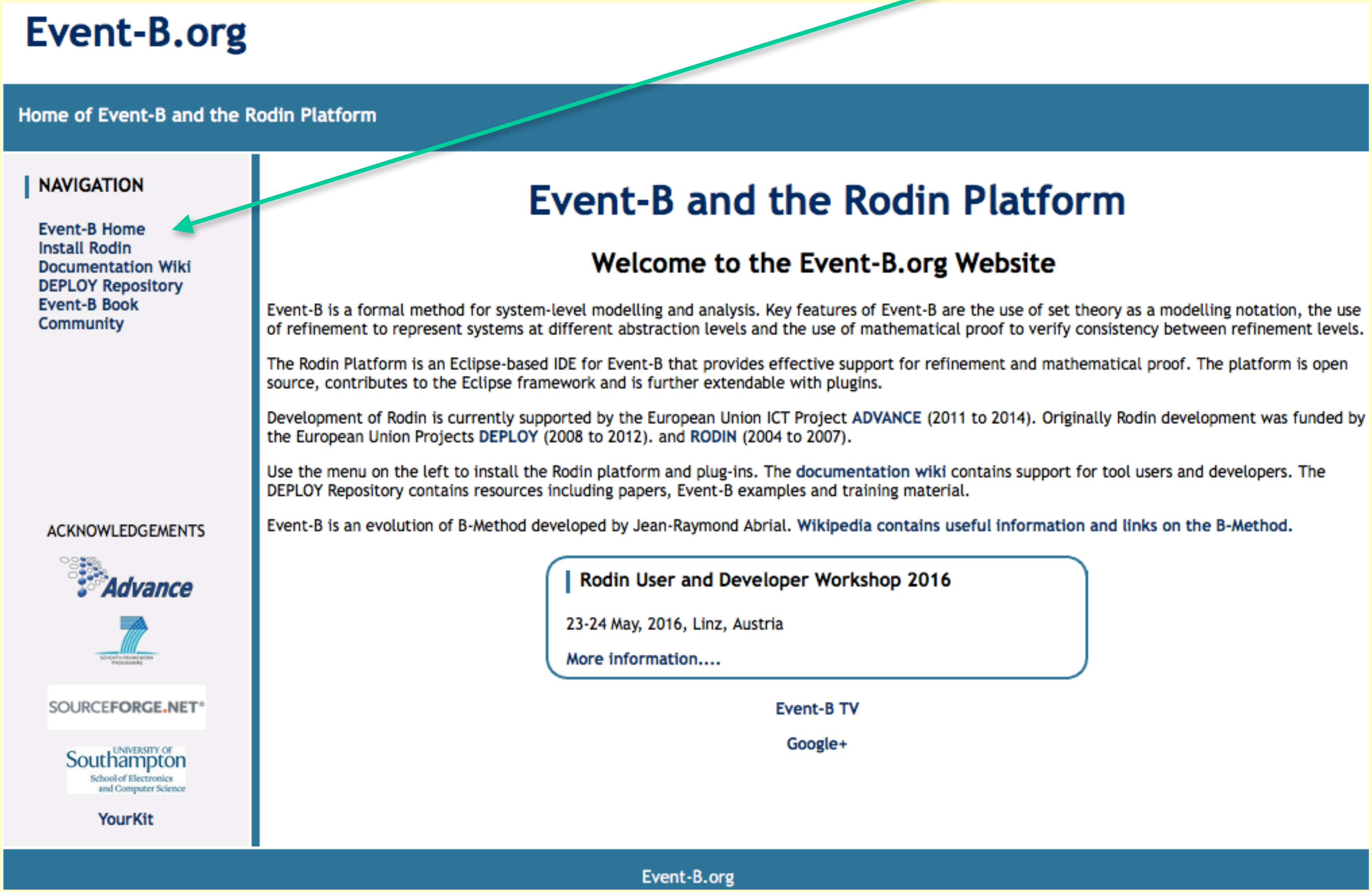
We will tackle complex architectures: our systems approach will support the construction of appropriate abstractions and provide techniques for their structured refinement and decomposition.

We will ensure cost effectiveness, the methods and platform will support reuse of existing software. We will thus extend existing FM with generic mechanisms to support **component reuse and composition**.

Tool support for construction, manipulation and analysis of models is crucial and we will concentrate on a comprehensive **tool platform** which is **openly available** and **openly extendable** and has the potential to set a European standard for industrial FM tools.

The methods and platform will be validated and assessed through industrial case studies.

http://www.event-b.org/index.html



Install Rodin

Event-B.org

Home of Event-B and the Rodin Platform

NAVIGATION

- Event-B Home
- Install Rodin
- Documentation Wiki
- DEPLOY Repository
- Event-B Book
- Community

ACKNOWLEDGEMENTS

YourKit

Event-B and the Rodin Platform

Welcome to the Event-B.org Website

Event-B is a formal method for system-level modelling and analysis. Key features of Event-B are the use of set theory as a modelling notation, the use of refinement to represent systems at different abstraction levels and the use of mathematical proof to verify consistency between refinement levels.

The Rodin Platform is an Eclipse-based IDE for Event-B that provides effective support for refinement and mathematical proof. The platform is open source, contributes to the Eclipse framework and is further extendable with plugins.

Development of Rodin is currently supported by the European Union ICT Project ADVANCE (2011 to 2014). Originally Rodin development was funded by the European Union Projects DEPLOY (2008 to 2012). and RODIN (2004 to 2007).

Use the menu on the left to install the Rodin platform and plug-ins. The documentation wiki contains support for tool users and developers. The DEPLOY Repository contains resources including papers, Event-B examples and training material.

Event-B is an evolution of B-Method developed by Jean-Raymond Abrial. Wikipedia contains useful information and links on the B-Method.

Rodin User and Developer Workshop 2016

23-24 May, 2016, Linz, Austria

More information....

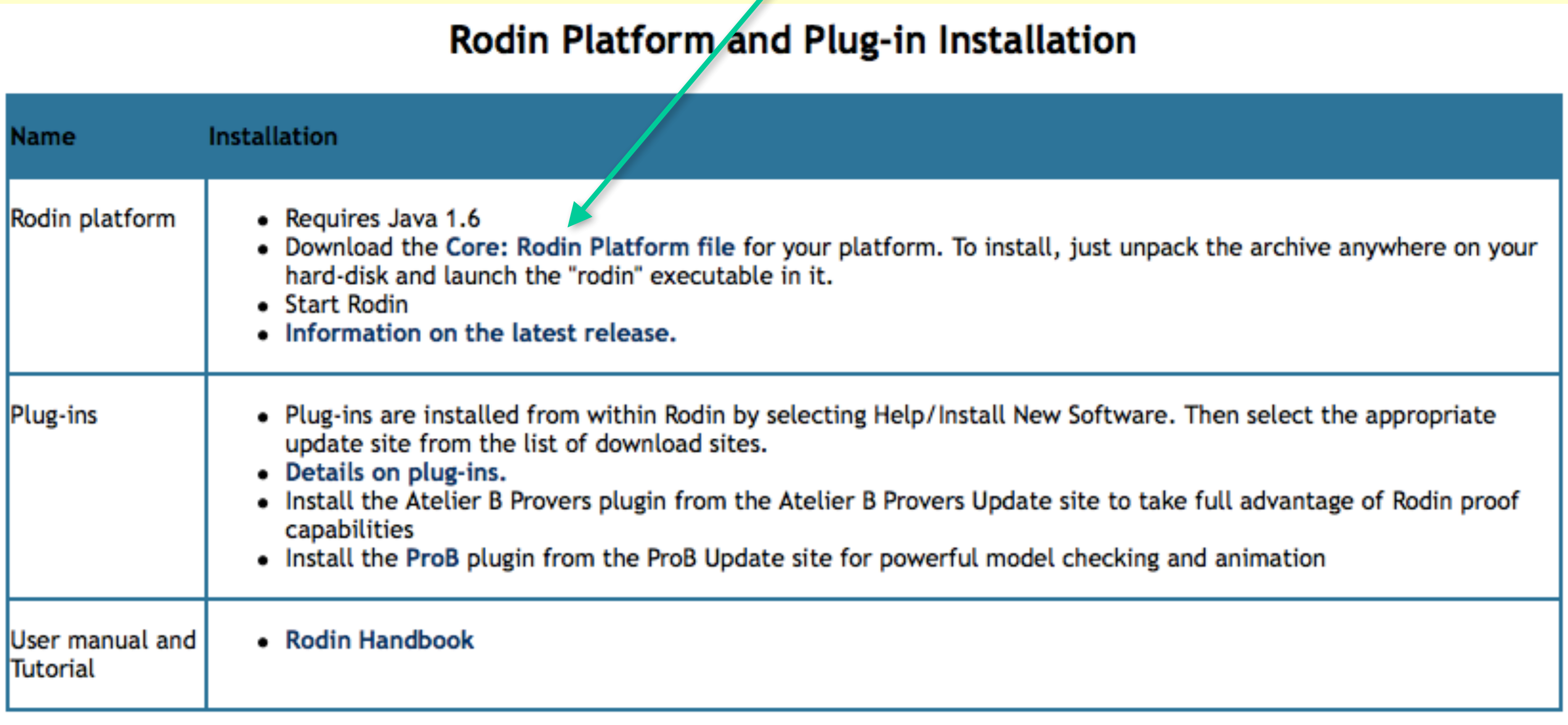
Event-B TV

Google+

Event-B.org

Install the Event-B RODIN tool

Download the core

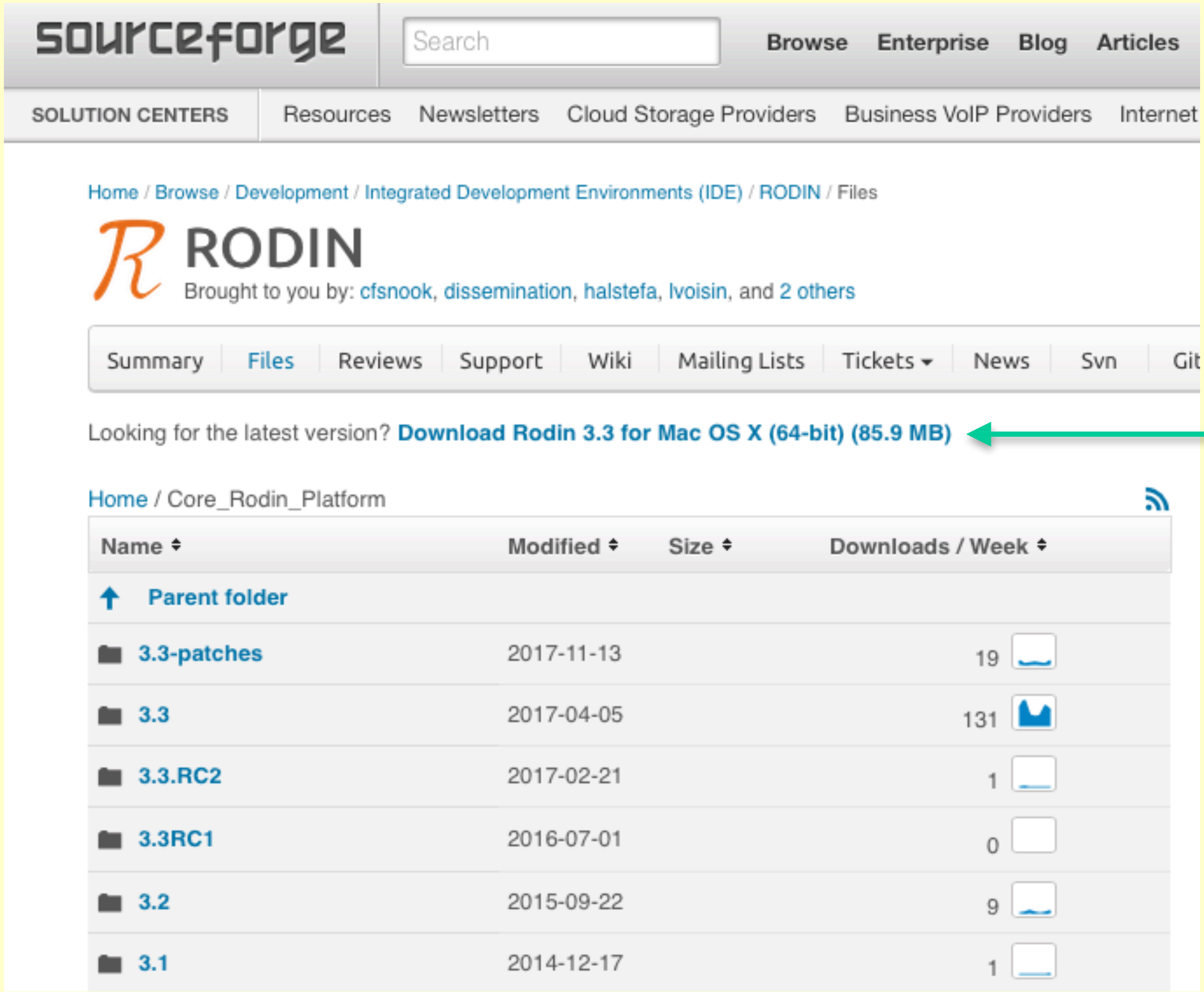


Rodin Platform and Plug-in Installation

Name	Installation
Rodin platform	- Requires Java 1.6 - Download the **Core: Rodin Platform file** for your platform. To install, just unpack the archive anywhere on your hard-disk and launch the "rodin" executable in it. - Start Rodin - **Information on the latest release.**
Plug-ins	- Plug-ins are installed from within Rodin by selecting Help/Install New Software. Then select the appropriate update site from the list of download sites. - **Details on plug-ins.** - Install the Atelier B Provers plugin from the Atelier B Provers Update site to take full advantage of Rodin proof capabilities - Install the **ProB** plugin from the ProB Update site for powerful model checking and animation
User manual and Tutorial	- **Rodin Handbook**

Install the Event-B RODIN tool

https://sourceforge.net/projects/rodin-b-sharp/files/Core_Rodin_Platform/



Home / Browse / Development / Integrated Development Environments (IDE) / RODIN / Files

RODIN

Brought to you by: cfsnook, dissemination, halstefa, lvoisin, and 2 others

Looking for the latest version? **Download Rodin 3.3 for Mac OS X (64-bit) (85.9 MB)**

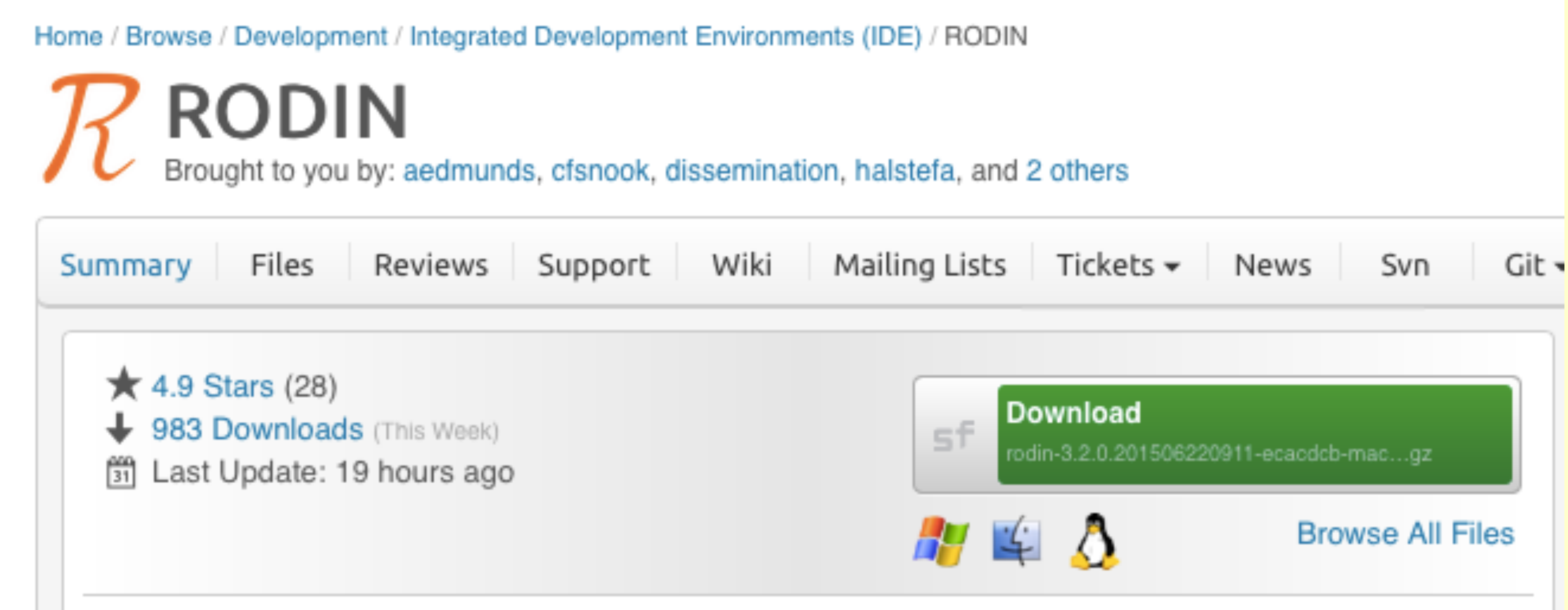
Home / Core_Rodin_Platform

Name	Modified	Size	Downloads / Week
Parent folder			
3.3-patches	2017-11-13		19
3.3	2017-04-05		131
3.3.RC2	2017-02-21		1
3.3RC1	2016-07-01		0
3.2	2015-09-22		9
3.1	2014-12-17		1

About 90MB

Install the Event-B RODIN tool

https://sourceforge.net/projects/rodin-b-sharp/



Home / Browse / Development / Integrated Development Environments (IDE) / RODIN

RODIN

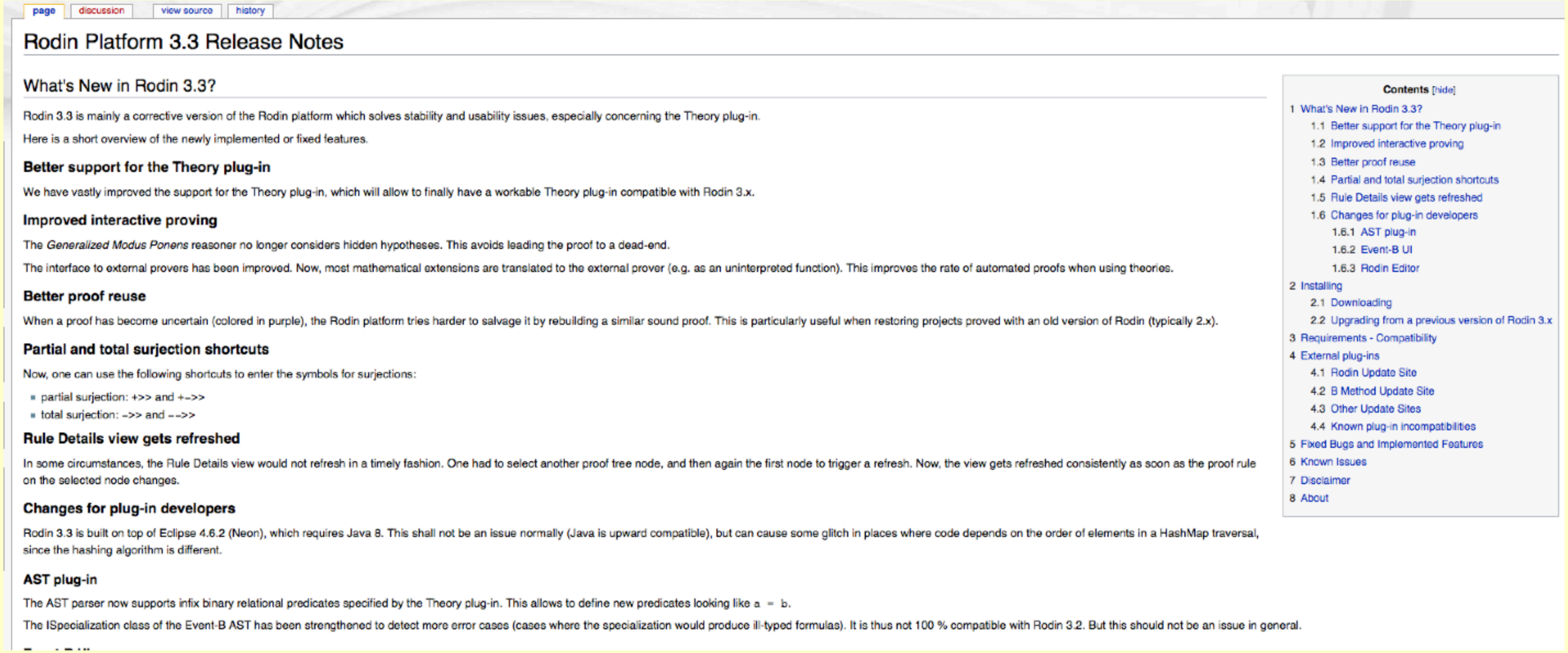
Brought to you by: aedmunds, cfsnook, dissemination, halstefa, and 2 others

★ 4.9 Stars (28)

983 Downloads (This Week)

Last Update: 19 hours ago

http://wiki.event-b.org/index.php/Rodin_Platform_3.3_Release_Notes



Rodin Platform 3.3 Release Notes

What's New in Rodin 3.3?

Rodin 3.3 is mainly a corrective version of the Rodin platform which solves stability and usability issues, especially concerning the Theory plug-in.

Here is a short overview of the newly implemented or fixed features.

Better support for the Theory plug-in

We have vastly improved the support for the Theory plug-in, which will allow to finally have a workable Theory plug-in compatible with Rodin 3.x.

Improved interactive proving

The *Generalized Modus Ponens* reasoner no longer considers hidden hypotheses. This avoids leading the proof to a dead-end.

The interface to external provers has been improved. Now, most mathematical extensions are translated to the external prover (e.g. as an uninterpreted function). This improves the rate of automated proofs when using theories.

Better proof reuse

When a proof has become uncertain (colored in purple), the Rodin platform tries harder to salvage it by rebuilding a similar sound proof. This is particularly useful when restoring projects proved with an old version of Rodin (typically 2.x).

Partial and total surjection shortcuts

Now, one can use the following shortcuts to enter the symbols for surjections:

- partial surjection: +>> and +->>
- total surjection: ->> and -->>

Rule Details view gets refreshed

In some circumstances, the Rule Details view would not refresh in a timely fashion. One had to select another proof tree node, and then again the first node to trigger a refresh. Now, the view gets refreshed consistently as soon as the proof rule on the selected node changes.

Changes for plug-in developers

Rodin 3.3 is built on top of Eclipse 4.6.2 (Neon), which requires Java 8. This shall not be an issue normally (Java is upward compatible), but can cause some glitch in places where code depends on the order of elements in a HashMap traversal, since the hashing algorithm is different.

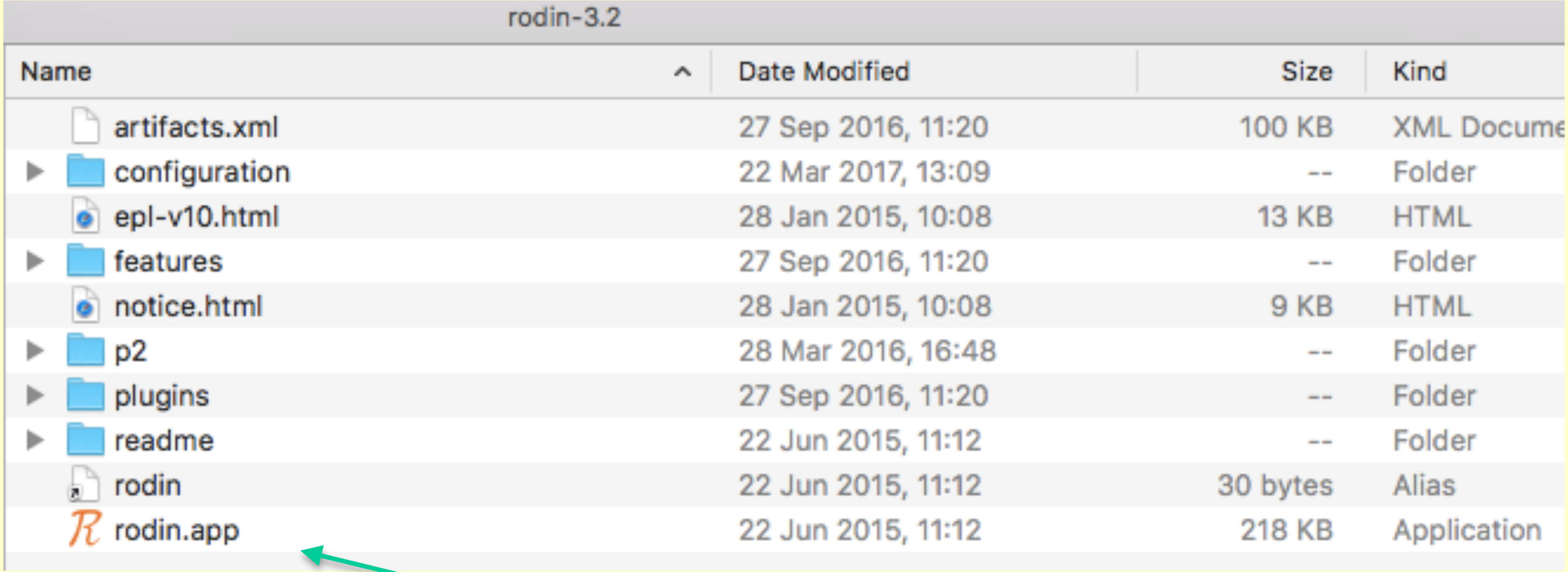
AST plug-in

The AST parser now supports infix binary relational predicates specified by the Theory plug-in. This allows to define new predicates looking like a = b.

The ISpecialization class of the Event-B AST has been strengthened to detect more error cases (cases where the specialization would produce ill-typed formulas). It is thus not 100 % compatible with Rodin 3.2. But this should not be an issue in general.

Contents [hide]

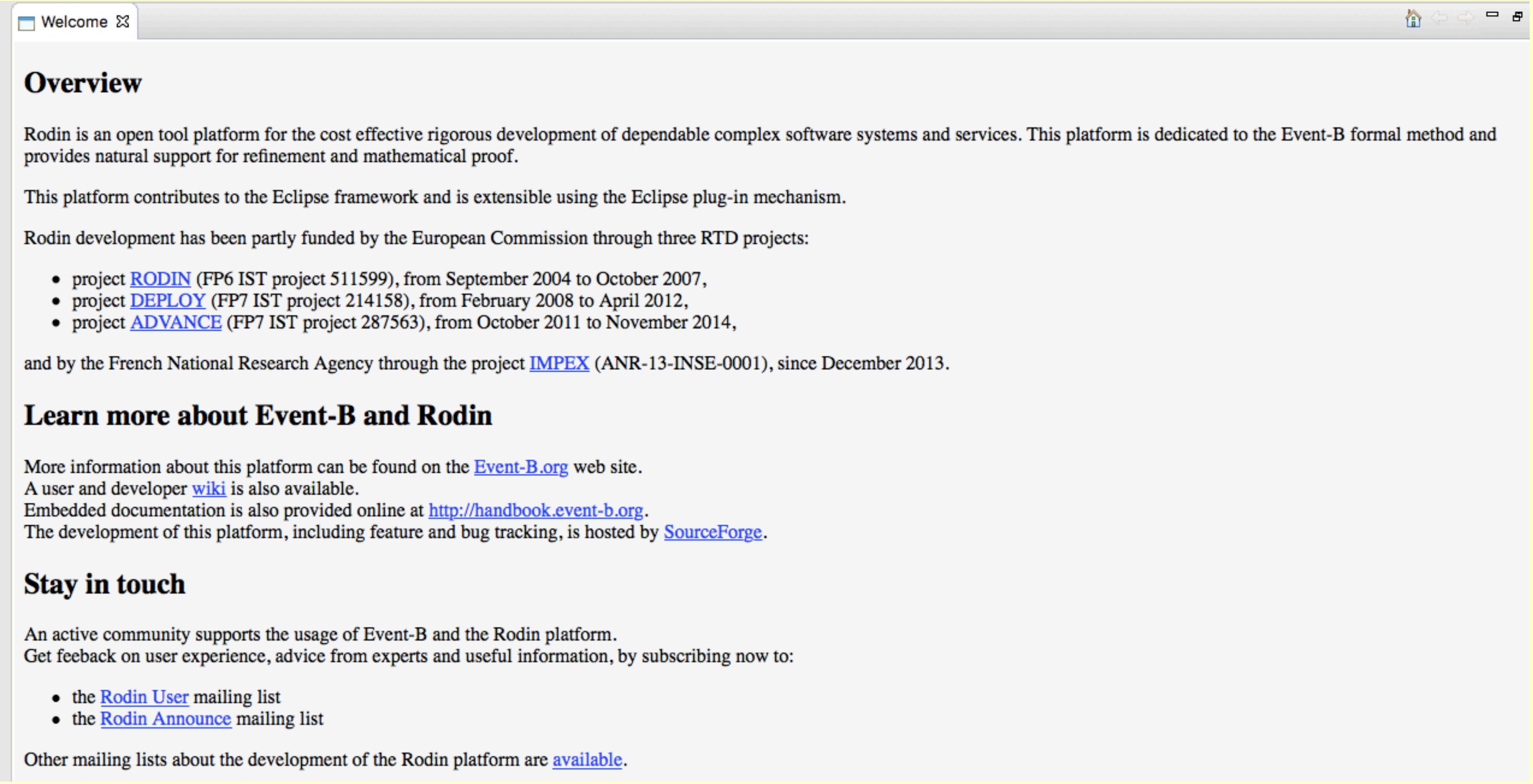
The archive file



rodin-3.2

Name	Date Modified	Size	Kind
artifacts.xml	27 Sep 2016, 11:20	100 KB	XML Docume
configuration	22 Mar 2017, 13:09	--	Folder
epl-v10.html	28 Jan 2015, 10:08	13 KB	HTML
features	27 Sep 2016, 11:20	--	Folder
notice.html	28 Jan 2015, 10:08	9 KB	HTML
p2	28 Mar 2016, 16:48	--	Folder
plugins	27 Sep 2016, 11:20	--	Folder
readme	22 Jun 2015, 11:12	--	Folder
rodin	22 Jun 2015, 11:12	30 bytes	Alias
rodin.app	22 Jun 2015, 11:12	218 KB	Application

Execute the application



Overview

Rodin is an open tool platform for the cost effective rigorous development of dependable complex software systems and services. This platform is dedicated to the Event-B formal method and provides natural support for refinement and mathematical proof.

This platform contributes to the Eclipse framework and is extensible using the Eclipse plug-in mechanism.

Rodin development has been partly funded by the European Commission through three RTD projects:

- project RODIN (FP6 IST project 511599), from September 2004 to October 2007,
- project DEPLOY (FP7 IST project 214158), from February 2008 to April 2012,
- project ADVANCE (FP7 IST project 287563), from October 2011 to November 2014,

and by the French National Research Agency through the project IMPEX (ANR-13-INSE-0001), since December 2013.

Learn more about Event-B and Rodin

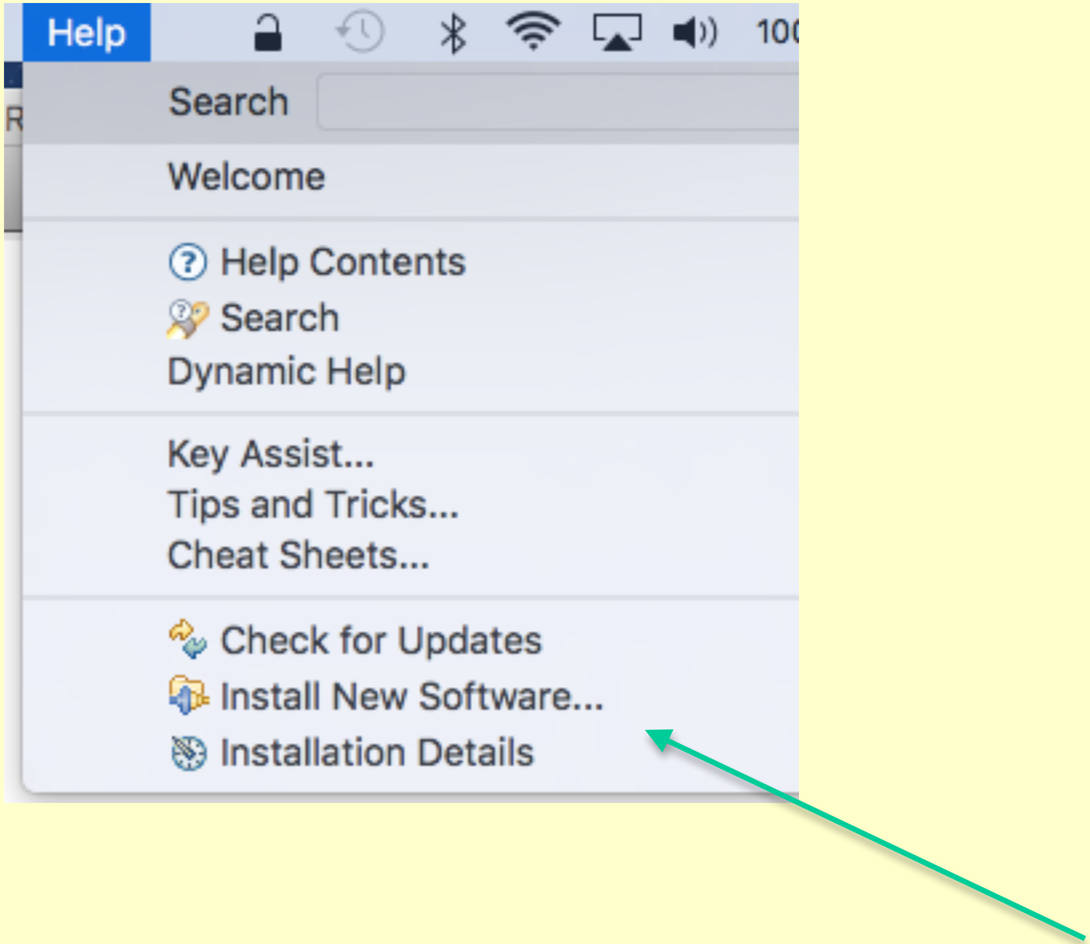
More information about this platform can be found on the Event-B.org web site.
A user and developer wiki is also available.
Embedded documentation is also provided online at http://handbook.event-b.org.
The development of this platform, including feature and bug tracking, is hosted by SourceForge.

Stay in touch

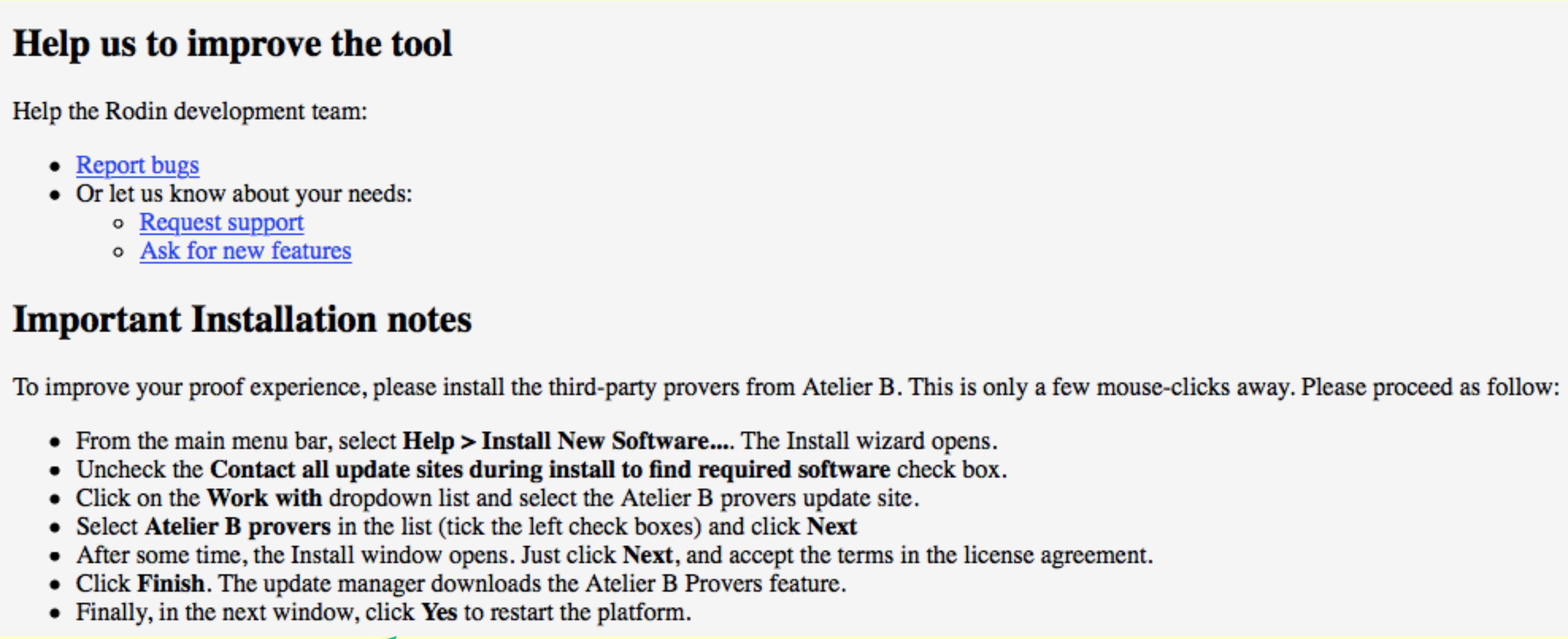
An active community supports the usage of Event-B and the Rodin platform.
Get feeback on user experience, advice from experts and useful information, by subscribing now to:

- the Rodin User mailing list
- the Rodin Announce mailing list

Other mailing lists about the development of the Rodin platform are available.



Important to update



Help us to improve the tool

Help the Rodin development team:

- Report bugs
- Or let us know about your needs:
  - Request support
  - Ask for new features

Important Installation notes

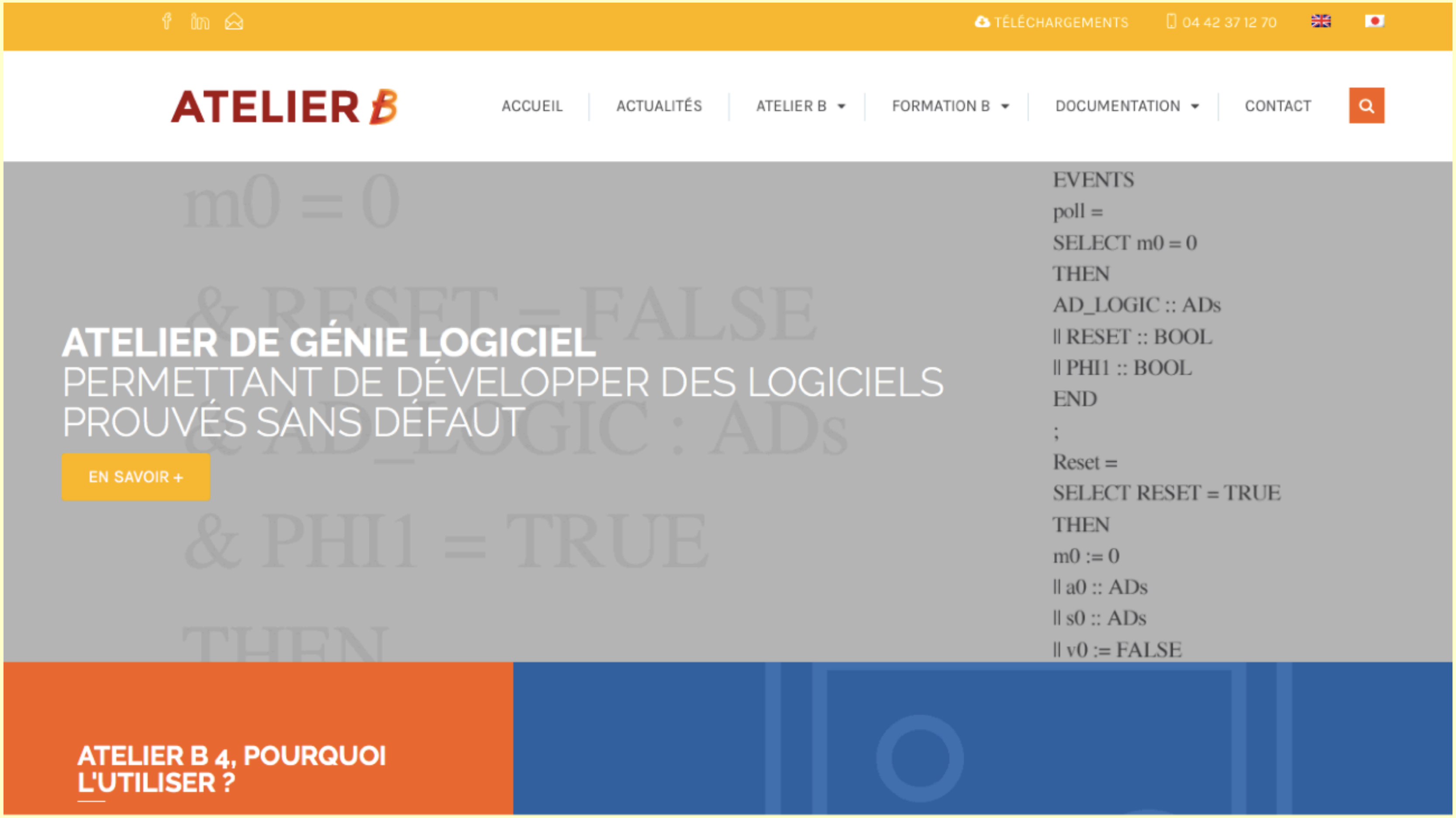
To improve your proof experience, please install the third-party provers from Atelier B. This is only a few mouse-clicks away. Please proceed as follow:

- From the main menu bar, select **Help > Install New Software...**. The Install wizard opens.
- Uncheck the **Contact all update sites during install to find required software** check box.
- Click on the **Work with** dropdown list and select the Atelier B provers update site.
- Select **Atelier B provers** in the list (tick the left check boxes) and click **Next**
- After some time, the Install window opens. Just click **Next**, and accept the terms in the license agreement.
- Click **Finish**. The update manager downloads the Atelier B Provers feature.
- Finally, in the next window, click **Yes** to restart the platform.

Check for updates … then install the provers

You will need to restart the RODIN platform

http://www.atelierb.eu

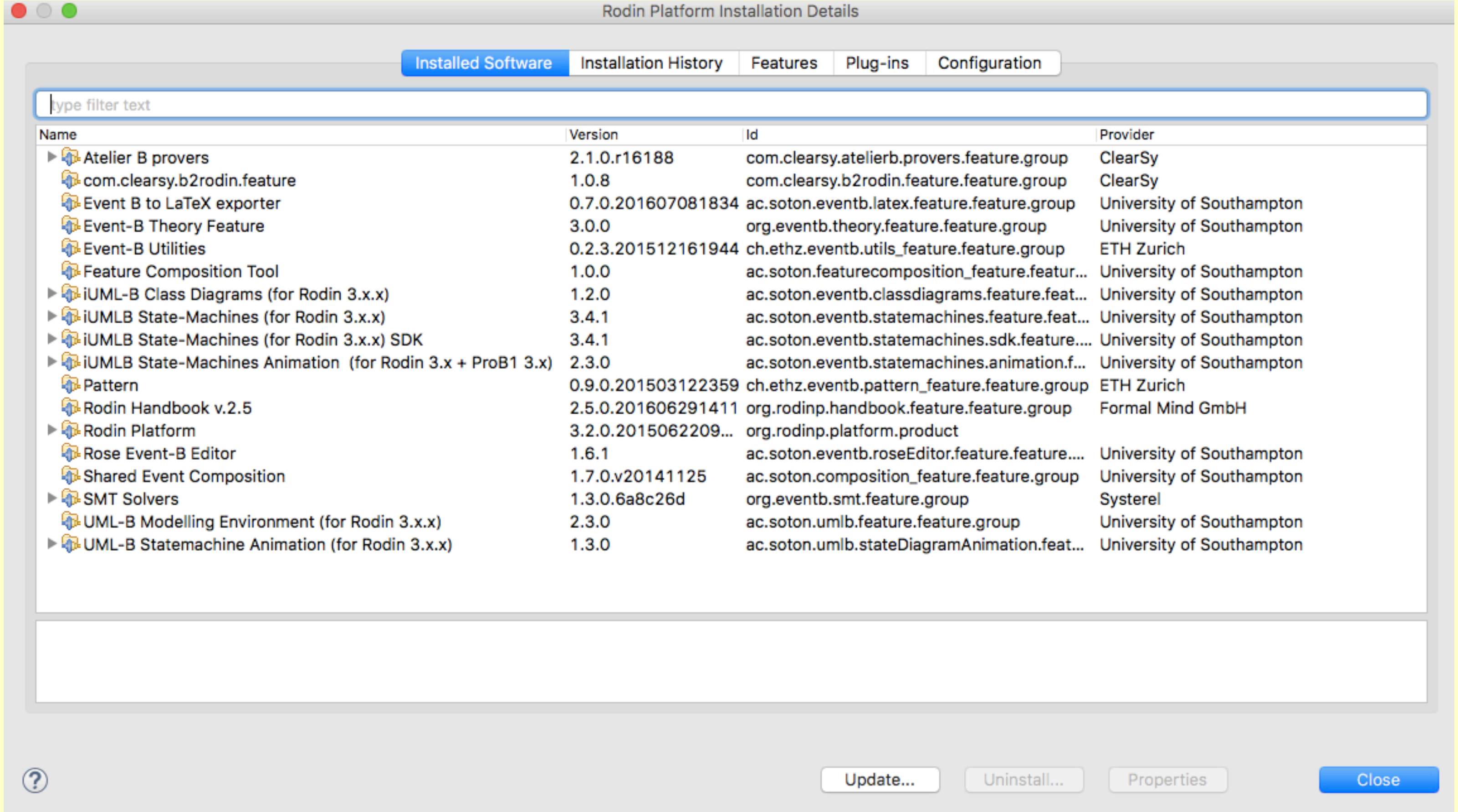


ATELIER DE GÉNIE LOGICIEL
PERMETTANT DE DÉVELOPPER DES LOGICIELS
PROUVÉS SANS DÉFAUT

```
EVENTS
poll =
SELECT m0 = 0
THEN
AD_LOGIC :: ADs
|| RESET :: BOOL
|| PHI1 :: BOOL
END
;
Reset =
SELECT RESET = TRUE
THEN
m0 := 0
|| a0 :: ADs
|| s0 :: ADs
|| v0 := FALSE
```

ATELIER B 4, POURQUOI L'UTILISER ?

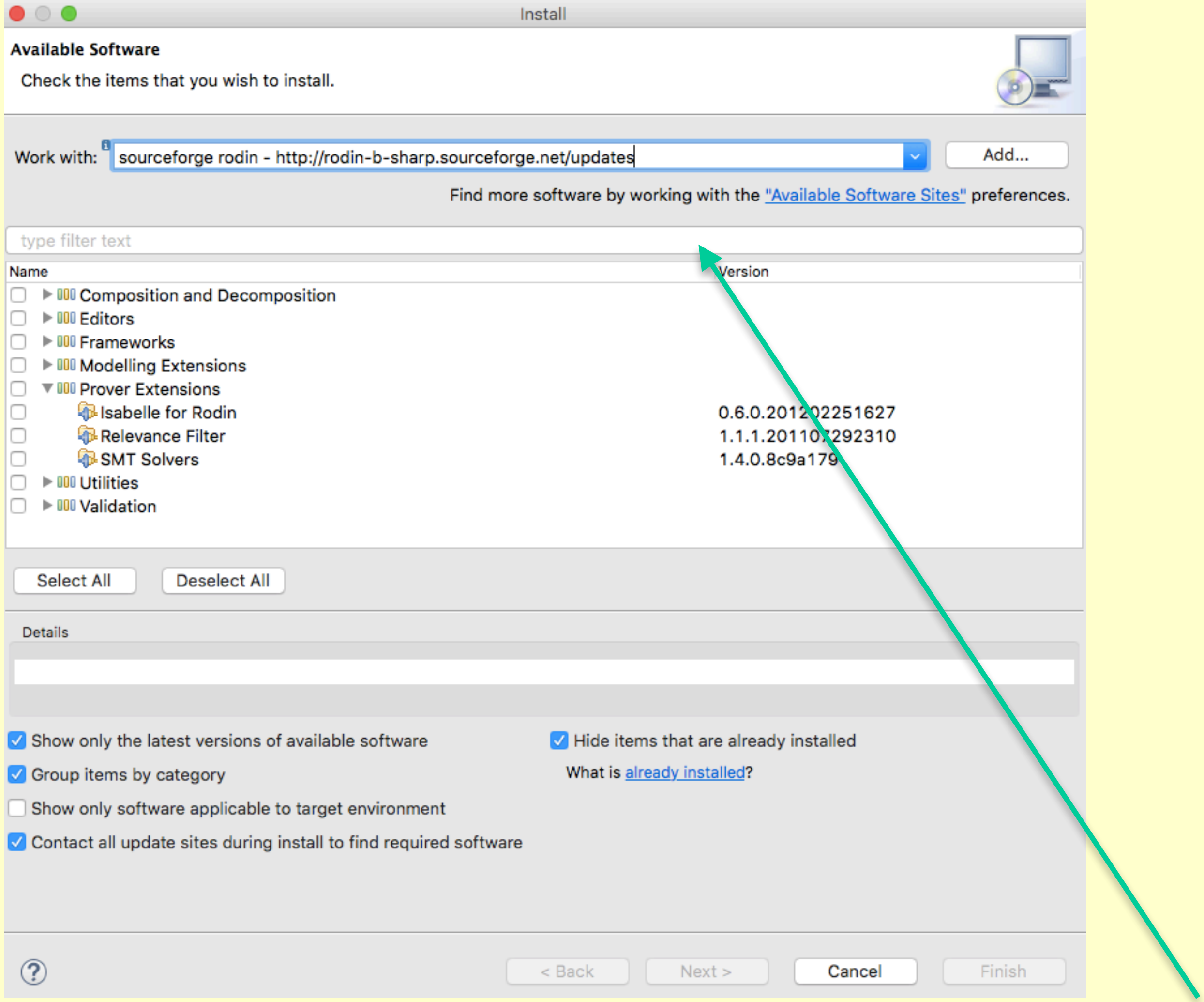
The provers come from here (do you recognise it?)



Name	Version	Id	Provider
▶ Atelier B provers	2.1.0.r16188	com.clearsy.atelierb.provers.feature.group	ClearSy
com.clearsy.b2rodin.feature	1.0.8	com.clearsy.b2rodin.feature.feature.group	ClearSy
Event B to LaTeX exporter	0.7.0.201607081834	ac.soton.eventb.latex.feature.feature.group	University of Southampton
Event-B Theory Feature	3.0.0	org.eventb.theory.feature.feature.group	University of Southampton
Event-B Utilities	0.2.3.201512161944	ch.ethz.eventb.utils_feature.feature.group	ETH Zurich
Feature Composition Tool	1.0.0	ac.soton.featurecomposition_feature.featur...	University of Southampton
▶ iUML-B Class Diagrams (for Rodin 3.x.x)	1.2.0	ac.soton.eventb.classdiagrams.feature.feat...	University of Southampton
▶ iUMLB State-Machines (for Rodin 3.x.x)	3.4.1	ac.soton.eventb.statemachines.feature.feat...	University of Southampton
▶ iUMLB State-Machines (for Rodin 3.x.x) SDK	3.4.1	ac.soton.eventb.statemachines.sdk.feature....	University of Southampton
▶ iUMLB State-Machines Animation (for Rodin 3.x + ProB1 3.x)	2.3.0	ac.soton.eventb.statemachines.animation.f...	University of Southampton
Pattern	0.9.0.201503122359	ch.ethz.eventb.pattern_feature.feature.group	ETH Zurich
Rodin Handbook v.2.5	2.5.0.201606291411	org.rodinp.handbook.feature.feature.group	Formal Mind GmbH
▶ Rodin Platform	3.2.0.2015062209...	org.rodinp.platform.product	
Rose Event-B Editor	1.6.1	ac.soton.eventb.roseEditor.feature.feature....	University of Southampton
Shared Event Composition	1.7.0.v20141125	ac.soton.composition_feature.feature.group	University of Southampton
▶ SMT Solvers	1.3.0.6a8c26d	org.eventb.smt.feature.group	Systerel
UML-B Modelling Environment (for Rodin 3.x.x)	2.3.0	ac.soton.umlb.feature.feature.group	University of Southampton
▶ UML-B Statemachine Animation (for Rodin 3.x.x)	1.3.0	ac.soton.umlb.stateDiagramAnimation.feat...	University of Southampton

You don’t need all these, just the Provers (and SMT solvers, perhaps?)

(The SMT solvers make the automated proof system more powerful)



Install

Available Software

Check the items that you wish to install.

Work with: sourceforge rodin - http://rodin-b-sharp.sourceforge.net/updates

Find more software by working with the "Available Software Sites" preferences.

Name	Version
□ ▶ Composition and Decomposition	
□ ▶ Editors	
□ ▶ Frameworks	
□ ▶ Modelling Extensions	
□ ▼ Prover Extensions	
□ Isabelle for Rodin	0.6.0.201202251627
□ Relevance Filter	1.1.1.201107292310
□ SMT Solvers	1.4.0.8c9a179
□ ▶ Utilities	
□ ▶ Validation	

Details

☑ Show only the latest versions of available software

☑ Group items by category

□ Show only software applicable to target environment

☑ Contact all update sites during install to find required software

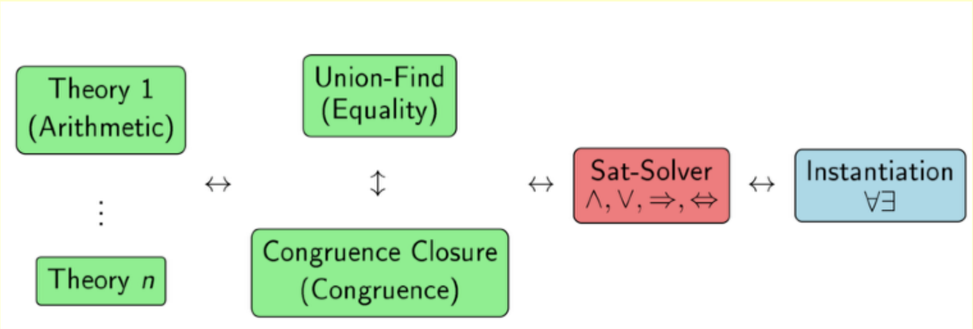
☑ Hide items that are already installed

What is already installed?

To install SMT solvers add the site http://rodin-b-sharp.sourceforge.net/updates to the site updates list

The SMT solvers make the automated proof system more powerful

An SMT instance is a generalisation of a Boolean SAT instance in which various sets of variables are replaced by predicates from a variety of underlying theories



Theory 1 (Arithmetic)

⋮

Theory n

↔

Union-Find (Equality)

↕

Congruence Closure (Congruence)

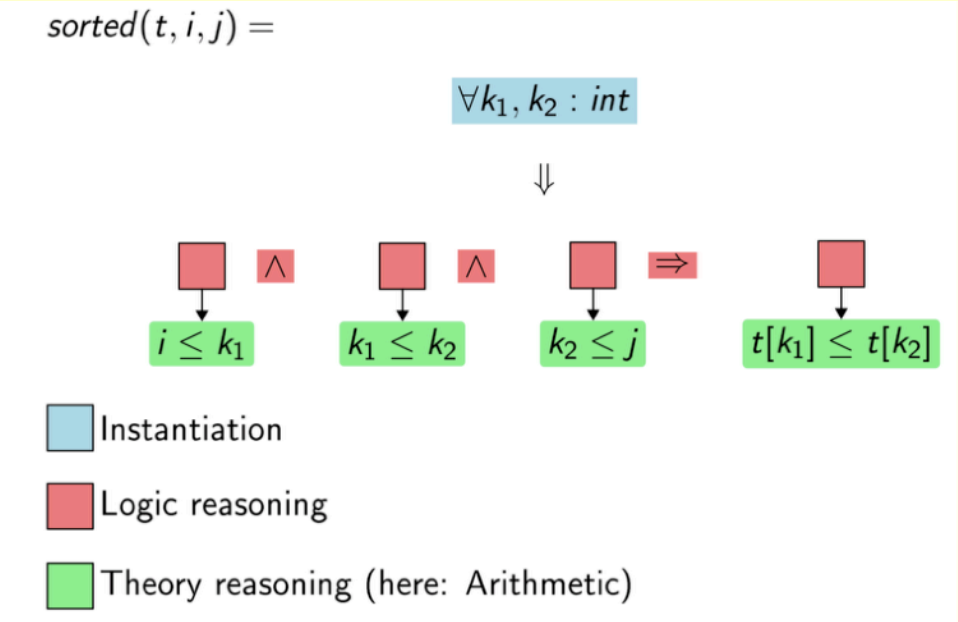
↔

Sat-Solver $\wedge, \vee, \Rightarrow, \Leftrightarrow$

↔

Instantiation $\forall\exists$

Example : prove list is sorted



$sorted(t, i, j) =$

$\forall k_1, k_2 : int$

$\Downarrow$

$i \leq k_1 \quad \wedge \quad k_1 \leq k_2 \quad \wedge \quad k_2 \leq j \quad \Rightarrow \quad t[k_1] \leq t[k_2]$

🟦 Instantiation

🟥 Logic reasoning

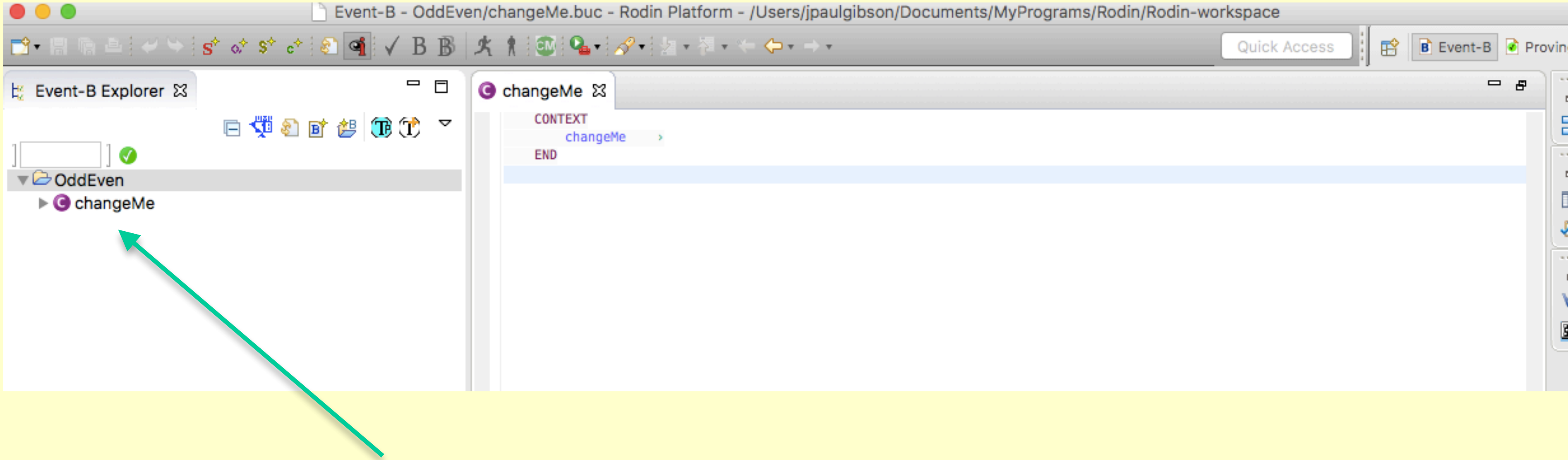
🟩 Theory reasoning (here: Arithmetic)

Odd Even Problem Preparation

1. Make a new Event-B project

2. Make a new Event-B component (context) called OddEven

3. This is for the problem you are going to try to solve



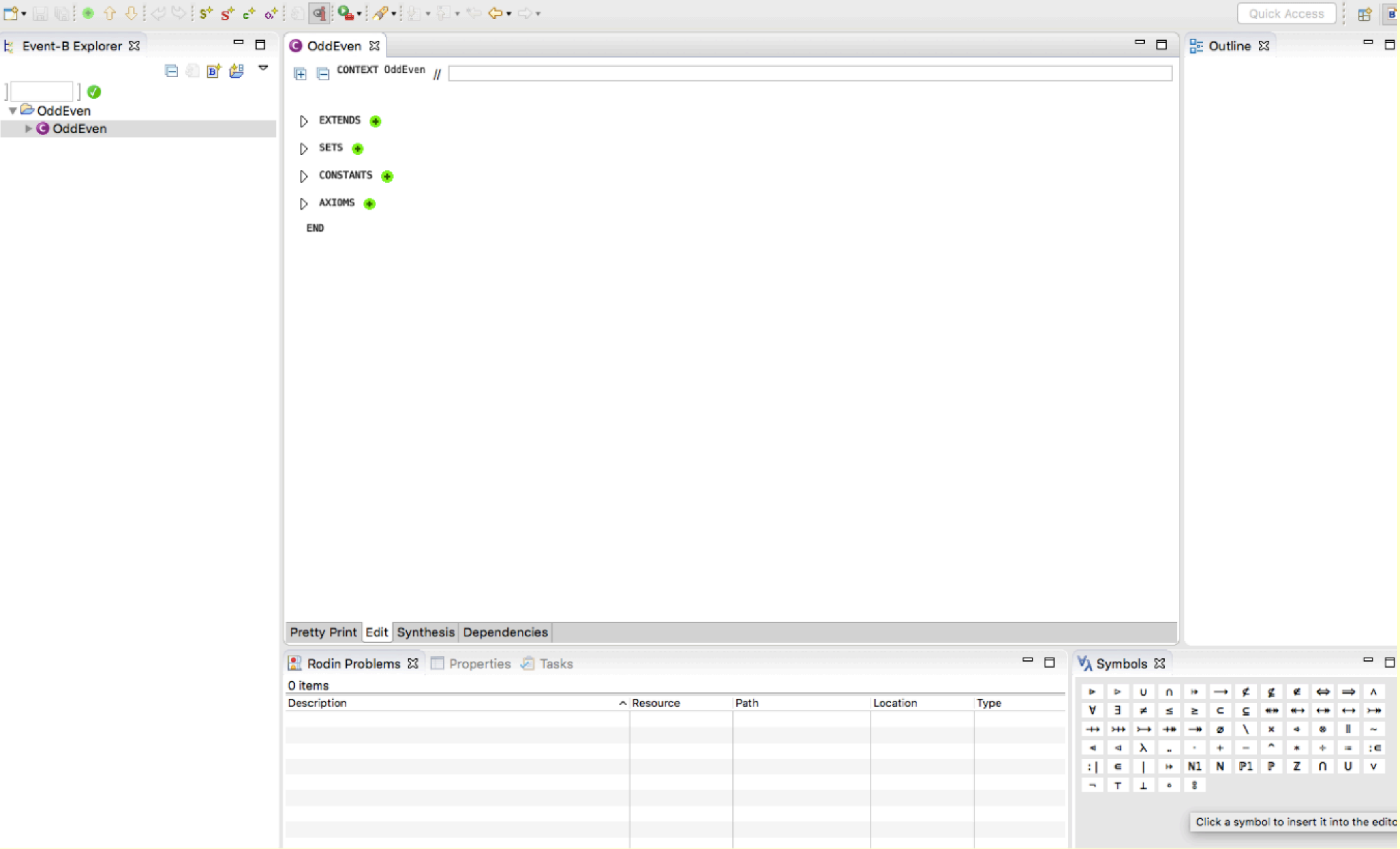
Event-B - OddEven/changeMe.buc - Rodin Platform - /Users/jpaulgibson/Documents/MyPrograms/Rodin/Rodin-workspace

```
CONTEXT
    changeMe
END
```

Open with (right click) Event-B context editor

Odd Even Problem Preparation

Editing a context



```
CONTEXT OddEven //

EXTENDS
SETS
CONSTANTS
AXIOMS
END
```

0 items

Description	Resource	Path	Location	Type

Click a symbol to insert it into the edito

Simple proofs on properties of odd and even numbers

Express as theorems and use RODIN to prove:

thm1. The addition of two even numbers is even

thm2. The difference between two odd numbers is even

thm3. The multiplication of an even number with an odd number is eve

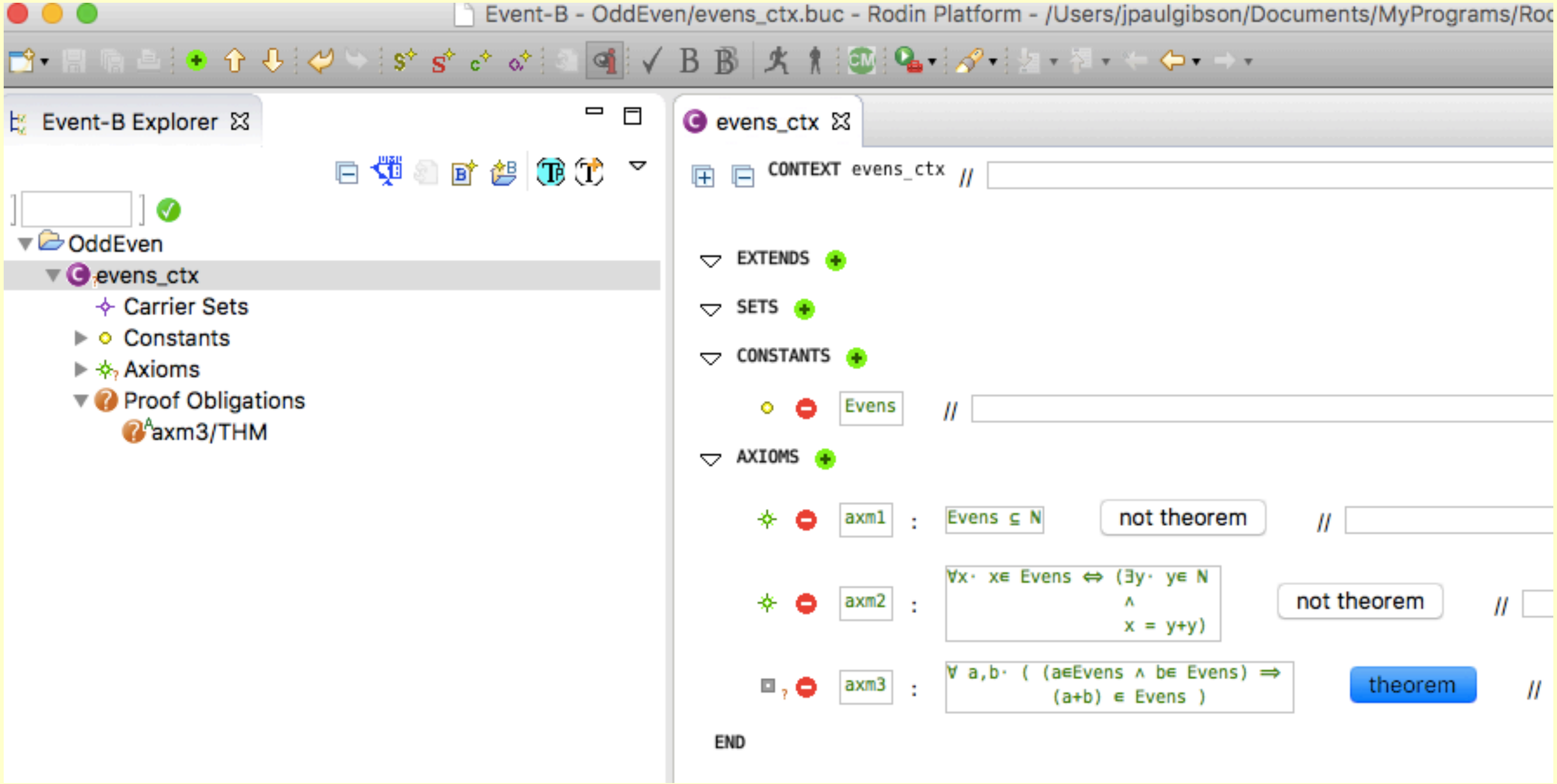
thm4. The multiplication of two odd numbers is odd

TODO: Complete the proofs by hand before trying to prove them with the tool.

Question: how many different ways can you express/specify these properties / theorems

Which specifications make the proof easier?

1. The addition of two even numbers is even


Event-B - OddEven/evens_ctx.buc - Rodin Platform - /Users/jpaulgibson/Documents/MyPrograms/Roc
Event-B Explorer
OddEven
evens_ctx
Carrier Sets
Constants
Axioms
Proof Obligations
axm3/THM
evens_ctx

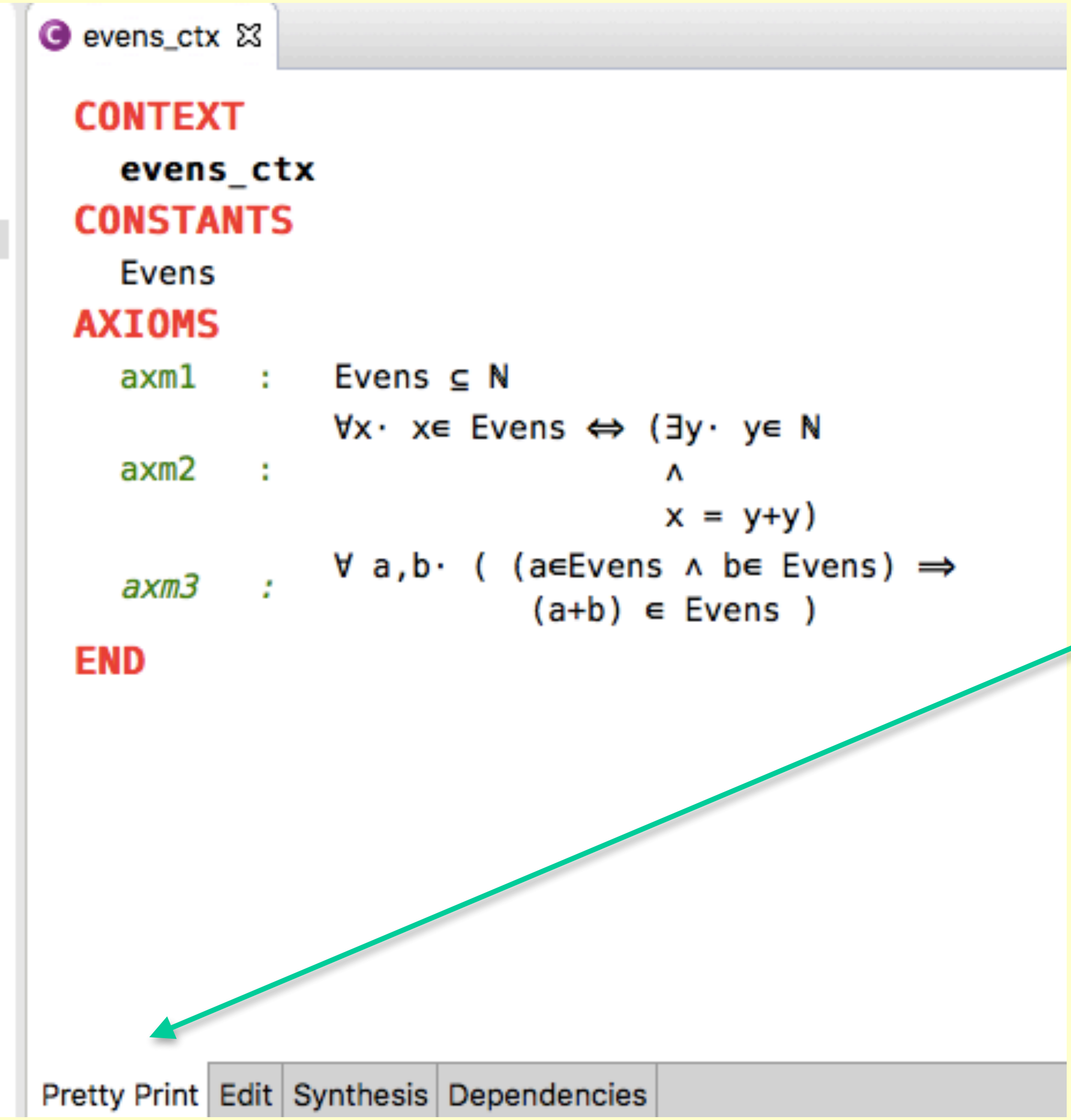
```
CONTEXT evens_ctx //

EXTENDS
SETS
CONSTANTS
    Evens //
AXIOMS
    axm1 : Evens ⊆ ℕ                          not theorem //
    axm2 : ∀x· x∈ Evens ⇔ (∃y· y∈ ℕ
                                 ∧
                                 x = y+y)     not theorem //
    axm3 : ∀ a,b· ( (a∈Evens ∧ b∈ Evens) ⇒
                   (a+b) ∈ Evens )            theorem //
END
```

TODO - edit the context specification

Question: what do you think of the editor/interface/usability?

1. The addition of two even numbers is even

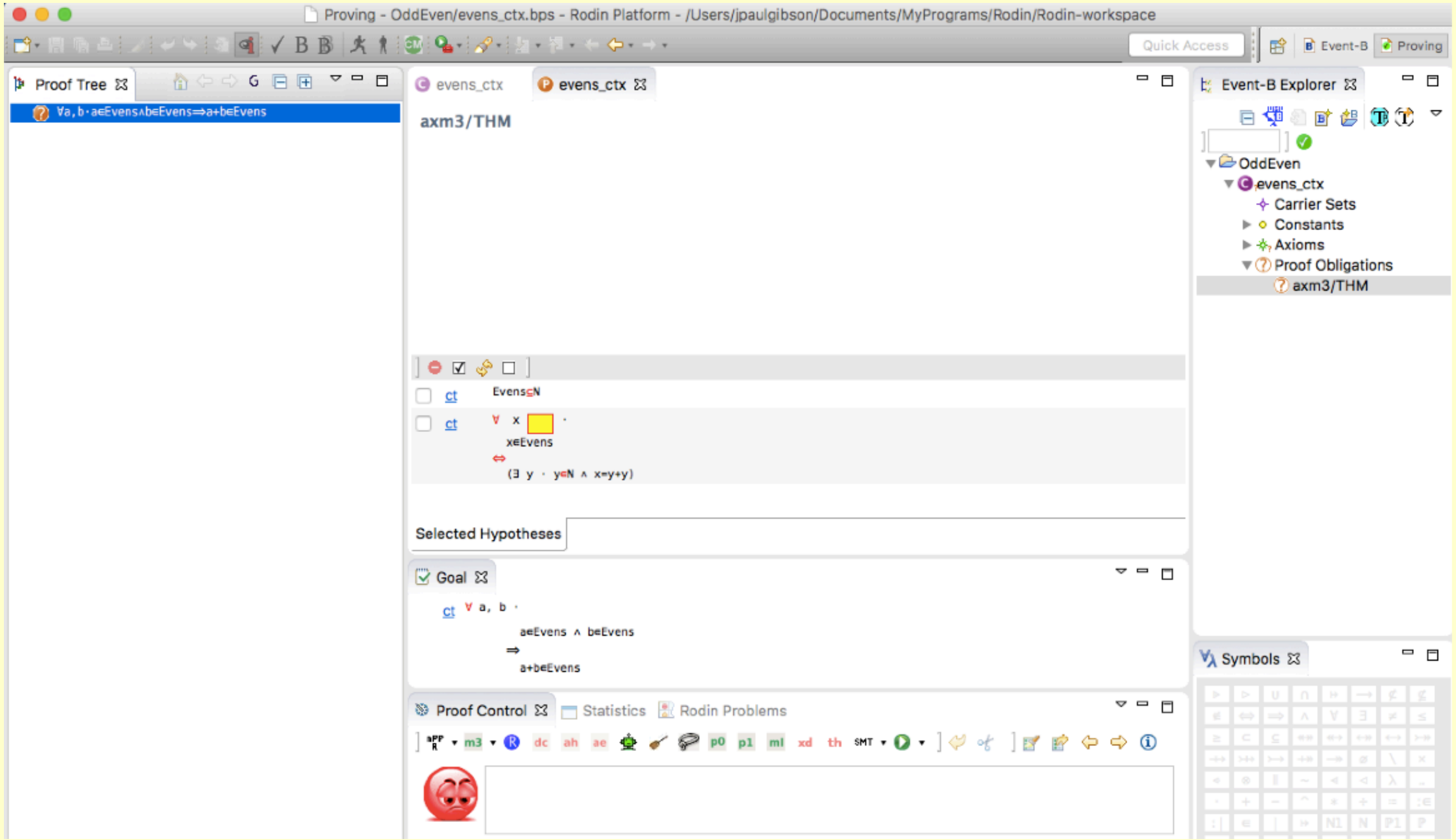


```
CONTEXT
   evens_ctx
CONSTANTS
   Evens
AXIOMS
   axm1    :    Evens ⊆ ℕ
                ∀x· x∈ Evens ⇔ (∃y· y∈ ℕ
   axm2    :                     ∧
                                 x = y+y)
                ∀ a,b· ( (a∈Evens ∧ b∈ Evens) ⇒
   axm3    :            (a+b) ∈ Evens )
END
```

The pretty print view

1. The addition of two even numbers is even

Window - perspective - open perspective - proving (and open proof tree view)



```
∀a,b·a∈Evens∧b∈Evens⇒a+b∈Evens
```

axm3/THM

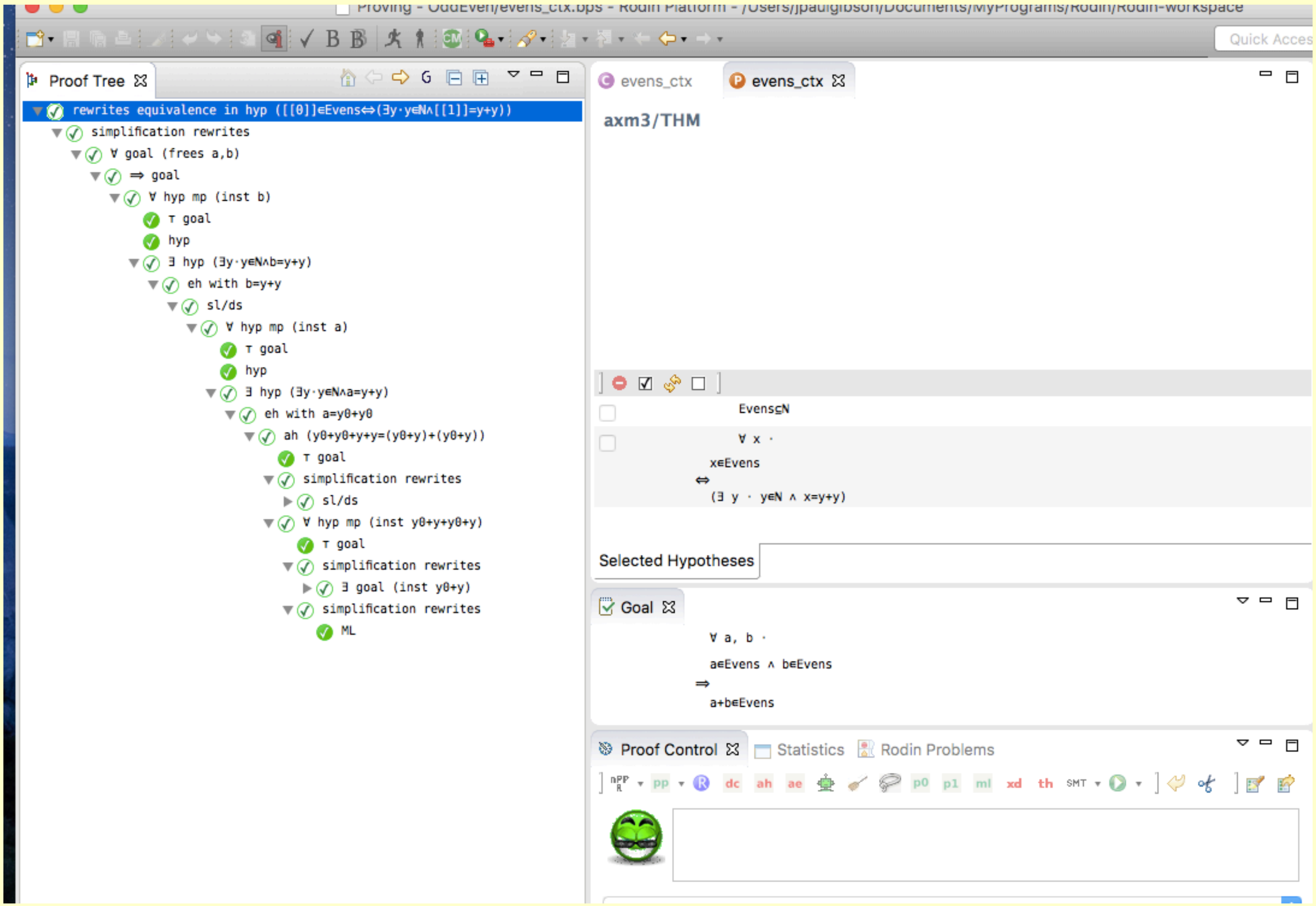
```
□ ct   Evens⊆ℕ
□ ct   ∀ x □ ·
         x∈Evens
       ⇔
         (∃ y · y∈ℕ ∧ x=y+y)
```

Goal

```
ct ∀ a, b ·
         a∈Evens ∧ b∈Evens
       ⇒
         a+b∈Evens
```

Question: can you prove the theorem?

1. The addition of two even numbers is even



Proving - OddEven/evens_ctx.bps - Rodin Platform - /Users/jpaulgibson/Documents/MyPrograms/Rodin/Rodin-workspace

Proof Tree

```
rewrites equivalence in hyp ([[0]]∈Evens⇔(∃y·y∈ℕ∧[[1]]=y+y))
  simplification rewrites
    ∀ goal (frees a,b)
      ⇒ goal
        ∀ hyp mp (inst b)
          ⊤ goal
          hyp
          ∃ hyp (∃y·y∈ℕ∧b=y+y)
            eh with b=y+y
              sl/ds
                ∀ hyp mp (inst a)
                  ⊤ goal
                  hyp
                  ∃ hyp (∃y·y∈ℕ∧a=y+y)
                    eh with a=y0+y0
                      ah (y0+y0+y+y=(y0+y)+(y0+y))
                        ⊤ goal
                        simplification rewrites
                          sl/ds
                        ∀ hyp mp (inst y0+y+y0+y)
                          ⊤ goal
                          simplification rewrites
                            ∃ goal (inst y0+y)
                          simplification rewrites
                            ML
```

evens_ctx | evens_ctx

axm3/THM

```
Evens⊆ℕ

∀ x ·
   x∈Evens
⇔
   (∃ y · y∈ℕ ∧ x=y+y)
```

Selected Hypotheses

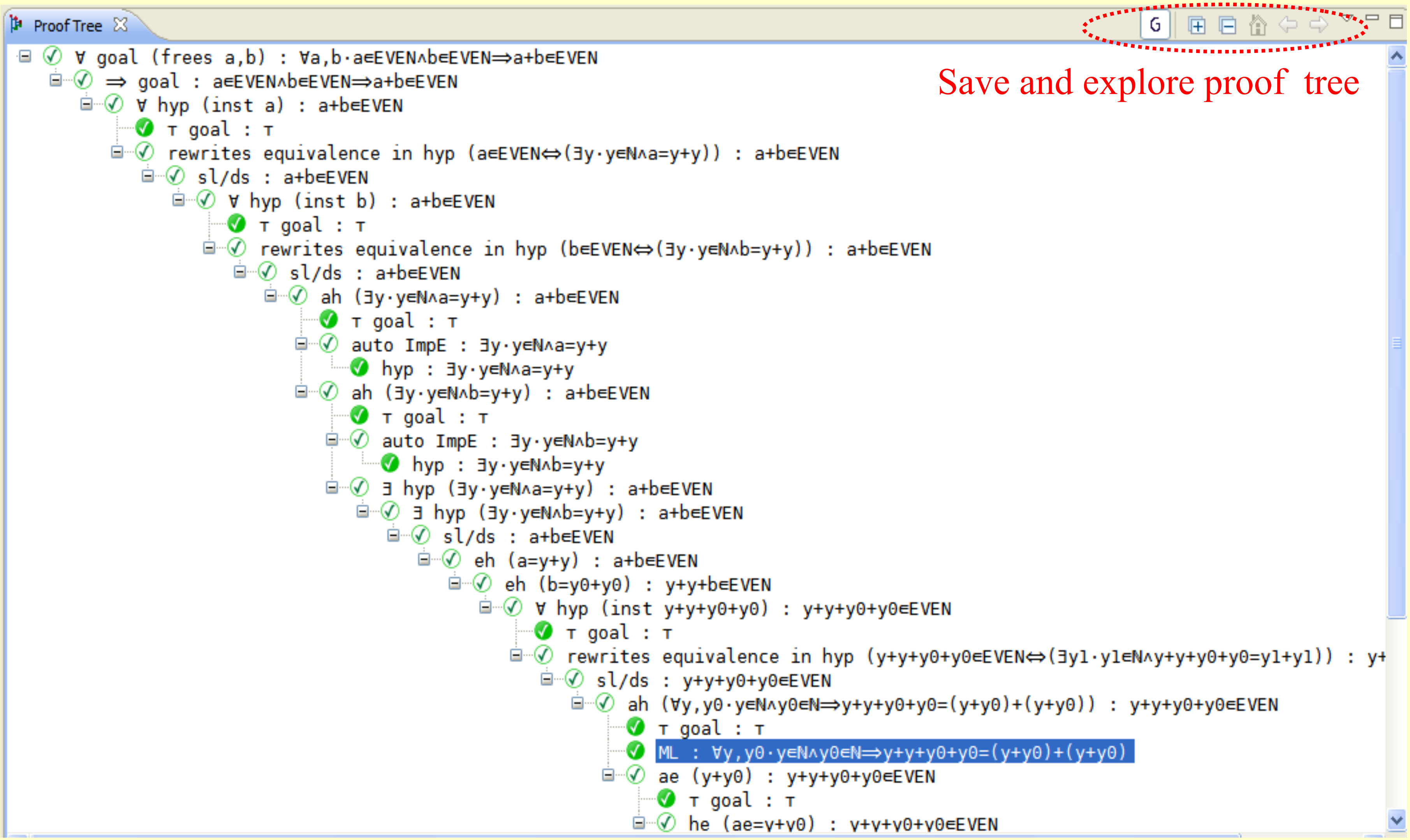
Goal

```
∀ a, b ·
   a∈Evens ∧ b∈Evens
⇒
   a+b∈Evens
```

Proof Control | Statistics | Rodin Problems

My proof - can you do better?

1. The addition of two even numbers is even

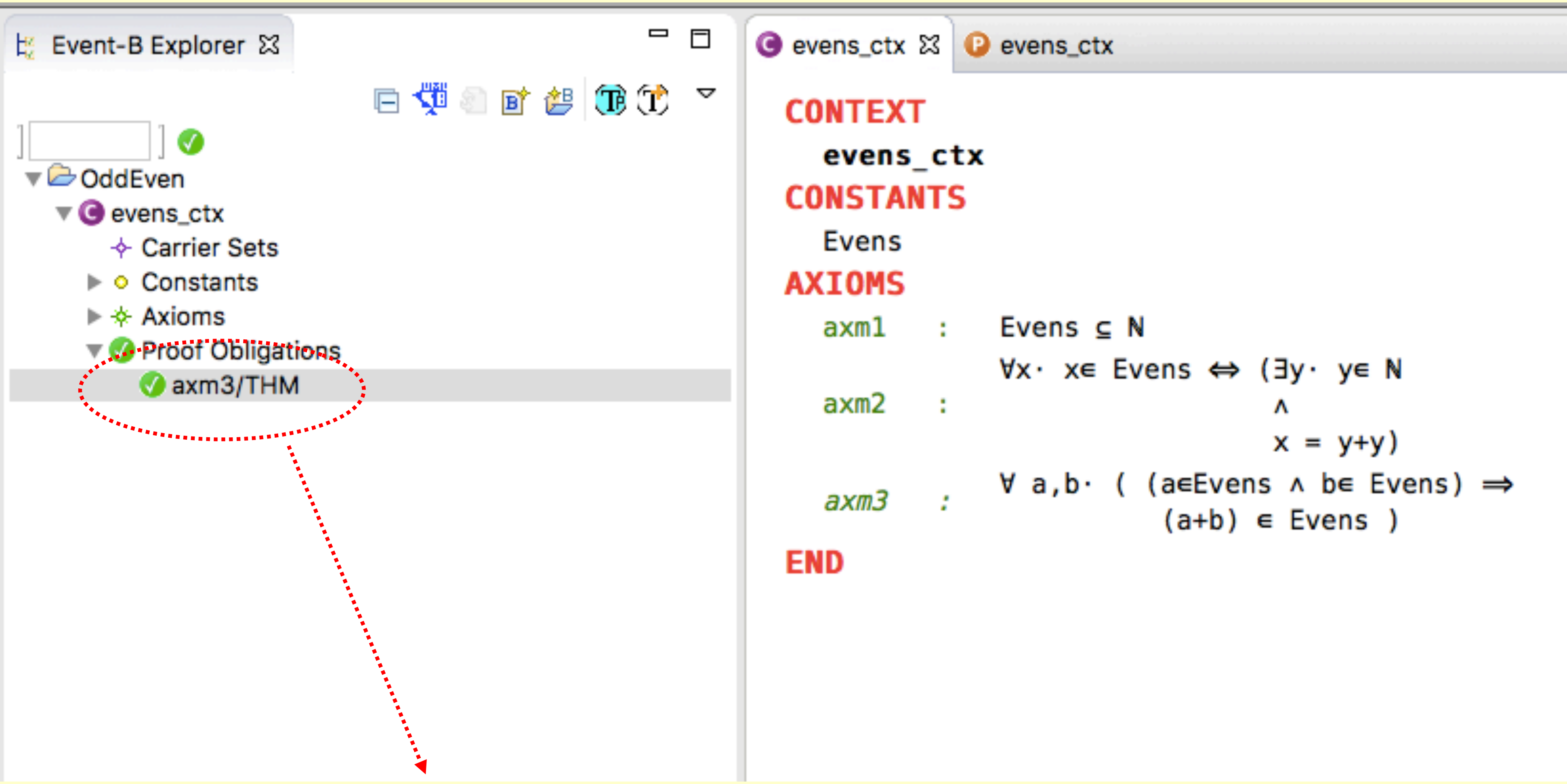


```
∀ goal (frees a,b) : ∀a,b·a∈EVEN∧b∈EVEN⇒a+b∈EVEN
  ⇒ goal : a∈EVEN∧b∈EVEN⇒a+b∈EVEN
    ∀ hyp (inst a) : a+b∈EVEN
      ⊤ goal : ⊤
      rewrites equivalence in hyp (a∈EVEN⇔(∃y·y∈ℕ∧a=y+y)) : a+b∈EVEN
        sl/ds : a+b∈EVEN
          ∀ hyp (inst b) : a+b∈EVEN
            ⊤ goal : ⊤
            rewrites equivalence in hyp (b∈EVEN⇔(∃y·y∈ℕ∧b=y+y)) : a+b∈EVEN
              sl/ds : a+b∈EVEN
                ah (∃y·y∈ℕ∧a=y+y) : a+b∈EVEN
                  ⊤ goal : ⊤
                  auto ImpE : ∃y·y∈ℕ∧a=y+y
                    hyp : ∃y·y∈ℕ∧a=y+y
                  ah (∃y·y∈ℕ∧b=y+y) : a+b∈EVEN
                    ⊤ goal : ⊤
                    auto ImpE : ∃y·y∈ℕ∧b=y+y
                      hyp : ∃y·y∈ℕ∧b=y+y
                    ∃ hyp (∃y·y∈ℕ∧a=y+y) : a+b∈EVEN
                      ∃ hyp (∃y·y∈ℕ∧b=y+y) : a+b∈EVEN
                        sl/ds : a+b∈EVEN
                          eh (a=y+y) : a+b∈EVEN
                            eh (b=y0+y0) : y+y+b∈EVEN
                              ∀ hyp (inst y+y+y0+y0) : y+y+y0+y0∈EVEN
                                ⊤ goal : ⊤
                                rewrites equivalence in hyp (y+y+y0+y0∈EVEN⇔(∃y1·y1∈ℕ∧y+y+y0+y0=y1+y1)) : y+
                                  sl/ds : y+y+y0+y0∈EVEN
                                    ah (∀y,y0·y∈ℕ∧y0∈ℕ⇒y+y+y0+y0=(y+y0)+(y+y0)) : y+y+y0+y0∈EVEN
                                      ⊤ goal : ⊤
                                      ML : ∀y,y0·y∈ℕ∧y0∈ℕ⇒y+y+y0+y0=(y+y0)+(y+y0)
                                      ae (y+y0) : y+y+y0+y0∈EVEN
                                        ⊤ goal : ⊤
                                        he (ae=y+y0) : y+y+y0+y0∈EVEN
```

Save and explore proof tree

This one looks a bit more complicated?

1. The addition of two even numbers is even

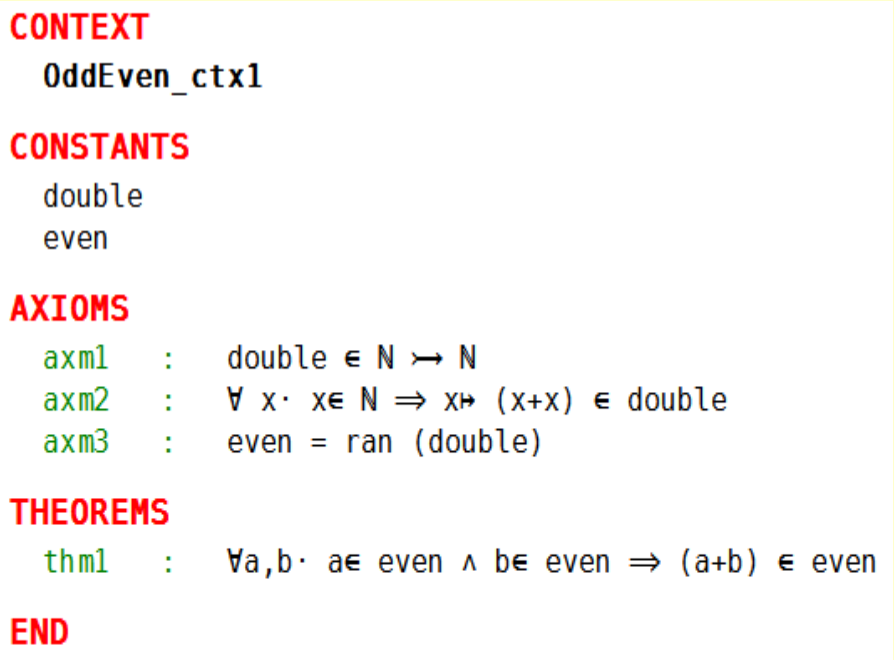


```
CONTEXT
   evens_ctx
CONSTANTS
   Evens
AXIOMS
   axm1   :   Evens ⊆ ℕ
              ∀x· x∈ Evens ⇔ (∃y· y∈ ℕ
   axm2   :                   ∧
                              x = y+y)
   axm3   :   ∀ a,b· ( (a∈Evens ∧ b∈ Evens) ⇒
                       (a+b) ∈ Evens )
END
```

The theorem has been marked as proved

1. The addition of two even numbers is even

What if we change specification of EVEN?



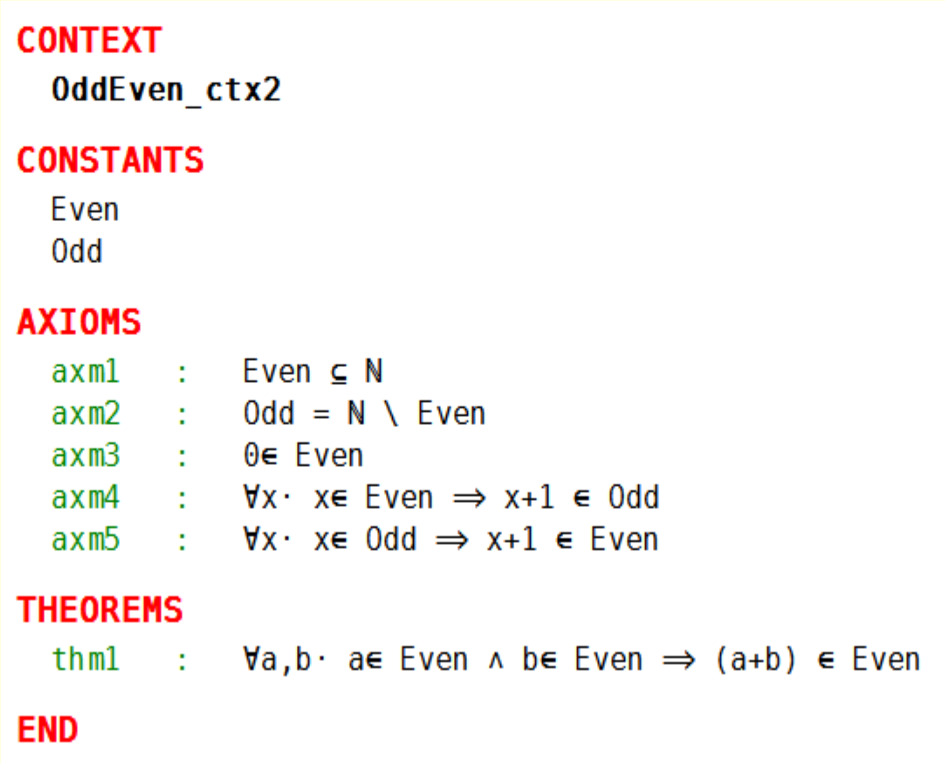
```
CONTEXT
  OddEven_ctx1

CONSTANTS
  double
  even

AXIOMS
  axm1   :   double ∈ ℕ ↣ ℕ
  axm2   :   ∀ x· x∈ ℕ ⇒ x↦ (x+x) ∈ double
  axm3   :   even = ran (double)

THEOREMS
  thm1   :   ∀a,b· a∈ even ∧ b∈ even ⇒ (a+b) ∈ even

END
```



```
CONTEXT
  OddEven_ctx2

CONSTANTS
  Even
  Odd

AXIOMS
  axm1   :   Even ⊆ ℕ
  axm2   :   Odd = ℕ \ Even
  axm3   :   0∈ Even
  axm4   :   ∀x· x∈ Even ⇒ x+1 ∈ Odd
  axm5   :   ∀x· x∈ Odd ⇒ x+1 ∈ Even

THEOREMS
  thm1   :   ∀a,b· a∈ Even ∧ b∈ Even ⇒ (a+b) ∈ Even

END
```

What if we change the theorem to prove?

Express as theorems and use RODIN to prove:

thm1. The addition of two even numbers is even

thm2. The difference between two odd numbers is even

thm3. The multiplication of an even number with an odd number is even

thm4. The multiplication of two odd numbers is odd

TODO : try these in RODIN

PROBLEM: write a short report on how you did it (or not)