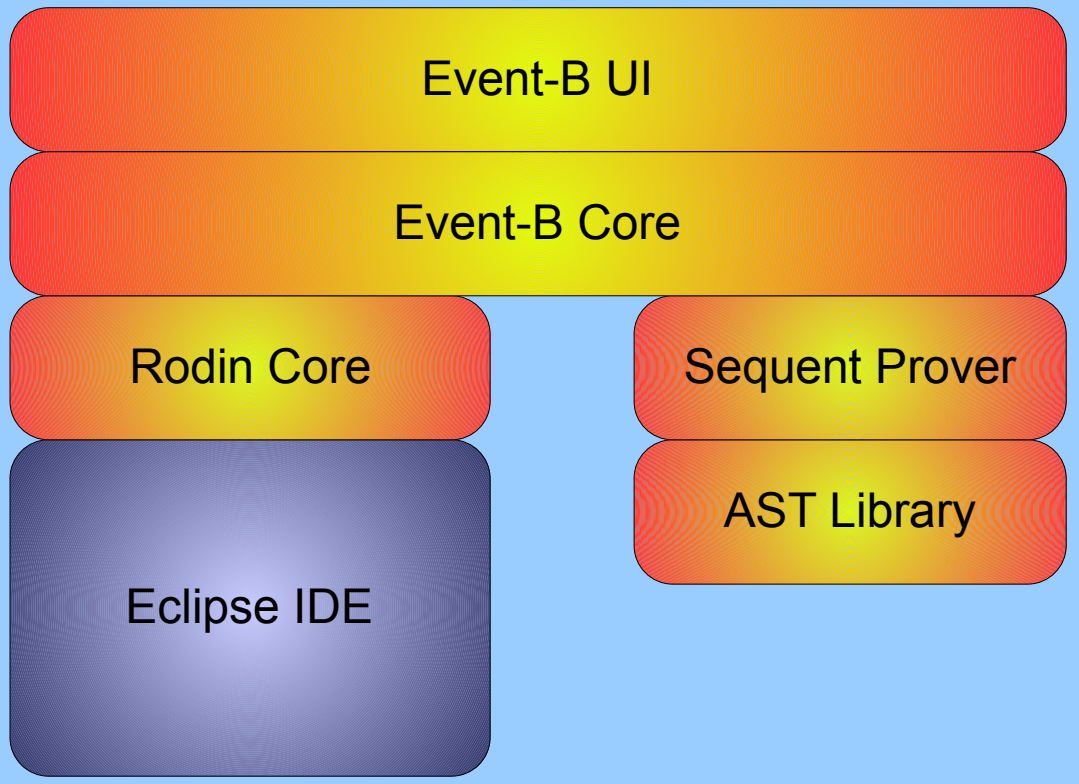
Contributing to the Rodin Platform

- Laurent Voisin (Systerel)

Contents

- Rodin Architecture
- Plug-ins and features
- The Rodin Database
- Contributing to the Event-B UI
- Tutorial plug-in

Rodin Architecture



Rodin Features (1/2)

- org.rodinp
  - depends on org.eclipse.platform
  - contributes the database and builder
- org.eventb.ide
  - depends on org.rodinp
  - contributes all stuff for event-B (core and UI)

Rodin Features (2/2)

- org.rodinp.platform
  - provides platform branding
- com.clearsy.atelierb.provers
  - depends on org.eventb.ide
  - contributes the Atelier B provers

Rodin Plug-ins

- Three kinds of plug-ins
  - core plug-ins (can be run headless)
  - UI plug-ins
  - branding plug-ins (same name as feature or product)

The Rodin Database

- Hierarchical repository (tree structured)
- Contains elements and attributes
- Elements capture the hierarchical structure
- Attributes are just annotations to certain elements
- Elements and attributes are typed
- All accesses are done through handles

Database Element Types

- Define the implementing Java classes
- Eclipse resource counterparts
  - RodinDB ~ WorkspaceRoot
  - Rodin project ~ Project
  - Rodin file ~ File
- Internal elements contributed by clients
- A file contains a single root element

Accessing the Database

- You need a handle to an element
- From an Eclipse resource:
  - RodinCore.valueOf(resource)
- From a Rodin project:
  - rodinProject.getRodinFile(fileName)
- From a Rodin file:
  - rodinFile.getRoot()
- From an internal element:
  - parent.getInternalElement(eType, eName)

Browsing the Database

- Handle-only method;
  - element.getElementName()
  - element.getElementType()
  - element.getParent()
- Traversing
  - element.exists()
  - element.getChildren()
  - element.getChildrenOfType(elementType)

Snapshots and Mutable Copies

- For Rodin files and internal elements
- Snapshot
  - always corresponds to version on disk
  - read-only
- Mutable Copy
  - in memory copy (desynchronized from disk)
  - can be modified

Attribute Types

- Have a unique id
- Have a kind defining their Java type:
  - boolean
  - handle
  - integer
  - long
  - string

Manipulating Attributes

- Only internal elements can carry attributes
- iElement.getAttributeNames()
- iElement.hasAttribute(aType)
- iElement.getAttributeValue(aType)
- iElement.setAttributeValue(aType, aValue, pm)
- iElement.removeAttribute(aType)

Reacting to Changes

- The Database implements the Observer design pattern
- Runs registered listeners on database changes
- Delta tree gives details of change
- RodinCore.addElementChangedListener(listener)

Mementos

- Handles can be serialized to a String
- element.getHandleMemento()
- RodinCore.valueOf(memento)

Runnables

- Similar to Workspace runnable
- Managed by the database
- RodinCore.run(runnable)
- Allow for atomic access (locking)

Modifying the Database

- element.create(...)
- element.copy / move / rename / delete (...)
- file.hasUnsavedChanges()
- file.save(...)
- file.makeConsistent(...)
- see also attribute manipulations

Contributing an Internal Element Type

- Extend org.rodinp.core.internalElementTypes
  - unique id
  - implementing class extending InternalElement
- Also provide an interface for clients

Contributing a Root Element Type

- First, define an Eclipse content-type
  - must subtype org.rodinp.core.rodin
  - must be based on file name extension
- Then, extend org.rodinp.core.fileAssociations
  - content-type
  - id of root element type

Contributing an Attribute Type

- Extend org.rodinp.core.attributeTypes
  - unique id
  - Java type (boolean, int, long, IRodinElement, String)

Important points

- Classes are implementing handles
  - Must be immutable (no state)
- Be careful not to mess up the database
  - Don't use any internal method
- Make two packages
  - org.xyz.core for interfaces
  - org.xyz.core.basis for classes

Contributing to the Rodin Builder

- Extension point org.rodinp.core.autoTools
- tool to get the work done
- extractor(s) to get the dependencies

Contributing to the Event-B UI

- Three extension points
- editorPages for adding pages to the editor
- proofTactics for contributing interactive proof commands
- editorItems for contributing additional elements or attributes

How to Start

- Look at existing implementations
- For instance, in org.eventb.core
- Post questions to the developer's mailing list

Setting up Rodin Sources

- Install Eclipse 3.4 on your computer
- Start Eclipse on an empty workspace
- Fetch the sources of Rodin 1.0 from SourceForge
- In Eclipse
  - click File > Import...
  - select General > Existing Projects into Workspace
  - enter path of archive containing sources
  - click Finish

Metrics Plug-in First Round

- Import project from archive tutorial-1.zip
- Then create plug-in org.eventb.metrics
- And fill it to make the tests pass

- Demonstrates reading from the database

Metrics Plug-in Second Round

- Import project from archive tutorial-2.zip
- Then update plug-in org.eventb.metrics to make the tests pass

- Demonstrates adding new element types to the database

Metrics Plug-in Third Round

- Import project from archive tutorial-3.zip
- Then update plug-in org.eventb.metrics to make the tests pass

- Demonstrates contributing to the builder