Event-B User Interfaces

Thai Son Hoang

Department of Computer Science
Swiss Federal Institute of Technology Zürich (ETH Zürich)

RODIN Meeting, 28th-29th March 2007, Southampton

What are the interfaces

- Modelling Interface: Entering the Event-B models.
- Proving Interface: Interactive proving the obligations.

Outline

Outline

Rodin

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Modelling Interface - Functionality

- Follow the standard Eclipse layout.
- There are several views:
  - *Project Explorer:* Tree-structured views of the projects.
  - *Content Outline:*
    - Reflects the structure;
    - provides quick navigation

  for the current editing editing component.
- and the *Event-B Editor:*
  - Multi-page,
  - Form editor.

Modelling Interface - Functionality

- Follow the standard Eclipse layout.
- There are several views:
  - *Project Explorer:* Tree-structured views of the projects.
  - *Content Outline:*
    - Reflects the structure;
    - provides quick navigation

    for the current editing editing component.
- and the *Event-B Editor:*
  - Multi-page,
  - Form editor.

Rodin

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Modelling Interface - Functionality

- Follow the standard Eclipse layout.
- There are several views:
  - *Project Explorer:* Tree-structured views of the projects.
  - *Content Outline:*
    - Reflects the structure;
    - provides quick navigation

    for the current editing editing component.
- and the *Event-B Editor:*
  - Multi-page,
  - Form editor.

Modelling Interface - Functionality

- Follow the standard Eclipse layout.
- There are several views:
  - *Project Explorer:* Tree-structured views of the projects.
  - *Content Outline:*
    - Reflects the structure;
    - provides quick navigation

  for the current editing editing component.
- and the *Event-B Editor:*
  - Multi-page,
  - Form editor.

Rodin

ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Event-B Editor

- Old editor: Table/Tree Editor.
  - Too different from classical Text Editor.
  - No support for multi-line editing.
  - Elements can be added but not attributes.

- Current developing editor: Text-like Editor.
  - More familiar with users.
  - Supporting multi-line editting.
  - Extension (both elements and attributes) is easy.

Event-B Editor

- Old editor: Table/Tree Editor.
  - Too different from classical Text Editor.
  - No support for multi-line editing.
  - Elements can be added but not attributes.
- Current developing editor: Text-like Editor.
  - More familiar with users.
  - Supporting multi-line editting.
  - Extension (both elements and attributes) is easy.

Event-B Editor

- Old editor: Table/Tree Editor.
  - Too different from classical Text Editor.
  - No support for multi-line editing.
  - Elements can be added but not attributes.
- Current developing editor: Text-like Editor.
  - More familiar with users.
  - Supporting multi-line editting.
  - Extension (both elements and attributes) is easy.

Event-B Editor

- Old editor: Table/Tree Editor.
  - Too different from classical Text Editor.
  - No support for multi-line editing.
  - Elements can be added but not attributes.

- Current developing editor: Text-like Editor.
  - More familiar with users.
  - Supporting multi-line editting.
  - Extension (both elements and attributes) is easy.

Event-B Editor

- Old editor: Table/Tree Editor.
  - Too different from classical Text Editor.
  - No support for multi-line editing.
  - Elements can be added but not attributes.
- Current developing editor: Text-like Editor.
  - More familiar with users.
  - Supporting multi-line editting.
  - Extension (both elements and attributes) is easy.

Event-B Editor

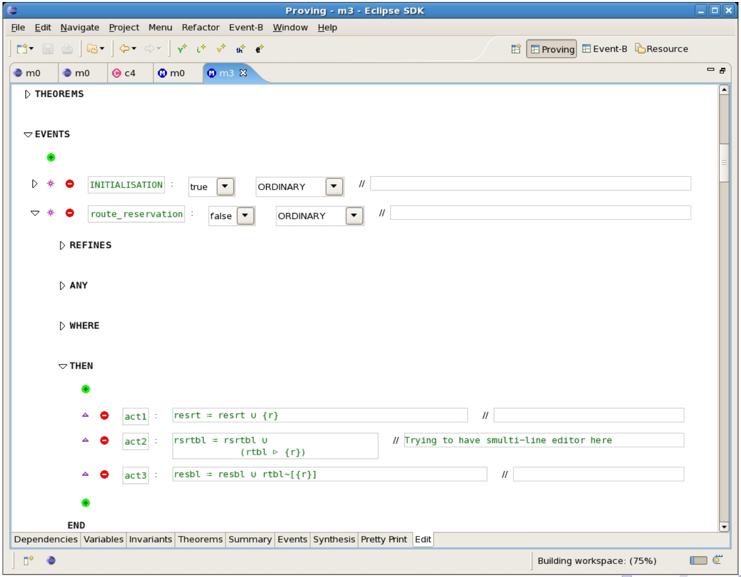
- Old editor: Table/Tree Editor.
  - Too different from classical Text Editor.
  - No support for multi-line editing.
  - Elements can be added but not attributes.

- Current developing editor: Text-like Editor.
  - More familiar with users.
  - Supporting multi-line editting.
  - Extension (both elements and attributes) is easy.

Event-B Editor

- Old editor: Table/Tree Editor.
  - Too different from classical Text Editor.
  - No support for multi-line editing.
  - Elements can be added but not attributes.
- Current developing editor: Text-like Editor.
  - More familiar with users.
  - Supporting multi-line editting.
  - Extension (both elements and attributes) is easy.

Event-B Editor Screen-shot


Proving - m3 - Eclipse SDK
File Edit Navigate Project Menu Refactor Event-B Window Help
Proving Event-B Resource
m0 m0 c4 m0 m3
Dependencies Variables Invariants Theorems Summary Events Synthesis Pretty Print Edit
Building workspace: (75%)

```
▷ THEOREMS

▽ EVENTS

  ▷ INITIALISATION : true ORDINARY //
  ▽ route_reservation : false ORDINARY //

      ▷ REFINES
      ▷ ANY
      ▷ WHERE
      ▽ THEN

          act1 : resrt ≔ resrt ∪ {r}            //
          act2 : rsrtbl ≔ rsrtbl ∪              // Trying to have smulti-line editor here
                     (rtbl ▷ {r})
          act3 : resbl ≔ resbl ∪ rtbl∼[{r}]     //

     END
```

Rodin
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Proving Interface - Functionality

- Follows standard *Eclipse* layout.
- Based on *Click'N'Prove* with improvements.
- There are several views:
  - Proof Tree: Tree-structured views of the current proof.
  - Proof Control: Issues proof command to discharge the obligation.
  - Proof Information: Shows related information to the current proof.
  - Search Hypothesis: Shows set of searched hypotheses.
  - Obligation Explorer: Shows the tree-like view of all proof obligations.
- and a Proof Editor.
  - Displays the current state of the proof: goal and hypotheses.
  - Issues proof commands either directly or indirectly on the formula.

Proving Interface - Functionality

- Follows standard *Eclipse* layout.
- Based on *Click'N'Prove* with improvements.
- There are several views:
  - Proof Tree: Tree-structured views of the current proof.
  - Proof Control: Issues proof command to discharge the obligation.
  - Proof Information: Shows related information to the current proof.
  - Search Hypothesis: Shows set of searched hypotheses.
  - Obligation Explorer: Shows the tree-like view of all proof obligations.
- and a Proof Editor.
  - Displays the current state of the proof: goal and hypotheses.
  - Issues proof commands either directly or indirectly on the formula.

Proving Interface - Functionality

- Follows standard *Eclipse* layout.
- Based on *Click'N'Prove* with improvements.
- There are several views:
  - Proof Tree: Tree-structured views of the current proof.
  - Proof Control: Issues proof command to discharge the obligation.
  - Proof Information: Shows related information to the current proof.
  - Search Hypothesis: Shows set of searched hypotheses.
  - Obligation Explorer: Shows the tree-like view of all proof obligations.
- and a Proof Editor.
  - Displays the current state of the proof: goal and hypotheses.
  - Issues proof commands either directly or indirectly on the formula.

Rodin

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Proving Interface - Functionality

- Follows standard *Eclipse* layout.
- Based on *Click'N'Prove* with improvements.
- There are several views:
  - Proof Tree: Tree-structured views of the current proof.
  - Proof Control: Issues proof command to discharge the obligation.
  - Proof Information: Shows related information to the current proof.
  - Search Hypothesis: Shows set of searched hypotheses.
  - Obligation Explorer: Shows the tree-like view of all proof obligations.
- and a Proof Editor.
  - Displays the current state of the proof: goal and hypotheses.
  - Issues proof commands either directly or indirectly on the formula.

Proving Interface - Functionality

- Follows standard *Eclipse* layout.
- Based on *Click'N'Prove* with improvements.
- There are several views:
  - Proof Tree: Tree-structured views of the current proof.
  - Proof Control: Issues proof command to discharge the obligation.
  - Proof Information: Shows related information to the current proof.
  - Search Hypothesis: Shows set of searched hypotheses.
  - Obligation Explorer: Shows the tree-like view of all proof obligations.
- and a Proof Editor.
  - Displays the current state of the proof: goal and hypotheses.
  - Issues proof commands either directly or indirectly on the formula.

Proving Interface - Functionality

- Follows standard *Eclipse* layout.
- Based on *Click'N'Prove* with improvements.
- There are several views:
  - Proof Tree: Tree-structured views of the current proof.
  - Proof Control: Issues proof command to discharge the obligation.
  - Proof Information: Shows related information to the current proof.
  - Search Hypothesis: Shows set of searched hypotheses.
  - Obligation Explorer: Shows the tree-like view of all proof obligations.
- and a Proof Editor.
  - Displays the current state of the proof: goal and hypotheses.
  - Issues proof commands either directly or indirectly on the formula.

Proving Interface - Functionality

- Follows standard *Eclipse* layout.
- Based on *Click'N'Prove* with improvements.
- There are several views:
  - Proof Tree: Tree-structured views of the current proof.
  - Proof Control: Issues proof command to discharge the obligation.
  - Proof Information: Shows related information to the current proof.
  - Search Hypothesis: Shows set of searched hypotheses.
  - Obligation Explorer: Shows the tree-like view of all proof obligations.
- and a Proof Editor.
  - Displays the current state of the proof: goal and hypotheses.
  - Issues proof commands either directly or indirectly on the formula.

Proving Interface - Functionality

- Follows standard *Eclipse* layout.
- Based on *Click'N'Prove* with improvements.
- There are several views:
  - Proof Tree: Tree-structured views of the current proof.
  - Proof Control: Issues proof command to discharge the obligation.
  - Proof Information: Shows related information to the current proof.
  - Search Hypothesis: Shows set of searched hypotheses.
  - Obligation Explorer: Shows the tree-like view of all proof obligations.
- and a Proof Editor.
  - Displays the current state of the proof: goal and hypotheses.
  - Issues proof commands either directly or indirectly on the formula.

Proving Interface - Extensions

“Proof commands” can be added to the proving interface.

- Globally: added to the Proof Control View.
- Goal: Directly / Indirectly in the predicate.
- Hypothesis: Directly / Indirectly in the predicate.

Justifications

- Correctness
  - Using Model-View-Controller pattern.
  - Unit tests for underlying model.
  - Tree structure is based on database layout.
- Efficiency
  - Editor is designed for efficiency updates in common cases.
  - Lazy loading of extensions
  - Sharing UI resources: icons, etc.
- Maintenance
  - Extension loading is encapsulated.
  - Restrict possible extensions.
    - Declarative.
    - Very little coding.

Rodin

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Justifications

- Correctness
  - Using Model-View-Controller pattern.
  - Unit tests for underlying model.
  - Tree structure is based on database layout.
- Efficiency
  - Editor is designed for efficiency updates in common cases.
  - Lazy loading of extensions
  - Sharing UI resources: icons, etc.
- Maintenance
  - Extension loading is encapsulated.
  - Restrict possible extensions.
    - Declarative.
    - Very little coding.

Rodin

ETH Eidgenössische Technische Hochschule Zürich Swiss Federal Institute of Technology Zurich

Justifications

- Correctness
  - Using Model-View-Controller pattern.
  - Unit tests for underlying model.
  - Tree structure is based on database layout.
- Efficiency
  - Editor is designed for efficiency updates in common cases.
  - Lazy loading of extensions
  - Sharing UI resources: icons, etc.
- Maintenance
  - Extension loading is encapsulated.
  - Restrict possible extensions.
    - Declarative.
    - Very little coding.

Outline

Rodin

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Modelling Interface - High priority

- Finishing the new editor.
- Displaying undefined attributes.
- Error markers.
- User Documents.
- Plug-in Developer's Guideline.
- Copy/Paste.
- Undo/Redo.

Modelling Interface - Low priority

- Re-factoring.
- Content assist.
- Search elements.
- Quick fixes for errors.
- Project Explorer (using Common Navigator Framework)
- Hierarchy View.
- Improving icons.

Proving Interface

- Keep hypotheses order (High priority).
- Display forward reasoning (Low priority).